Computational Methods in Applied Sciences

Sergio R. Idelsohn *Editor*

Numerical Simulations of Coupled Problems in Engineering

ECCOMAS

European Community on Computational Methods in Applied Sciences

Springer

Numerical Simulations of Coupled Problems in Engineering

Computational Methods in Applied Sciences

Volume 33

Series editor

E. Oñate
International Center for Numerical Methods in Engineering (CIMNE)
Technical University of Catalonia (UPC)
Edificio C-1, Campus Norte UPC
Gran Capitán, s/n
08034 Barcelona, Spain
e-mail: onate@cimne.upc.edu
url: http://www.cimne.com

Sergio R. Idelsohn
Editor

Numerical Simulations of Coupled Problems in Engineering

Springer

Editor
Sergio R. Idelsohn
International Center for Numerical
Methods in Engineering (CIMNE)
Catalan Institution for Research
and Advanced Studies (ICREA)
Barcelona
Spain

ISSN 1871-3033
ISBN 978-3-319-06135-1 ISBN 978-3-319-06136-8 (eBook)
DOI 10.1007/978-3-319-06136-8
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014939045

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This book contains state-of-the-art contributions in the field of Coupled Problems in Engineering. A selected specialist has written each chapter as an extended version of the paper presented at the conference “Fifth Computational Methods for Coupled Problems in Science and Engineering” held in Ibiza in July 2013. This Conference brought together more than 400 participants from 41 countries and was dedicated to celebrate the 60th birthday of Prof. Eugenio Oñate.

The Conference was included as one of the Thematic Conferences of the European Community on Computational Methods in Applied Sciences (ECCOMAS) and a Special Interest Conference of the International Association for Computational Mechanics (IACM). It was also supported by other scientific organizations in Europe and worldwide.

This book contains 16 chapters written by distinguished authors, who present and discuss mathematical models, numerical methods, and computational techniques for solving Coupled Problems of multidisciplinary character. The goal of this book is to take a step forward in the formulation and solution of real life problems with a multidisciplinary vision, accounting for all the complex couplings involved in the physical description of the problem.

Topics treated in the various chapters include developments and applications of Coupled Problems in a wide variety of situations such as Non-Linear Materials, Cardiovascular Fluid Mechanics, Multi-Fluid Flows, or Fluid-Structure Interactions, using different techniques like particle methods, reduced order models or partitioned parallelization techniques.

This book includes contributions submitted directly by authors. The editor cannot accept responsibility for any inaccuracies, comments, and opinions contained in the text.

The editor would like to take this opportunity for thanking all authors for submitting their excellent contributions on time. Many thanks also to ECCOMAS and Springer for accepting the publication of this book in the series “Computational Methods in Applied Sciences.”

Sergio R. Idelsohn

Contents

Part I
Non-Linear Materials in Coupled Problems

Generalized Viscoplasticity Based on Overstress (GVBO) for Large Strain Single-Scale and Multiscale Analyses

Vasilina Filonova, Yang Liu and Jacob Fish

Abstract A generalized viscoplasticity based on the overstress (GVBO) model that successfully reproduces large strain and strain rate experimental data has been developed. The GVBO model confirmed increased shear resistance of polyurea at very high strain rates (10^5–$10^6\,\mathrm{s}^{-1}$) observed in the experiments. With the proposed GVBO model, we conducted numerical simulation of fragment simulating projectile impacting polyurea/ high-hard-steel bi-layers at high impact velocities (>1000 m/s). For model validation, two positions of the polymer coating with respect to the steel plate have been considered: the front and back coating, with a front side being a target. Numerical impact simulations utilizing a single-scale GVBO model predicted that a polyurea bi-layer with a front coating increases penetration velocity by about 15.4 % (against 23 % observed in the experiments), while the steel plate with a back coating raises penetration velocity by 7.5 % (as opposed to 8.8 % in the experiment) in comparison to the ballistic limit of the blank steel plate. This minor discrepancy between experimental and simulation results is qualitatively attributed to the space-time multiscale phenomena. We show that a possible anisotropy introduced by material heterogeneity increases resistance of the polyurea layer by partially transforming the pressure wave into a dissipated shear wave. We further demonstrate that dispersion further enhances energy absorption of the polyurea coating.

Keywords Polyurea · Viscoplasticity model based on the overstress · Multiscale · Anisotropy · Dispersion

V. Filonova · Y. Liu · J. Fish (✉)
Columbia University, Newyork, USA
e-mail: fishj@columbia.edu

1 Introduction

The present manuscript focusses on a single-scale and multiscale modeling of copolymers. A copolymer is a polymer derived from two (or more) monomeric species, as opposed to a homopolymer where only one monomer is used. Polyurea, which is a generic term for a block copolymer, comprises of homopolymer subunits linked by covalent bonds. The union of the homopolymer subunits may require an intermediate non-repeating subunit, known as a junction block. Polyurea exhibits a wide range of mechanical properties, from soft rubber to hard plastic depending on the chemistry. The range of properties together with their rapid reaction has led to many applications as coatings, for example on tunnels, bridges, roofs, parking decks, storage tanks, freight ships, truck beds, etc. Polyurea coatings have been applied to military armor to increase its resistance to ballistic penetration [1].

Many copolymer models [2–6] are derived from the standard linear solid model (SLS), capable of predicting both creep and relaxation as shown in Fig. 1.

The arm 1 in Fig. 1 is a hyperelastic arm, whereas the bottom arm in Fig. 1 is the Maxwell arm consisting of a dashpot and hyperelastic spring connected in series. For large strain problems [5, 6], the deformation gradient in the bottom arm, $\mathbf{F}_2 = \mathbf{F}_1 \equiv \mathbf{F}$, which is identical to the deformation gradient in the top arm, is decomposed into elastic and inelastic deformation gradients

$$\mathbf{F}_2 = \mathbf{F}_2^e \mathbf{F}_2^{in} \tag{1}$$

The elastic response in the hyperelastic arm is assumed to obey Neo-Hookean law, whereas the stress in the Maxwell arm $\boldsymbol{\sigma}_2$ depends on left stretch tensor $\mathbf{V}_2^e$ and pressure p_2 in the Maxwell arm, i.e. $\boldsymbol{\sigma}_2 = f\left(\mathbf{V}_2^e, p_2\right)$. For more details we refer to [5, 6]. These models, however, have been considered for up to the strain rates of $10^4\ \mathrm{s}^{-1}$.

The primary objectives of the present chpater are as follows:

- Develop a block copolymer constitutive model for large strains and strain rates in the range of 10^5–$10^6\ \mathrm{s}^{-1}$. Rather than utilizing the framework of multiplicative decomposition (1), we will pursue an additive decomposition of the rate of deformation
$$\mathbf{D} = \mathbf{D}^e + \mathbf{D}^{in} \tag{2}$$

  The evolution of the inelastic responses will follow the framework of the VBO model [7, 8] with only exception that the elastic constants and viscosity will be assumed to be deformation-dependent. While there are a number of available models of polyurea in the literature [9], to our knowledge, there is no published work utilizing the framework of viscoplasticity model based on overstress (VBO) to describe exceedingly large strains and strain rates observed in the experiments. In the following we will refer to the generalization of the VBO model to large strains and strain rates as the GVBO.

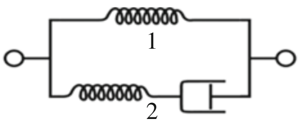


Fig. 1 Rheological model of the polymer. Arm 1—spin model, arm 2—Maxwell arm consisting of spring and dashpot

- Validate the proposed GVBO constitutive model. We will consider recent experiments conducted at Brown University (impact of a steel flyer on a steel/polyurea/steel sandwich plate under high shear strain rates) [10–12].
- Validate the GVBO model on the structural impact problem. We will consider the impact of the projectile onto the polyurea/steel bi-layer where a polymer layer is placed on the front or back of the plate. We will verify experimental observation suggesting considerable advantage of placing the polyurea layer on the top of the target plate at very high impact velocities (>1000 m/s) [13–15]. It is noteworthy to point out that for low to moderate impact velocities the back polyurea coating of the plate shows better resistance than the front coating [16].
- The last part of the manuscript will consider space-time multiscale analysis of the block copolymer to explain some of the discrepancies of the single-scale GVBO model. The polyurea microstructure will be modeled as a two-phase heterogeneous anisotropic material with the inclusion phase being elastic and the soft domains obeying GVBO constitutive model. The multiple temporal scale effect, which gives rise to dispersion, will be taken into account using the recently developed micro-inertia approach [17].

2 Generalized Viscoplasticity Based on Overstress (GVBO) for Large Strains

In this section we present the generalized viscoplasticity model based on overstress (GVBO) for large strain and strain rate problems. First, in Sect. 2.1, the model is stated in the corotational (or rotation-free) frame. Subsequently, in Sects. 2.2 and 2.3, we develop a deformation-dependent elastic modulus and viscosity function for the classical VBO model [7, 8].

2.1 VBO for Large Rotations

We start form the constitutive equation based on the additive decomposition of the rate of deformation

$$\overset{\circ}{\boldsymbol{\sigma}} = \mathbf{L} : \left(\mathbf{D} - \mathbf{D}^{in}\right) \tag{3}$$

where $\overset{\circ}{\boldsymbol{\sigma}}$ is an objective Cauchy stress rate; **D** is the rate of deformation tensor, and $\mathbf{D}^{in}$ is the inelastic rate of deformation. The elastic constitutive tensor **L** for isotropic material is given by

$$\mathbf{L} = \lambda \mathbf{1} \otimes \mathbf{1} + 2G\mathbf{I} \tag{4}$$

where λ is the first Lame constant, and G is the shear modulus.

To account for large rotations, we consider the Green-Naghdi rate of Cauchy stress, which describes material response in the initial rotation-free frame. The Green-Naghdi rate of the Cauchy stress is defined by

$$\overset{\circ}{\boldsymbol{\sigma}} = \dot{\boldsymbol{\sigma}} + \boldsymbol{\sigma}\left(\dot{\boldsymbol{R}}\boldsymbol{R}^T\right) - \left(\dot{\boldsymbol{R}}\boldsymbol{R}^T\right)\boldsymbol{\sigma} \tag{5}$$

The flow rule for the case of finite rotations but small strains is given by

$$\mathbf{d} = \mathbf{d}^e + \mathbf{d}^{in} = \frac{1+\nu}{E}\overset{\circ}{\mathbf{s}} + \frac{3}{2}\frac{\mathbf{s}-\mathbf{g}}{Ek} \tag{6}$$

where **d** is the deviatoric part of the rate of deformation tensor **D** decomposed into elastic and inelastic parts, $\mathbf{d}^e$ and $\mathbf{d}^{in}$, respectively. **s** is the deviatoric Cauchy stress tensor; **g** is the deviatoric part of the equilibrium stress; E is the Young's modulus; ν is the Poisson's ratio, and k is the viscosity function defined as

$$k = k_1\left(1 + \frac{\Gamma}{k_2}\right)^{-k_3} \tag{7}$$

where k_1, k_2, k_3 are material constants. The overstress invariant is defined by

$$\Gamma = \sqrt{\frac{2}{3}(\mathbf{s}-\mathbf{g}) : (\mathbf{s}-\mathbf{g})} \tag{8}$$

The evolution law for the equilibrium stress is given by

$$\overset{\circ}{\mathbf{g}} = \frac{\psi}{E}\left(\overset{\circ}{\mathbf{s}} + \frac{\mathbf{s}-\mathbf{g}}{k} - \frac{\Gamma}{k}\frac{(\mathbf{g}-\mathbf{f})}{A}\right) + \left(1 - \frac{\psi}{E}\right)\overset{\circ}{\mathbf{f}} \tag{9}$$

where ψ is the shape function defined as

$$\psi = a_1 + (a_2 - a_1)\exp(-a_3\Gamma) \tag{10}$$

with a_1, a_2, a_3 being material constants.

The evolution laws of the kinematic stress function $\overset{\circ}{\mathbf{f}}$ is defined as

$$\overset{\circ}{\mathbf{f}} = \bar{E}_t\frac{\mathbf{s}-\mathbf{g}}{Ek}; \quad \bar{E}_t \equiv \frac{2}{3}\frac{E_t}{(1 - E_t/E)} \tag{11}$$

and of the isotropic stress function $\dot{A}$ as

$$\dot{A} = A_c \left(A_f - A\right) \frac{\Gamma}{Ek}; \quad A[t = 0] = A_0 \tag{12}$$

where A_c, A_f are material constants.

For more details about the classical VBO model see [7, 8].

2.2 Deformation-Dependent Elastic Constitutive Tensor

We introduce the following deformation-dependent Lame constants related to small strain elastic constitutive tensor (4)

$$\begin{aligned} \lambda[J] &= J^p \lambda_0 \\ G[J] &= J^p \left(G_0 - \lambda_0 \ln J\right) \end{aligned} \tag{13}$$

where λ_0, G_0 are the initial (small deformation) values and p is a material parameter; the Jacobian is defined as usual by $J = \det(\mathbf{F})$ where $\mathbf{F}$ is the deformation gradient tensor. The elastic constitutive tensor (4) is assumed to be a function of the Jacobian

$$\mathbf{L}[J] = \lambda[J]\mathbf{1} \otimes \mathbf{1} + 2G[J]\mathbf{I} \tag{14}$$

The initial elastic properties E_0, ν_0 are related to the initial Lame constants by

$$\lambda_0 = \frac{E_0 \nu_0}{(1 + \nu_0)(1 - 2\nu_0)}; \quad G_0 = \frac{E_0}{2(1 + \nu_0)} \tag{15}$$

The deformation-dependent Young's modulus and Poisson ratio are related to deformation-dependent Lame parameters (13) by

$$E[J] = \frac{G[J]\left(3\lambda[J] + 2G[J]\right)}{\lambda[J] + G[J]}; \quad \nu[J] = \frac{\lambda[J]}{2\left(\lambda[J] + G[J]\right)} \tag{16}$$

The elastic parameters are allowed to increase only from their initial values: $G \geq G_0$, $\lambda \geq \lambda_0$, $E \geq E_0$; i.e. if elastic parameters decrease based on Eq. (13), the initial values are taken instead.

2.3 The Deformation-Dependent Viscosity Function

We assume the viscosity k to be a function of overstress and Jacobian as follows

Table 1 Generalized VBO model summary

GVBO	Parameters 15: $E_0, \nu_0, E_t, p, a_1, a_2, a_3, k_1, k_2, k_3, k_4, E_m, A_0, A_c, A_f$
Elastic moduli	$E[J] = \frac{G[J](3\lambda[J]+2G[J])}{\lambda[J]+G[J]}$ $\lambda[J] = J^p \lambda_0; \quad G[J] = J^p (G_0 - \lambda_0 \ln J)$ $\lambda_0 = \frac{E_0 \nu_0}{(1+\nu_0)(1-2\nu_0)}; \quad G_0 = \frac{E_0}{2(1+\nu_0)}$
Flow law	$\mathbf{d} = \frac{1+\nu}{E[J]} \overset{\circ}{\mathbf{s}} + \frac{3}{2} \frac{\mathbf{s}-\mathbf{g}}{E[J]k[\Gamma,J]}$
Equilibrium stress evolution law	$\overset{\circ}{\mathbf{g}} = \frac{\psi}{E[J]} \left(\overset{\circ}{\mathbf{s}} + \frac{\mathbf{s}-\mathbf{g}}{k[\Gamma,J]} - \frac{\Gamma}{k[\Gamma,J]} \frac{(\mathbf{g}-\mathbf{f})}{A} \right) + \left(1 - \frac{\psi}{E[J]} \right) \overset{\circ}{\mathbf{f}}$
Kinematic stress evolution law	$\overset{\circ}{\mathbf{f}} = \bar{E}_t[J] \frac{\mathbf{s}-\mathbf{g}}{E[J]k[\Gamma,J]}; \quad \bar{E}_t[J] \equiv \frac{2E_t}{3(1-E_t/E[J])}$
Isotropic stress function evolution law	$\dot{A} = A_c \left(A_f - A \right) \frac{\Gamma}{E[J]k[\Gamma,J]}; \quad A\,(t=0) = A_0$
Shape function	$\psi = a_1 + (a_2 - a_1)\, e^{-a_3 \Gamma}$
Viscosity function	$k[\Gamma, J] = k_1 \left(1 + \frac{\Gamma}{\bar{k}_2[J]} \right)^{-k_4}; \quad \bar{k}_2[J] \equiv k_2 + k_3 \frac{E[J]-E_0}{E_m-E_0}$

$$k[\Gamma, J] = k_1 \left(1 + \frac{\Gamma}{\bar{k}_2[J]} \right)^{-k_4}; \quad \bar{k}_2[J] \equiv k_2 + k_3 \frac{E[J]-E_0}{E_m-E_0} \tag{17}$$

where k_1, k_2, k_3, k_4, E_m are material parameters. Note that $\bar{k}_2 = k_2$ is constant when Young's modulus is constant, i.e. the deformation is small. The parameter E_m is a maximum Young's modulus and k_3 defines the sensitivity to large deformation.

The generalized VBO model for the finite deformation theory is summarized in Table 1.

3 Model Validation

3.1 Experimental Setup

The experiments investigating resistance of elastomer to shearing and failure at extreme loading conditions were conducted by Clifton and coworkers [10–12]. The polyurea is cast between two steel plates. The flyer impacts the sandwich plate at high velocity. For pressure-shear tests, the flyer is slightly inclined at angle $\theta = 18°$ to produce a shear response of the polyurea as shown in Fig. 2. We consider two pressure-shear experiments: Shot#1201 and Shot#404 at shear-strain rates of $4.1 \times 10^5\ \text{s}^{-1}$ and $2.4 \times 10^5\ \text{s}^{-1}$, respectively. The main difference between the two shots is impact velocity (see Table 3), which is higher for Shot#1201. In addition, we consider a pure compression experiment, Shot#1203, which is required to identifying pure compression stress-strain relation. The geometry, material properties and impact velocities are described in Tables 2 and 3, (for details see references [10, 11]).

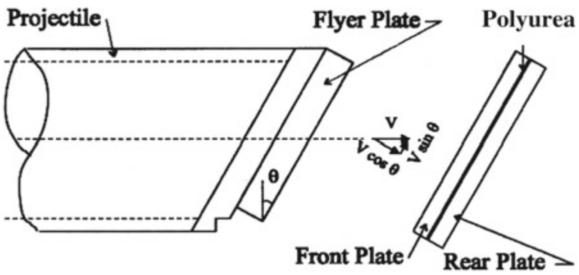


Fig. 2 Experiment setting [10]

Table 2 Flyer and plate steel material parameters

Shot No.	Steel	Density (g/cm^3)	Young's modulus (MPa)	Poisson's ratio
1201, 1203	Pure WC	15.4	609100	0.2
404	Hampden steel	7.861	213700	0.29

Table 3 Thickness of the flyer, sandwich plates, and impact velocity for two experiments

Shot No.	Sample –polyurea (mm)	Front plate (mm)	Rare plate (mm)	Flyer (mm)	Impact velocity V_0 (m/s)	Angle θ (°)
1201	0.097	3.582	5.578	7.411	183.45	18
1203	0.089	3.588	5.898	10.5	175	0
404	0.11	2.896	7.041	6.991	112.6	18

3.2 Numerical Simulation

For numerical simulations we consider an idealized model: a narrow longitudinal strip (in 3D) of a sandwich and a flyer as shown in Fig. 3. Such an idealization was originally proposed in reference [18]. The polyurea is modeled by one 8-node hexahedral solid element. The flyer and the steel plates were meshed by 8-node hexahedral solid elements and modeled as elastic material with parameters described in Table 2. The boundary and initial conditions are listed in Table 4. The calibrated material parameters of the GVBO model of polyurea are listed in Table 5. The parameter E_m is taken to be the maximum Young's modulus observed in Shot#404. The parameters $k_1, \ldots, k_4$ and p are calibrated against the experimental data from Shots#1201 and #404. Note that we use the same set of polyurea parameters for both experiments in Shot#1201 and Shot#404.

We also consider cohesive elements at the interfaces between the polymer layer and the steel. The cohesive element is described by an anisotropic traction-separation law. The damage initiates at a maximum effective stress, and the total damage corresponds to the maximum effective displacement as shown in Table 6 . We use different parameters for the cohesive layer in the experiments Shot#1201 and Shot#404 since

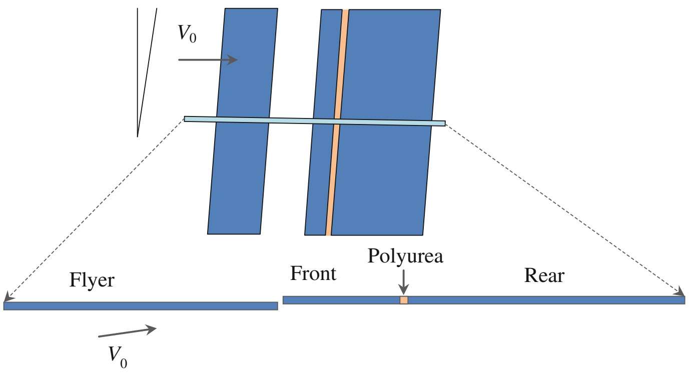


Fig. 3 Simplified numerical simulation model [18]

Table 4 Boundary and initial conditions considered in numerical simulation

Flyer plate	Sandwich plate assembly
Initial velocity: $V_1 = V_0 \cos\theta$, $V_2 = V_0 \sin\theta$	Constraints: $u_1 = \text{free}$, $u_{2\_top} = u_{2\_bottom}$, $u_3 = 0$
Constraints: $u_{2\_top} = u_{2\_bottom}$; $u_3 = 0$	

Table 5 GVBO material parameters for polyurea

Density (g/cm^3)	E_0 (MPa)	ν_0	E_t (MPa)	p	a_1 (MPa)	a_2 (MPa)	a_3 (1/MPa)
1.1	278	0.491	23.3	−6	0	10	0
k_1	k_2 (MPa)	k_3 (MPa)	k_4	E_m (MPa)	A_0 (MPa)	A_c (s^{-1})	A_f (MPa)
17	0.47	1.47	4	13600	1	0	0

Table 6 Material parameters for cohesive interface modeled by traction-separation law

Shot No.	Density (g/cm^3)	Elastic properties (MPa)			Maximum nominal stress (MPa)			Maximum effective displacement (mm)
		E	G_1	G_2	S_n	S_1	S_2	
1201, 1203	1.56	12000	14000	14000	6000	7000	7000	0.8
404	1.56	175	175	175	120	120	120	0.8

the steel plates are different. The presence of the cohesive layer does not influence significantly the numerical results.

The comparison of numerical simulation against experimental data of Shot#1201 is depicted in Figs. 4 and 5. The simulation results can be seen to be in good agreement with the experiments except for the normal velocity that behaves differently after the

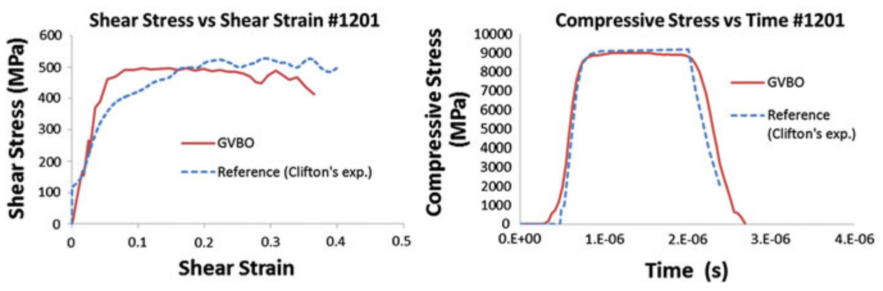


Fig. 4 Shear and compressive stress versus shear strain and time, respectively, for Shot#1201

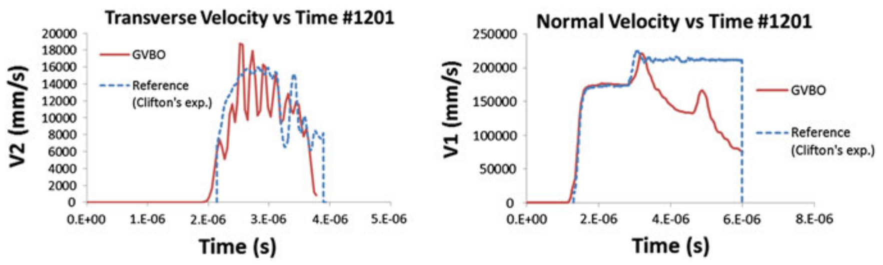


Fig. 5 Transverse and normal velocity versus time for Shot#1201

first peak. This is most likely related to the arrival of the boundary wave and the use of idealized geometry considered in the simulations.

It can be seen that for Shot#404 numerical results are in good agreement with the experimental results (see Figs. 6, 7, 8 (right)).

The compressive stress-strain curve related to pure compression loading-unloading experiments (Shot#1203) is plotted in Fig. 8 (left). The loading compressive stress-strain curve for Shot#404 is depicted in Fig. 8 (right) and compared with the experimental observations taken from reference [11].

The pressure dependence of shearing resistance is approximated by a linear function in Fig. 9 (left), [10]. The numerical simulation results shown in Fig. 9 (right) correspond to the problem geometry and steel material taken from Shot#1201 and Shot#404 (Table 2). The results are obtained for impact velocities ranging between $V_0 = 112.6$ m/s and $V_0 = 183.45$ m/s, and fit nicely the linear function predicted by the experiments.

3.3 Confined Monotonic Loading

There is a lack of experimental data for uniaxial monotonic compressive loading at strain rates in the range of about $10^5\,\mathrm{s}^{-1}$ and in the presence of large distortions. Instead, we consider experimental data from reference [18] related to confined com-

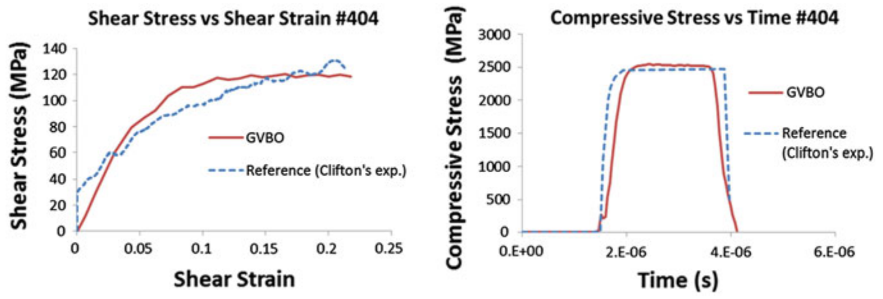


Fig. 6 Shear and compressive stress versus shear strain and time, respectively, for Shot#404

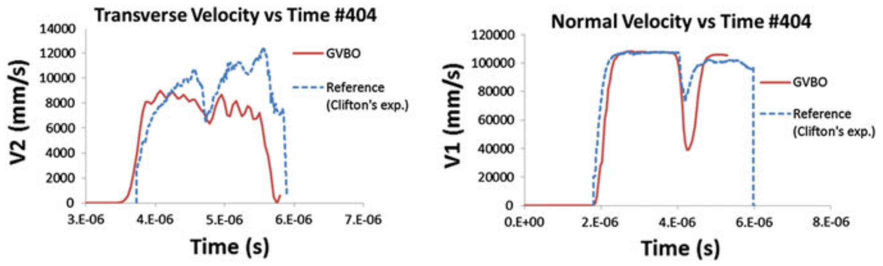


Fig. 7 Transverse and normal velocity versus time for Shot#404

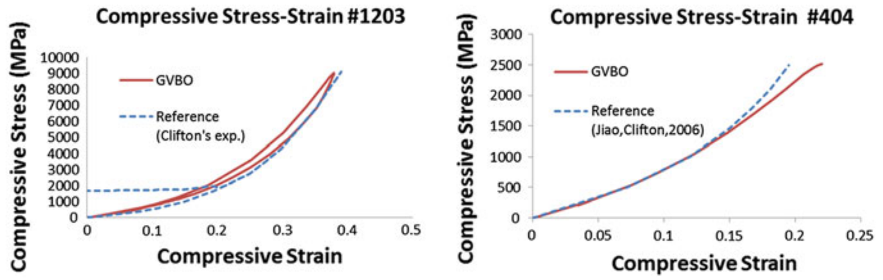


Fig. 8 Compressive stress versus compressive strain for pure-compression Shot#1203 (*left*) and for pressure-shear Shot#404 [11] (*right*)

pressive test of polyurea subjected to a strain rate of $2600\,\mathrm{s}^{-1}$ at 273 K. We model confined compression for a single element of polyurea. Figure 10 shows a reasonable agreement with experimental data. The result has been found to be almost insensitive to inelastic material parameters.

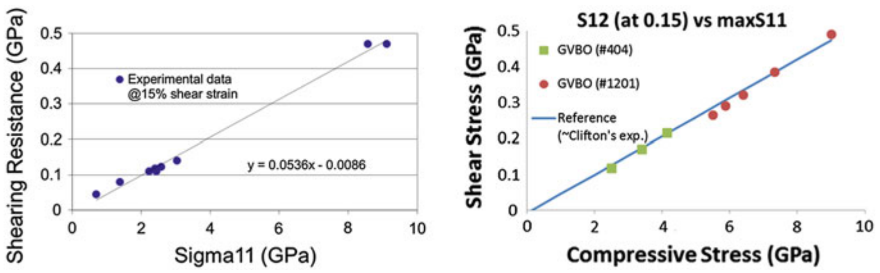


Fig. 9 Dependence of shear resistance on pressure: experimental data [2] (*left*) and numerical simulations (*right*)

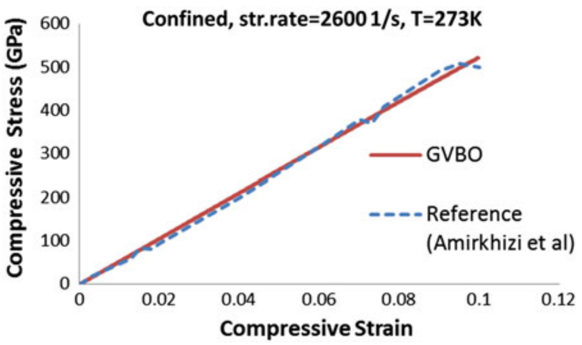


Fig. 10 Confined compressive stress-strain curves for GVBO and experimental data [18]

3.4 Validation for Polyurea/Steel Bi-Layer

In this section we consider a composite polyurea/steel plate impacted by FSP (fragment simulating projectile) at very high velocities (>1000 m/s). Two positions of the polymer layer with respect to the steel have been considered: the front and back coating, with a front side being the target.

The polyurea is modeled by the generalized viscoplasticity model based on overstress (GVBO) outlined in the previous section. For failure criterion, we considered the maximum principal strain. For high-hard steels we studied MIL-A46100 steel which is modeled by the Johnson-Cook constitutive model.

The ballistic limit (or V_{50} velocity) is estimated for a blank steel plate, polyurea-steel plate and steel-polyurea plate. It has been observed that the bi-layer with a front polyurea coating increases penetration velocity by about 15.4 %, while the plate with back coating raises penetration velocity by only 7.5 % for MIL-A46100 steel with respect to the ballistic limit of the blank steel plate.

The polyurea material is modeled by the generalized viscoplasticity model based on overstress. The calibrated material parameters are shown in Table 5. The failure

Table 7 Physical properties of the steel

Steel	Young's Modulus (MPa)	Poisson ratio	Density (g/cm^3)	Inelastic heat fraction, χ	Specific heat (μJ/kgK)	Coefficient of linear thermal expansion (1/K)
MIL-A46100	215000	0.29	7.85	0.9	4.8e+08	11.5e-6

Table 8 The Johnson-Cook parameters for the steel [19, 21]

Steel	A (MPa)	B (MPa)	n	$\dot{\varepsilon}_0$ (s^{-1})	C	m	T_0 (K)	T_m (K)
MIL-A46100	1050	250	0.12	1	0.02	0.5	298	1720

criterion for the polyurea is a maximum principal strain, and the critical damage value is $D^{pol} = 1.4$.

Material properties for the MIL-A46100 steel [19] are listed in Table 7. The parameter χ is the Taylor-Quinney empirical coefficient that represents proportion of plastic work converted into heat, and the value 0.9 is considered in this study following the reference [20].

The steel is modeled by the Johnson-Cook model with material constants taken from references [21, 22]. The von Mises tensile flow stress is defined by

$$\sigma = \left[A + B\varepsilon^n\right]\left[1 + C\ln\dot{\varepsilon}^*\right]\left[1 - T^{*m}\right] \tag{18}$$

where ε is the equivalent (or effective) plastic strain, $\dot{\varepsilon}$ and $\dot{\varepsilon}_0$ are the current and reference strain rates; $\dot{\varepsilon}^* = \dot{\varepsilon}/\dot{\varepsilon}_0$ is the dimensionless plastic strain rate for $\dot{\varepsilon}_0 = 1.0\,\mathrm{s}^{-1}$. The homologous temperature T^* is defined as

$$T^* = \frac{T - T_0}{T_m - T_0} \tag{19}$$

where T_m and T_0 are the melting and room temperatures, respectively. The Johnson-Cook parameters for the steel are summarized in Table 8.

The Johnson-Cook model defines the damage variable as

$$D = \sum \frac{\Delta\varepsilon}{\varepsilon^f} \tag{20}$$

where $\Delta\varepsilon$ is an increment of equivalent plastic strain, and ε^f is the equivalent strain to fracture determined as

$$\varepsilon^f = \left[D_1 + D_2\exp\left(D_3\sigma^*\right)\right]\left[1 + D_4\ln\dot{\varepsilon}^*\right]\left[1 + D_5T^*\right] \tag{21}$$

For the Johnson-Cook model, fracture is assumed to occur when $D = D^{steel} = 1$. The Johnson-Cook failure parameters for the MIL-A46100 steel are not available in

Table 9 The Johnson-Cook failure parameters for the 4340 steel [22]

D_1	D_2	D_3	D_4	D_5
0.05	3.44	−2.12	0.002	0.61

Table 10 Ballistic limit of different plates

Steel	Ballistic limit	Blank steel	Polyurea-steel	Steel-polyurea
High-hard steel, experiments [13, 14]	V_{50} (m/s)	1184 ± 5	1483 ± 7	–
	Difference w.r.t. to blank steel	–	23 %	∼ 8.8 %
MIL-A46100, simulation	V_{50} (m/s)	1188	1371 ± 1	1277
	Difference w.r.t. to blank steel	–	15.4 %	7.5 %

the open literature and have been assumed to be the same as for the 4340 steel, (see Table 9 [22]).

The 3D plate is constructed from two layers: steel and polyurea of thickness 12.7 mm. The plate radius considered in the simulations is 125 mm. We consider the impact by a fragment simulating projectile of 0.5 caliber, 12.7 diameter, and mass of 12.4 g.

The impact problem is simulated using Abaqus/Explicit solver with built-in Johnson-Cook material and failure models and the user-defined material subroutine (describing GVBO) for the polyurea. Only one quarter of the plate was simulated assuming its symmetry. The bullet is modeled as a rigid body. The plate is assumed to be free.

The model was meshed using 8-node hexahedral element with reduced integration. For stabilization in Abaqus/Explicit, linear and quadratic bulk viscosity parameters were chosen as 0.54 and 2.4, respectively.

The ballistic limit (V_{50}) is estimated in the simulation as an average velocity between the minimum impact velocity for penetration and maximum velocity before penetration. It is listed in Table 10 for the blank steel and coated steel plates. These simulation results are comparable with the results obtained in the experiments [13, 14]. The difference in the ballistic limit between the polyurea-steel and the blank steel is about 15.4 %.

Note that the enhancement obtained by the front polyurea layer is slightly smaller than in the numerical simulation reported in [23] (where an impact problem of FSP penetrating polyurea-4340 steel was considered, and the polyurea was modeled using different viscoplasticity model [15]). In reference [23], the ballistic limits of the blank steel plate and the polyurea-steel bi-layer were overestimated.

Figure 11 shows the snapshot of a projectile penetration into the bi-layer of polyurea and MIL-A46100 steel at impact velocity of $V = 1370\,\mathrm{m/s}$. The figure compares the relative position of the projectile in the polyurea layer in front and back

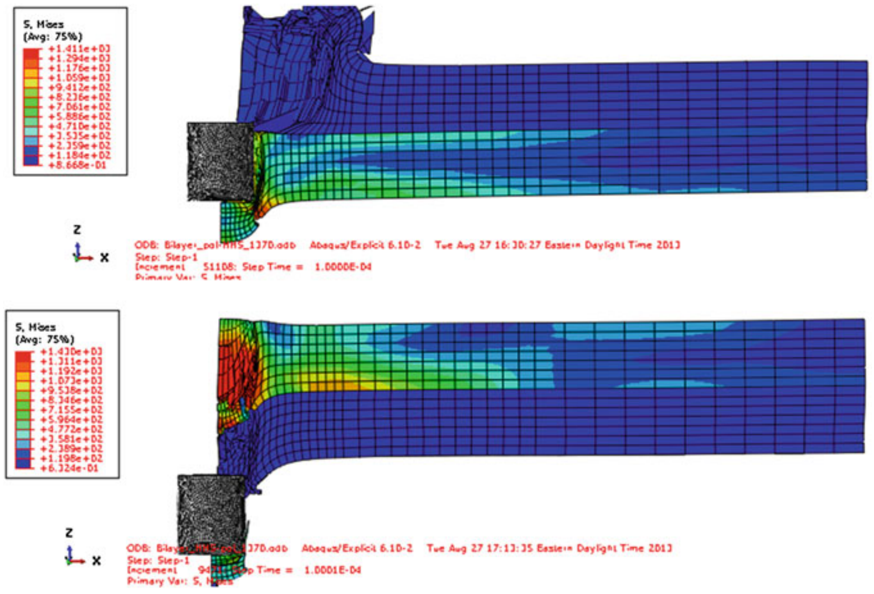


Fig. 11 The FSP impact on polyurea-MIL-A46100 steel plate (*top*) and MIL-A46100 steel-polyurea plate (*bottom*) at impact velocity of 1370 m/s at time 1.e-04s

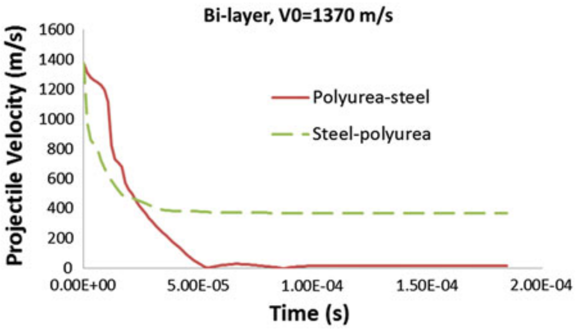


Fig. 12 FSP velocity for polyurea-steel and steel-polyurea plates at impact velocity of 1370 m/s

coating configurations. The results suggest a delay in the projectile penetration in the case of the front polyurea coating polyurea in comparison to the back coating.

The evolution of the projectile velocity in time for both composite plate configurations is shown in Fig. 12. It can be seen that the decrease in the projectile velocity is more pronounced in the case of front polyurea coating of the steel.

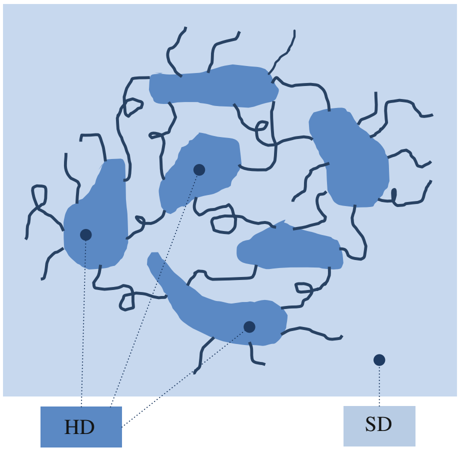


Fig. 13 Micromechanical model of polyurethane consisting of hard domains (HD) and soft domains (SD) with a low HD content [25]

4 Multiscale Modeling of Polymer

4.1 Block Copolymers

Microstructure of the polyurea is composed of hard domains (HD) and soft domain (SD) forming a two-phase microstructure as shown in Fig. 13. Hard segments are with high glass-transition temperature T_g and soft segments are with low T_g. The soft segment has its glass transition below the normal operating temperature and is, therefore, rubbery. The hard segment has its glass transition or its melting temperature above the ordinary operating temperature and is, therefore, either glassy and/or crystalline. It is well known that the microphase separation of hard and soft domains is responsible for the versatile properties of this broad class of polymers [24, 25]. Recent studies [26] have shown that T_g of the soft domain is on average 80 °C higher at the free surface than in the interior and 60 °C higher than at the circumference. For the low strain rate tensile specimens, the T_g increases with strain and reaches a maximum value at a strain of 3.6. These increases in the glass transition temperatures is believed to be due to mixing of the hard and soft segments, but the precise mechanism is not well understood and cannot be investigated without performing micromechanical analysis of the polymer.

Numerous experimental studies (see for example [27]) suggested a significant shift in glass transition T_g with strain rate. The precise effect of strain rate on hard and soft domains is not known, except on the overall behavior of the polymer. We will identify the rate dependent properties of SD and HD domain using inverse method.

4.2 Multiscale Model of Copolymers

Consider the following governing equations of a block copolymer over the composite domain Ω^ζ

$$\begin{aligned} \sigma^\zeta_{ij,j} + b^\zeta_i &= \rho^\zeta \ddot{u}^\zeta_i \quad \text{on} \quad \Omega^\zeta \\ \Delta\sigma^\zeta_{ij} &= L^\zeta_{ijkl}\left(\Delta\varepsilon^\zeta_{kl} - \Delta\mu^\zeta_{kl}\right) \quad \text{on} \quad \Omega^\zeta \\ \Delta\varepsilon^\zeta_{ij} &= \Delta u^\zeta_{(i,j)} \equiv \frac{1}{2}\left(\Delta u^\zeta_{i,j} + \Delta u^\zeta_{j,i}\right) \quad \text{on} \quad \Omega^\zeta \\ u^\zeta_i &= \bar{u}^\zeta_i \quad \text{on} \quad \Omega^{u\zeta} \\ \sigma^\zeta_{ij} n^\zeta_j &= t^\zeta_i \quad \text{on} \quad \Omega^{t\zeta} \end{aligned} \tag{22}$$

where $\mathbf{u}^\zeta$ is displacement, ρ^ζ is density; $\Delta u^\zeta_{i,j}$ denotes the derivative of displacement increment with respect to the midstep; $\Delta u^\zeta_{(i,j)}$ denotes the symmetric gradient; $\Delta\varepsilon^\zeta_{kl} = \Delta u^\zeta_{(i,j)}$ is an integral of the rate of deformation over the time step; $\boldsymbol{\mu}^\zeta$ is an eigenstrain; and σ^ζ_{ij} is Cauchy stress. The microscopic coordinate system is $\mathbf{y} = \mathbf{x}/\zeta$, and it is considered to be independent on the macroscopic system of coordinates, $\mathbf{x}$, when $\zeta \ll 1$. Then the spatial derivative rule is $\partial_{,i} = \partial_{,x_i} + \partial_{,y_i}/\zeta$.

For a viscoplastic material considered in Sect. 2 undergoing large strains, it is convenient to decompose the eigenstrain increment as follows

$$\begin{aligned} \Delta\mu^\zeta_{kl} &= \Delta\mu^{in}_{kl} + \Delta\mu^{el}_{kl} \\ \Delta\mu^{el}_{kl} &= \left(I_{klij} - \left(L^\zeta_{klmn}\left(E^\zeta_0, G^\zeta_0\right)\right)^{-1} L^\zeta_{mnij}\left(E^\zeta, G^\zeta\right)\right)\left(\Delta\varepsilon^\zeta_{ij} - \Delta\mu^{in}_{ij}\right) \end{aligned} \tag{23}$$

where $\Delta\mu^{el}_{kl}$ is contribution to eigenstrain resulting from deformation-dependent elastic properties (16), and $\Delta\mu^{in}_{kl}$ is the usual inelastic strain increment.

The displacement field is expanded asymptotically as

$$u^\zeta_i(\boldsymbol{x}, \boldsymbol{y}, t) = u^c_i(\boldsymbol{x}, t) + \zeta u^{(1)}_i(\boldsymbol{x}, \boldsymbol{y}, t) + O\left(\zeta^2\right) \tag{24}$$

where $\mathbf{u}^c$ and $\mathbf{u}^{(1)}$ are coarse and fine-scale displacements, respectively.

The first order strain field is given by

$$\begin{aligned} \Delta\varepsilon^\zeta_{ij}(\boldsymbol{x}, \boldsymbol{y}, t) &= \Delta\varepsilon^c_{ij}(\boldsymbol{x}, t) + \Delta u^{(1)}_{(i,y_j)}(\boldsymbol{x}, \boldsymbol{y}, t) + O(\zeta) \\ \Delta\varepsilon^c_{ij}(\boldsymbol{x}, t) &= \Delta u^c_{(i,x_j)}(\boldsymbol{x}, t) = \frac{1}{|\Theta|}\int_\Theta \Delta\varepsilon^\zeta_{ij}(\boldsymbol{x}, \boldsymbol{y}, t)\, d\Theta \end{aligned} \tag{25}$$

The fine-scale displacement is assumed to be decomposed as [28–32]

$$\Delta u_i^{(1)}(\boldsymbol{x},\boldsymbol{y},t) = H_i^{kl}(\boldsymbol{y},t)\,\Delta\varepsilon_{kl}^{c}(\boldsymbol{x},t) + \int\limits_{\Theta} g_i^{kl}(\boldsymbol{y},\tilde{\boldsymbol{y}},t)\,\Delta\mu_{kl}^{\zeta}(\boldsymbol{x},\tilde{\boldsymbol{y}},t)\,d\tilde{\Theta} \tag{26}$$

where H_i^{kl}, g_i^{kl} are $\boldsymbol{y}$-periodic instantaneous functions for elastic and inelastic deformation, respectively. Note that for large strain problems, the unit cell domain may evolve and therefore the influence functions may change from one increment to another. In the present manuscript we introduce an approximation by which we approximate the influence functions by their initial values

$$\begin{aligned} H_i^{kl}(\boldsymbol{y}) &\equiv H_i^{kl}(\boldsymbol{y},t) = H_i^{kl}(\boldsymbol{y},0) \\ g_i^{kl}(\boldsymbol{y},\tilde{\boldsymbol{y}}) &\equiv g_i^{kl}(\boldsymbol{y},\tilde{\boldsymbol{y}},t) = g_i^{kl}(\boldsymbol{y},\tilde{\boldsymbol{y}},0) \end{aligned} \tag{27}$$

In the model reduction approach [28–33] the eigenstrain field is discretized over volume partitions as follows

$$\mu_{ij}^{\zeta}(\boldsymbol{x},\boldsymbol{y},t) = \sum_{\alpha=1}^{n} N^{(\alpha)}(\boldsymbol{y})\,\mu_{ij}^{c(\alpha)}(\boldsymbol{x},t) \tag{28}$$

where n is a number of partitions or phases. A shape function $N^{(\alpha)}$ is either continuous (for compatible eigenstrains, see [33]) or piece-wise constant as follows

$$N^{(\alpha)}(\boldsymbol{y}) = \begin{cases} 1 & \boldsymbol{y} \in \Theta^{(\alpha)} \\ 0 & \boldsymbol{y} \notin \Theta^{(\alpha)} \end{cases} \quad \bigcup_{\alpha=1}^{n} \Theta^{(\alpha)} = \Theta; \quad \bigcap_{\alpha=1}^{n} \Theta^{(\alpha)} = \emptyset \tag{29}$$

and

$$\mu_{ij}^{c(\beta)}(\boldsymbol{x},t) = \frac{1}{\left|\Theta^{(\beta)}\right|} \int\limits_{\Theta^{(\beta)}} \mu_{kl}^{\zeta}(\boldsymbol{x},\boldsymbol{y},t)\,d\Theta^{(\beta)} \tag{30}$$

Inserting (27) and (28) into (26) and introducing $S_i^{kl(\alpha)}(y) = \int_{\Theta} g_i^{kl}(\boldsymbol{y},\tilde{\boldsymbol{y}})\,N^{(\alpha)}(\tilde{\boldsymbol{y}})\,d\tilde{\Theta}$ yields

$$\Delta u_i^{(1)}(\boldsymbol{x},\boldsymbol{y},t) = H_i^{kl}(\boldsymbol{y})\,\Delta\varepsilon_{kl}^{c}(\boldsymbol{x},t) + \sum_{\alpha=1}^{n} S_i^{kl(\alpha)}(\boldsymbol{y})\,\Delta\mu_{kl}^{c(\alpha)}(\boldsymbol{x},t) \tag{31}$$

Inserting (31) into (25) gives

$$\Delta\varepsilon_{kl}^{\zeta}(\boldsymbol{x},\boldsymbol{y},t) = \left(I_{klmn} + H_{(k,y_l)}^{mn}(\boldsymbol{y})\right)\Delta\varepsilon_{mn}^{c}(\boldsymbol{x},t) + \sum_{\alpha=1}^{n} S_{(k,y_l)}^{mn(\alpha)}(\boldsymbol{y})\,\Delta\mu_{mn}^{c(\alpha)}(\boldsymbol{x},t) \tag{32}$$

Given the rate of deformation increment and the previous converged value of Cauchy stress in each phase of the unit cell the stress can be updated using the GVBO model outlined in the previous section.

Finally, the coarse-scale Cauchy stress is obtained by averaging fine-scale stresses

$$\sigma_{ij}^{c} = \frac{1}{|\Theta|} \int\limits_{\Theta} \sigma_{ij}^{\zeta} d\Theta \tag{33}$$

for the coarse-scale equilibrium equation

$$\sigma_{ij,j}^{c} + b_i^c = \rho^c \ddot{u}_i^c \tag{34}$$

where $\mathbf{b}^c$ and ρ^c are averages of fine-scale body force $\mathbf{b}^\zeta$ and density ρ^ζ, respectively.

4.3 Dispersion Contribution

In dynamic multiscale problems there is an effect of micro-inertia, which arises due to material heterogeneity. We account for the micro-inertia effect in the coarse-scale problem by modifying the coarse-scale stress (for more details see [17])

$$\bar{\sigma}_{ij}^{c} = \sigma_{ij}^{c} + \zeta^2 D_{ijkl} \ddot{\varepsilon}_{kl}^{c} \tag{35}$$

The coarse-scale problem (34) is then redefined as

$$\begin{aligned} \bar{\sigma}_{ij,j}^{c} + b_i^c &= \rho^c \ddot{u}_i^c \quad \text{on} \quad \Omega \\ \bar{\sigma}_{ij,j}^{c} n_j^c &= \bar{t}_i^c \quad \text{on} \quad \partial\Omega^t \end{aligned} \tag{36}$$

where D_{ijkl} is a dispersion coefficient that depends on the material impedance, unit cell size and overall density [17].

To compute the dispersion coefficient it is convenient to decompose it to linear and nonlinear contributions

$$D_{ijkl} = D_{ijkl}^{\text{lin}} + D_{ijkl}^{\text{nonlin}} \tag{37}$$

The linear term is defined as

$$D_{ijkl}^{\text{lin}} \equiv \rho^c \frac{1}{|\Theta|} \int\limits_{\Theta} h_s^{ij} h_s^{kl} d\Theta \tag{38}$$

where h_i^{kl} is $\boldsymbol{y}$-periodic solution of the following problem

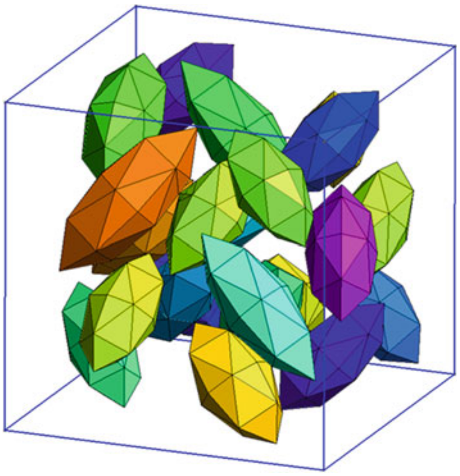


Fig. 14 Microstructure of the unit cell with particles preferentially oriented at a 45° with respect to a vertical axis

$$\begin{aligned} & h^{kl}_{(i,y_j)}(y) = H^{kl}_{(i,y_j)}(y) + \frac{\Delta\rho(y)}{\rho^c} I_{ijkl} \\ & \int_\Theta h_s^{ij}\, d\Theta = 0;\ \Delta\rho(y) = \rho^c - \rho(y) \end{aligned} \tag{39}$$

The nonlinear term is defined as

$$D^{\text{nonlin}}_{ijkl} \equiv \sum_{\alpha=1}^{n}\sum_{\beta=1}^{n} \frac{\partial\Delta\mu^{(\alpha)}_{st}}{\partial\Delta\varepsilon^c_{ij}} R^{(\alpha\beta)}_{stpq} \frac{\partial\Delta\mu^{(\beta)}_{pq}}{\partial\Delta\varepsilon^c_{kl}} + \sum_{\alpha=1}^{n} Q^{(\alpha)}_{ijpq}\frac{\partial\Delta\mu^{(\alpha)}_{pq}}{\partial\Delta\varepsilon^c_{kl}} + \sum_{\alpha=1}^{n} Q^{(\alpha)}_{klpq}\frac{\partial\Delta\mu^{(\alpha)}_{pq}}{\partial\Delta\varepsilon^c_{ij}}$$

$$R^{(\alpha\beta)}_{stpq} = \rho^c \frac{1}{|\Theta|}\int_\Theta S_r^{st(\alpha)} S_r^{pq(\beta)} d\Theta;\ Q^{(\alpha)}_{ijpq} = \rho^c\frac{1}{|\Theta|}\int_\Theta h_r^{ij} S_r^{pq(\alpha)} d\Theta \tag{40}$$

Note that the dispersion effect becomes significant when (i) the unit cell size is large, (ii) material impedance is considerable (i.e. large difference in elastic moduli or densities between phases) and (iii) spatial gradient of acceleration is high.

5 Verification of the Multiscale Model

For modeling polyurea microstructure, we consider an anisotropic unit cell with ellipsoidal particles (volume fraction 19 %) shown on Fig. 14. The particles are oriented at preferential angle of 45° with respect to the loading direction.

The soft domains are modeled by the GVBO model while hard domains are kept elastic with considerably higher elastic modulus than the initial modulus in the soft domain. Material parameters of the phases (Table 11) are calibrated to the experimental data of normal dynamic impact test [10] and confined test under monotonic

Table 11 GVBO parameters for polyurea in two-scale analysis

Soft domains	E_0 (MPa)	ν_0	E_t (MPa)	p	a_1 (MPa)	a_2 (MPa)	a_3 (1/MPa)
	156	0.494	23.3	−6	0	10	0
k_1	k_2 (MPa)	k_3 (MPa)	k_4	E_m (MPa)	A_0 (MPa)	A_c (s^{-1})	A_f (MPa)
17	0.47	1.47	4	13600	1	0	0
Hard domains	E_0 (MPa)	ν_0					
	890	0.491					

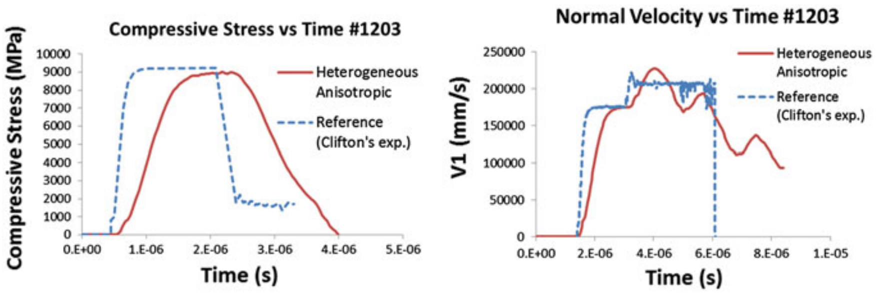


Fig. 15 Compressive stress and normal velocity versus time. Comparison of multiscale simulation with calibrated parameters and experimental data, Shot#1203 [10]

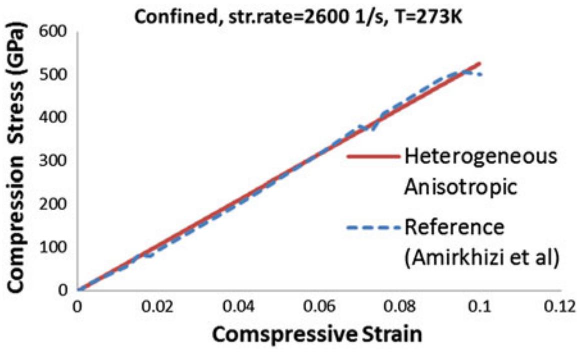


Fig. 16 Confined compressive stress-strain curves. Comparison of multiscale simulation with calibrated parameters and experimental data [18]

compressive loading [18]. The results are shown in Figs. 15 and 16. The two-scale model of polyurea is implemented in Abaqus /Explicit with MDS plugin http://multiscale.biz.

The multiscale model of polyurea is studied for impact problem on polyurea/steel bi-layer and single polyurea layer. In these studies we have not considered material failure and thus relatively low impact velocity have been analyzed.

The evolution of projectile velocity is shown in Figs. 17 and 19 for bi-layer and single layer, respectively. The response of the heterogeneous anisotropic material

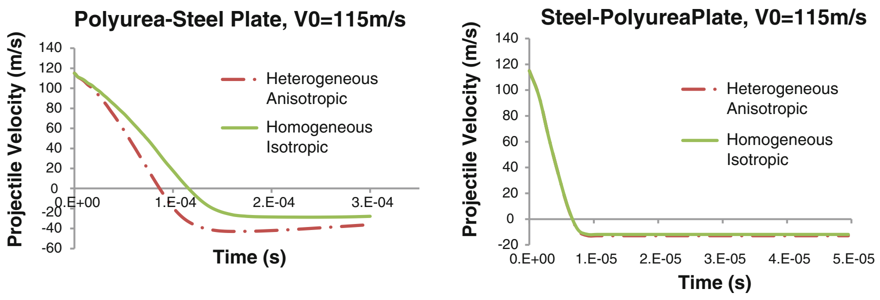


Fig. 17 Evolution of the FSP velocity impacted to bi-layers at initial velocity of 115 m/s. Comparison of the multiscale simulation with anisotropic heterogeneous material and a single-scale model with isotropic homogeneous material properties

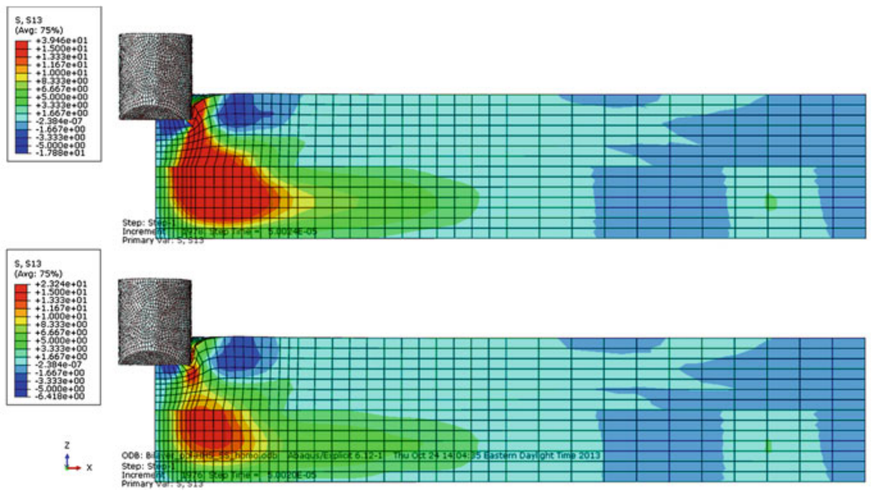


Fig. 18 Shear stress in the polyurea-steel bi-layer at initial velocity of 115 m/s at time 5.e-05 s. Comparison of multiscale simulation with anisotropic heterogeneous material (*top*) and a single-scale model with isotropic homogeneous material properties (*bottom*)

(Table 11) is compared with the homogeneous single scale GVBO model (Table 5). It can be seen that a heterogeneous anisotropic polymer layer alone or placed on the top of the bi-layer (Fig. 17, left) delays the projectile considerably better than a homogenous polymer. In other words, the energy has been dissipated much faster and much more effectively by heterogeneous anisotropic material. This is due to the shear wave propagation as opposed pressure wave propagation in the case of an isotropic polymer. When the polymer is placed on the bottom of the steel there is no difference between the homogeneous and heterogeneous polymer (Fig. 17, right).

The snapshots showing the shear wave stress distribution in the polyurea layer placed on the top of the bi-layer and in the polyurea plate alone are depicted in

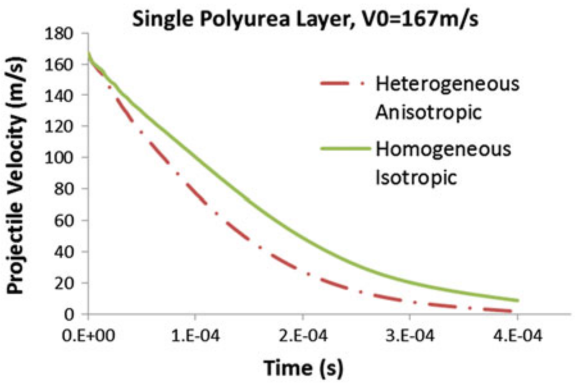


Fig. 19 Evolution of the FSP velocity impacted on the single polyurea layer with initial velocity of 167 m/s. Comparison of the multiscale simulation with anisotropic heterogeneous material and a single-scale model with isotropic homogeneous material properties

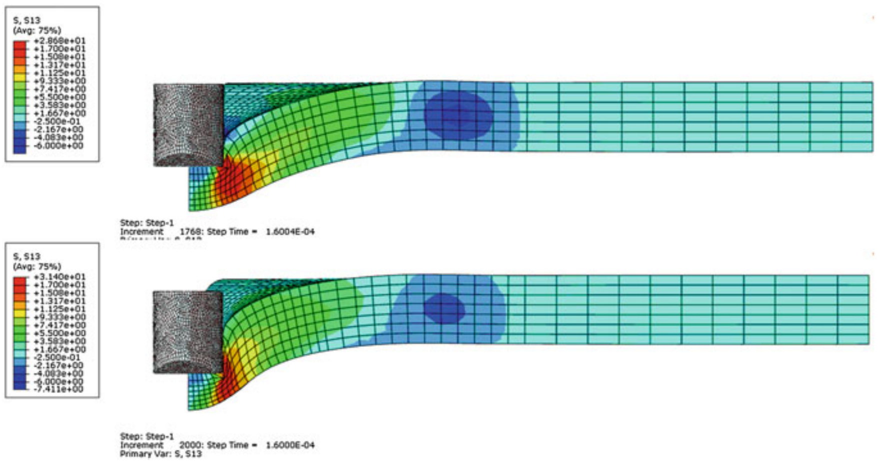

S, S13
(Avg: 75%)
Step: Step-1
Increment 1768: Step Time = 1.6004E-04
Increment 2000: Step Time = 1.6000E-04
Primary Var: S, S13

Fig. 20 Shear stress in the polyurea layer model at impact velocity of 167 m/s at time 1.6e-04 s. Comparison of multiscale simulation with anisotropic heterogeneous material (*top*) and a single-scale model with isotropic homogeneous material properties (*bottom*)

Figs. 18 and 20, respectively. It can be seen that due to preferential orientation of the hard domains more energy is dissipated than in a homogeneous polyurea.

To study the dispersion effect we consider an impact onto the polymer plate with initial velocity of 300 m/s. For demonstration purposes we consider relatively large size of the unit cell (about 1 mm) and high ratio of phase densities (hard domain is ten times denser than a soft domain). A failure of polymer material is not included here. The micro-inertia effect is implemented in Abaqus/Explicit solver by adding the dispersion term into overall stress (35). The simulation results depicted in Figs. 21

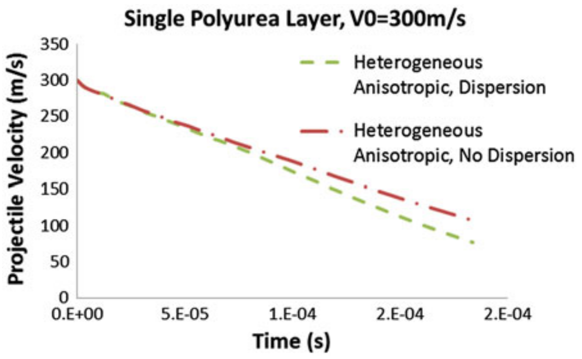


Fig. 21 Evolution of the FSP velocity impacted on the single polyurea layer at initial velocity 300 m/s. Comparison of the multiscale simulation for anisotropic heterogeneous material with and without dispersion

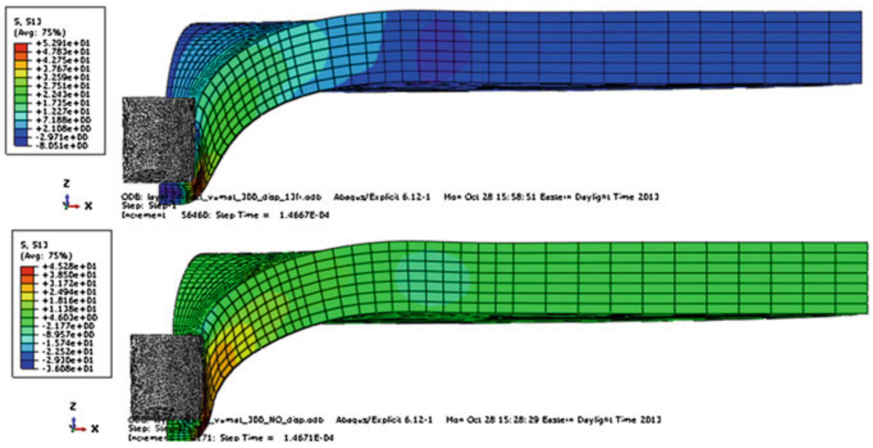


Fig. 22 Shear stress distribution in a polyurea layer impacted by a projectile at initial velocity of 300 m/s at time 1.47e-04 s. Comparison of the multiscale simulation with anisotropic heterogeneous material and with (*top*) and without (*bottom*) consideration of dispersion

and 22 demonstrate that dispersion indeed contributes to the decrease in the projectile velocity.

6 Summary

In this study we presented a new constitutive model of polyurea, the generalized viscoplasticity based on overstress (GVBO) for finite deformations at very high strain rates (10^5 s^{-1}). The model parameters were identified against experimental results and a good predictability of polyurea shear resistance at various pressure levels has been observed.

The GVBO polyurea model was used to simulate the penetration of FSP into polyurea/steel bi-layers for high hard steel at high impact velocities. The simulation results qualitatively confirmed experimental results suggesting the front position of the polyurea layer to be superior to the back polyurea layer. However, the projectile velocities observed in the experiments were higher than those predicted in the simulations.

Finally, we demonstrated that energy can be dissipated by means of polymer anisotropy and polymer heterogeneity. We showed that it is possible to engineer such a polymer microstructure that will dissipate energy by means of shear waves as opposed to pressure waves that cannot dissipate energy. Furthermore, we developed a dispersive multiscale model that accounts for energy absorption by means of dispersion within material microstructure.

Acknowledgments The support of ONR grant N00014-12-1-0558 is gratefully acknowledged.

References

1. Roland CM, Casalini R (2007) Effect of hydrostatic pressure on the viscoelastic response of polyurea. Polymer 48:5747–5752
2. Green MS, Tobolsky AV (1946) A new approach to the theory of relaxing polymeric media. J Chem Phys 14(2):80–92
3. Johnson AR, Quigley J, Freese CE (1995) A viscohyperelastic finite element model for rubber. Comput Methods Appl Mech Eng 127:163–180
4. Roland CM (1989) Network recovery from uniaxial extension: i. elastic equilibrium. Rubb Chem Technol 62:863–879
5. Bergstrom JS, Boyce MC (1998) Constitutive modeling of the large strain time-dependent behavior of elastomers. J Mech Phys Solids 46(5):931–954
6. Jiao T, Clifton RJ, Grunschel SE (2009) Pressure-sensitivity and constitutive modeling of an elastomer at high strain rates. AIP Conf Proc 1195:1229–1232
7. Colak OU (2004) Modeling of large simple shear using a viscoplastic overstress model and classical plasticity model with different objective stress rates. Acta Mechanica 167:171–187
8. Gomaa S, Sham T-L, Krempl E (2004) Finite element formulation for finite deformation, isotropic viscoplasticity theory based on overstress (fvbo). Int J Solids Struct 41:3607–3624
9. Grujicic M, He T, Pandurangan B et al (2012) Experimental characterization and material-model development for microphase-segregated polyurea: an overview. J Mater Eng Perform 21:2–16
10. Clifton RJ, Jiao T (2012) Resistance of elastomers to shearing and failure at extreme loading conditions. ONR-Workshop
11. Jiao T, Clifton RJ, Grunschel SE (2006) High strain rate response of an elastomer. AIP Conf Proc 845:809–812
12. Jiao T, Clifton RJ (2013) Measurement of the response of an elastomer at pressures up to 9gpa and strain rates of 10^5–$10^6 s^{-1}$. In: 18th Biennial international conference of the APS topical group on shock compression of condensed matter, Washington, July 2013
13. Roland CM, Fragiadakis D, Gamache RM et al (2012) Factors influencing the ballistic impact resistance of elastomer-coated metal substrates. Philos Mag 93(5):468–477

14. Roland CM, Fragiadakis D, Gamache RM (2010) Elastomer-steel laminate armor. Compos Struct 92:1059–1064
15. Roland CM, Twigg JN, Vu Y et al (2007) High strain rate mechanical behavior of polyurea. Polymer 48:574–578
16. Amini MR, Isaacs J, Nemat-Nasser S (2010) Investigation of effect of polyurea on response of steel plates to impulsive loads in direct pressure-pulse experiments. Mech Mater 42:628–639
17. Fish J, Filonova V, Kuznetsov S (2012) Micro inertia effects in nonlinear heterogeneous media. Int J Numer Meth Eng 91(13):1406–1426
18. Amirkhizi AV, Isaacs J, McGee J et al (2006) An experimentally-based viscoelastic constitutive model for polyurea, including pressure and temperature effects. Philos Mag 86(36):5847–5866
19. Grujicic M, Ramaswami S, Snipes JS et al (2013) Multiphysics modeling and simulations of Mil A46100 armor-grade martensitic steel gas metal arc welding process. J Mater Eng Perform 22: 2950–5969
20. Borvik T, Hopperstad OS, Dey S et al (2005) Strength and ductility of weldox 460 e steel at high strain rates, elevated temperatures and various stress triaxialities. Eng Fract Mech 72(7):1071–1087
21. Johnson GR, Cook WH (1983) A constitutive model and data for metals subjected to large strains, high strain rartes and high temperatures. In: Proceedings of 7th international symposium on ballistics, The Hague, 1983
22. Johnson GR, Cook WH (1985) Fracture characteristics of 3 metals subjected to various strains, strain rates, temperatures and pressures. Eng Fract Mech 21(1):31–48
23. Irshidat M, Al-Ostaz A, Cheng AH-D (2011) Predicting the response of polyurea coated high hard steel plates to ballistic impact by fragment simulating projectiles. Ole Miss Project 90031. http://www.serri.org/publications/Documents. Accessed 12 May 2011
24. Yi J, Boyce MC, Lee GF et al (2006) Large deformation rate-dependent stress-strain behavior of polyurea and polyurethanes. Polymer 47:319–329
25. Qi HJ, Boyce MC (2005) Stress-strain behavior of thermoplastic polyurethanes. Mech Mater 37:817–839
26. Lee G, Mock W, Fedderly J et al (2007) The effect of mechanical deformation on the glass transition temperature of polyurea. AIP Conf Proc 955:711–714
27. Sharma A, Shukla A, Prosser RA (2002) Mechanical characterization of soft materials using high speed photography and split hopkinson pressure bar technique. J Mater Sci 37(5):1005–1017
28. Oskay C, Fish J (2008) Fatigue life prediction using 2-scale temporal asymptotic homogenization. Comput Mech 42(2):181–195
29. Oskay C, Fish J (2007) Eigendeformation-based reduced order homogenization. Comp Meth Appl Mech Eng 196:1216–1243
30. Yuan Z, Fish J (2009) Multiple scale eigendeformation-based reduced order homogenization. Comput Methods Appl Mech Eng 198(21–26):2016–2038
31. Yuan Z, Fish J (2009) Hierarchical model reduction at multiple scales. Int J Numer Meth Eng 79:314–339
32. Fish J, Yuan Z (2008) N-scale model reduction theory. In Fish J (ed) Multiscale methods: bridging the scales in science and engineering. Oxford University Press, New York
33. Fish J, Filonova V, Yaun Z (2013) Hybrid impotent-incompatible eigenstrain based homogenization. Int J Numer Meth Eng 95(1):1–32

Numerical Simulation of Double Cup Extrusion Test Using the Arbitrary Lagrangian Eulerian Formalism

Romain Boman, Roxane Koeune and Jean-Philippe Ponthot

Abstract In this chapter Double Cup Extrusion Test (DCET) is modelled using the finite element method with the help of the Arbitrary Lagrangian Eulerian (ALE) formalism. DCET is a tribological test involving very large deformations which are traditionally dealt with complicated and costly remeshing algorithms. Since the topology of ALE meshes should remain constant throughout the simulation, two very thin layers of auxiliary elements are added to the initial mesh of the billet where the material is expected to flow. This numerical trick is combined with an original and efficient node relocation procedure which allows the model to take into account complex geometries of punches. The presented model is firstly validated for limited punch strokes thanks to a purely Lagrangian simulation. It is then compared with results from the literature. Eventually the general nature and the effectiveness of this numerical strategy is demonstrated by a fully-coupled thermomechanical simulation of thixoforming where the final shape of the billet is compared to experimental measurements.

1 Introduction

The Double Cup Extrusion Test (DCET) is a tribological test dedicated to forging operations. Before the conception of DCET, one of the easiest way to quantify friction for this type of processes was the *ring compression test* (see Male and

R. Boman (✉) · R. Koeune · J.-P. Ponthot
Department of Aerospace and Mechanical Engineering, University of Liege, 1 chemin des Chevreuils, 4000 Liège, Belgium
e-mail: r.boman@ulg.ac.be

R. Koeune
e-mail: r.koeune@gmail.com

J. Ponthot
e-mail: jp.ponthot@ulg.ac.be

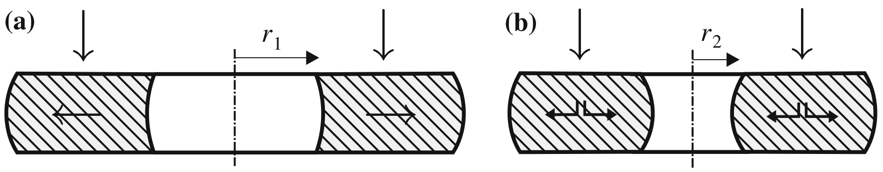


Fig. 1 Schematic description of the ring compression test which may be used to quantify friction in forging operations [25] **a** low friction (good lubrication) **b** high friction (bad lubrication)

Cockcroft [17]), which consists in crushing a flat ring until a prescribed thickness is obtained, as depicted on Fig. 1. If the contacts are well lubricated (Fig. 1a), the material flows outwards and the inner radius of the ring increases. If the friction becomes higher (Fig. 1b), this radial motion is slowed down. A smaller radius is then obtained ($r_2 < r_1$). Consequently, the final inner radius can be used as an indirect measure of friction. However, this simple tribological test reproduces rather badly the real contact conditions and the very high deformations that can be observed at the interfaces between the material and the tools of real forging operations. Indeed, according to Bay [2], it is common to reach pressures close to 2.5 GPa, surface temperatures higher than 600 °C and local surface elongation up to 3000 %. DCET was conceived by Geiger [12] in order to measure friction and to test lubricants in tribological conditions that are closer to these values.

Although DCET is much more elaborated than the ring compression test, the measured friction can still be deduced from very simple geometrical quantities. The experimental setup can be described as follows (see Fig. 2a): a cylindrical billet is placed in a hollow container of same diameter between two punches. During the test, the lower punch is not moving while the upper punch goes down and crunches the specimen. Therefore the material is forced to flow along both punches in such a way that two *cups* are gradually formed. If the contacts were perfectly lubricated, i.e. in the frictionless case, the material would flow symmetrically upwards and downwards. The H-shaped section of the forged billet would have two branches with the same height ($h_1 = h_2$).

In practise, friction is unavoidable and induces a dissymmetry in the process. The obtained final section looks like the one represented in Fig. 2b. The height of the upper cup (h_1) is always higher than the lower one (h_2). The friction can then be quantified by the *cup height ratio* h_1/h_2. The higher this ratio is, the higher the friction was during the test.

A first direct application to this tribological test is the classification of lubricants according to their respective efficiency in forging conditions. For example, Gariety et al. Gariety et al. [11] have compared four lubricants thanks to DCET. They have also studied the possibility of jamming by visualising the grooves on the free surfaces of the billet after the test.

A second interesting application is the numerical estimation of a friction coefficient with the help of the finite element method. In the case of forging, the Tresca's law is usually chosen to model friction:

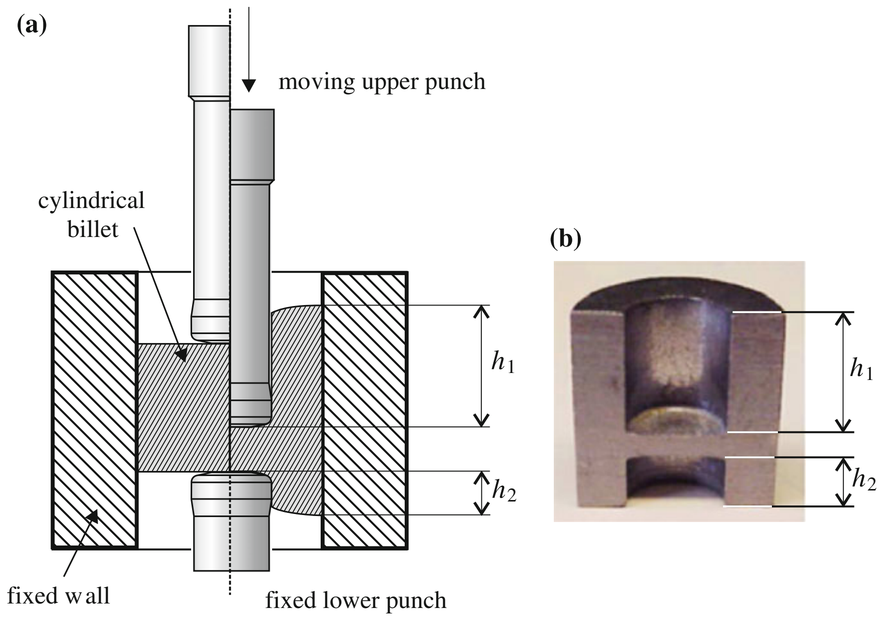


Fig. 2 **a** Principle of the double cup extrusion test from [23]—**b** Picture of a deformed billet after DCET (from Gariety et al. [11])

$$\tau \leq m\, \tau_{\max} \tag{1}$$

where τ is the friction shear stress at the contact interface, m is the friction coefficient and $\tau_{\max}$ is the shear yield stress of the material. A series of numerical simulations of DCET can be performed using a range of friction values m and the corresponding curves of cup height ratios h_1/h_2 versus upper punch displacements can be plotted. This set of calibration curves and the experimental measurement are then used to identify a mean coefficient m for the process [7, 9, 26]. This friction value might be used later, with much care, in more complex numerical simulations of forging which would involve the same material and the same lubricant. The finite element models of the previously cited authors were all using the commercial code DEFORM-2D [24] which conveniently provides an automatic remeshing procedure for quadrangular meshes.

It is important to notice that the relevance of DCET to evaluating friction in forging may be somewhat questionable. In fact, the material flow is mostly influenced by the friction between the billet and the wall of the container. The friction between the punch and the material, which is more representative of a forging operation, plays a less significant role on the dissymmetry of the final shape of the billet. Moreover, some authors, such as Schrader et al. [23], think that the pressures exerted by the billet on the container are not high enough to use a Tresca's law in the finite element models. A Coulomb's law should be more appropriate. Nevertheless, despite these issues, modelling this tribological test is still very interesting from a numerical point of view.

This chapter is organised in the following way: after a brief review of the ALE formalism, a simplified extrusion model is presented in order to explain the numerical trick that will be used to keep the topology of the ALE mesh constant. Then, this technique is extended to the case of extrusion with curved punches. Next, the model is validated for small punch strokes by comparison with a classical Lagrangian model and a simplistic ALE model using mesh smoothing. For larger punch strokes, the model is compared to results from the literature obtained with a complete remeshing strategy. Finally, a fully-coupled thermomechanical problem of semi-solid forming is described and the final predicted shape of the billet is compared to experimental observations.

This work has been done with Metafor Ponthot [22], an in-house implicit finite element code developed at the University of Liège in Belgium.

2 Overview of the ALE Formalism

In the ALE formalism, unlike in the Lagrangian case which is commonly used in Solid Mechanics, the mesh no longer follows the material motion. Consequently, a new grid coordinate system R_χ is defined and the conservation laws and the constitutive equations are rewritten in terms of the new coordinates χ [3, 4, 8, 22]:
Mass:

$$\left.\frac{\partial \rho}{\partial t}\right|_\chi + \boldsymbol{c}\cdot\nabla\rho + \rho\nabla\cdot\boldsymbol{v} = 0 \tag{2}$$

Momentum:

$$\rho\left(\left.\frac{\partial \boldsymbol{v}}{\partial t}\right|_\chi + (\boldsymbol{c}\cdot\nabla)\,\boldsymbol{v}\right) = \nabla\cdot\boldsymbol{\sigma} + \rho\boldsymbol{b} \tag{3}$$

Energy:

$$\rho\left(\left.\frac{\partial u}{\partial t}\right|_\chi + \boldsymbol{c}\cdot\nabla u\right) = \boldsymbol{\sigma} : \mathbf{D} + \rho r + \nabla\cdot\boldsymbol{q} \tag{4}$$

Material:

$$\left.\frac{\partial \boldsymbol{\sigma}}{\partial t}\right|_\chi + (\boldsymbol{c}\cdot\nabla)\,\boldsymbol{\sigma} = \mathscr{H} : \mathbf{D} + \mathbf{W}\boldsymbol{\sigma} - \boldsymbol{\sigma}\mathbf{W} \tag{5}$$

where ρ is the mass density, $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{b}$ and r are the specific body forces and heat sources, u is the specific internal energy, $\mathbf{D}$ and $\mathbf{W}$ are the symmetric and antisymmetric part of the velocity gradient tensor, $\boldsymbol{q}$ is the heat flux, and $\mathscr{H}$ is a material tensor depending on the constitutive parameters, the stresses, and the loading history. The last two terms in Eq. (5) result from the particular choice of the Jaumann's objective time derivative.

The convective velocity $\boldsymbol{c} = \boldsymbol{v} - \boldsymbol{v}^*$ is the difference between the material velocity $\boldsymbol{v}$ and the mesh velocity $\boldsymbol{v}^*$. In the case of nonlinear problems, such as metal forming simulations, $\boldsymbol{v}^*$ should ideally depend on the solution. It is thus an additional unknown of the latter system of equations.

In order to simplify the solution procedure and remain competitive against Lagrangian models, the set of ALE equations is usually solved using an *operator-split* procedure. Each time increment, from time t to $t + \Delta t$, is divided into two successive steps. The first one is performed exactly in the same way as in the classical Lagrangian case. During this *Lagrangian step* the mesh follows the material motion ($\boldsymbol{v}^* = \boldsymbol{v}$, $\boldsymbol{c} = 0$) until an equilibrated configuration is obtained. The second step, also called the *Eulerian step*, is divided into two substeps: the definition of an appropriate mesh velocity $\boldsymbol{v}^*$ by relocating each node of the mesh to a more suitable position Boman and Ponthot [5], followed by the data transfer from the old mesh configuration to the new one Boman and Ponthot [6]. This transfer involves the Gauss-point values (stress tensor components, history variables of the material such as the equivalent plastic strain) as well as nodal values (velocities, accelerations and temperature).

In the case of simulations of tribological tests such as DCET, the computation of friction forces is obviously very important. Nevertheless this evaluation is not as easy as in the Lagrangian case for which the position of each node of the mesh corresponds to the same material particle during the whole simulation. The following strategy is thus implemented: during the Lagrangian step, a classical penalty method is used to compute the friction occurring at the nodes in contact with a tool. Then, after the Eulerian step, the equilibrated internal forces are recomputed from the transferred stress field on the new mesh. The friction forces are calculated by projection onto the tools and the tangential gaps are recovered from these updated forces.

3 Basic ALE Model of Extrusion

The extrusion process, or more precisely wiredrawing, was first investigated using the ALE formalism by Huétink et al. [15] in 1990. In that early work, the studied problem was 2D axisymmetric, the mesh was purely Eulerian and the stationary solution was sought. Later, Van Haaren et al. [27] and Geijselaers and Huétink [13] built an extrusion model in order to analyse their respective novel ALE convection schemes. Similarly the mesh was fixed in space and a transient computation was performed until the stationary state was reached. In those chapters, the analysis of the results is not exhaustive: the plastic strain is solely visualised in order to compare the numerical diffusion of the newly developed advection schemes. In particular, the friction modelling is not discussed at all.

Transient models of extrusion have been also proposed by Atzema and Huétink [1] and by Ponthot [21]. The ALE formalism can be very useful in this context. In these

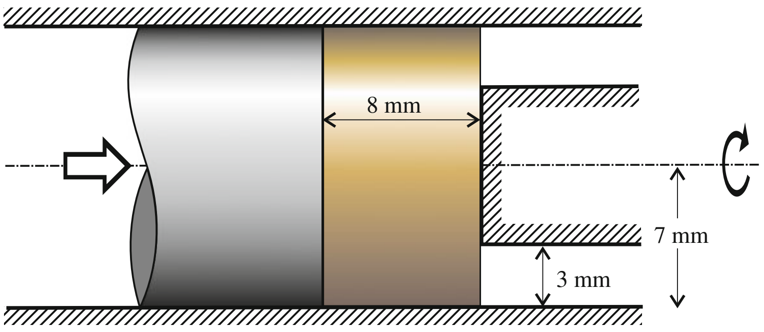


Fig. 3 Axisymmetric geometry of the extrusion model of [21]. A cylindrical specimen is constrained to flow into a *narrow* channel in order to produce a *hollow* cylinder. The specimen shapes at the beginning and the end of the extrusion are completely different

kind of models, the mesh is not Eulerian anymore. Since the ALE formalism requires a constant mesh topology and thus a constant number of finite elements throughout the simulation, it is important to take advantage of the approximate knowledge of the final shape of the extruded billet in order to build the initial mesh. As an illustration of the particular mesh management procedure, Ponthot's model is presented in Fig. 3. The extrusion problem is axisymmetric. A cylindrical billet is pushed by a punch into a narrower channel so that a hollow cylinder is formed. The material is elastoplastic ($E = 200\,\text{GPa}$, $\nu = 0.3$, $\sigma_Y = 210 + 10\,\bar{\varepsilon}^p$ MPa) and the friction on the boundaries is modelled by a Coulomb's law with a friction coefficient $\mu = 0.15$.

The transient solution is made up of two quadrangular regions (see Fig. 5): the first region corresponds to the part of the crushed cylindrical billet that still remains between the punches and the second one contains the material which has been already extruded and which lies between the fixed punch and the container wall. At time $t = 0$, the second region should be ideally empty. In order to keep a unique mesh from the beginning to the end of the simulation, Ponthot initially assigns a very small thickness ($h = 0.01$ mm) to this second region and creates a mesh on it. This artificial region is called *auxiliary region* in the remainder of this work. The resulting finite elements of the auxiliary region are thus very flat, but they can *inflate* as a result of the material flux coming from the first region. The node relocation strategy is relatively simple (see Fig. 4). Most of the vertices of the mesh are Lagrangian (i.e. they follow the material motion). Only two vertices are Eulerian (i.e. fixed in space). The line defining the nose of the fixed punch and its neighbour separating both regions of the mesh are also Eulerian. The nodes of the other lines are relocated by defining a cubic spline through them. These splines are then remeshed so that the initial node distribution and their respective curvilinear abscissa are preserved. As far as the inner nodes are concerned, they are continuously relocated thanks to the same transfinite mesher that was used to generate the meshes. These node-relocation methods are fully described in a previous chapter Boman and Ponthot [5].

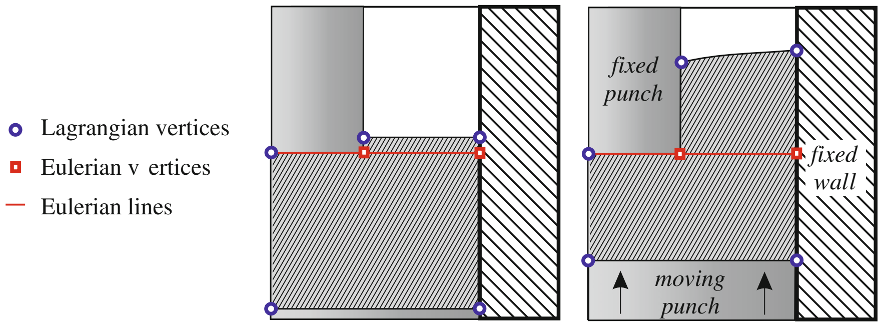


Fig. 4 Node relocation procedure. Thanks to the simple geometry of the fixed punch, the definition of the new mesh is made very easy. The nodes and the *line* in *red* are Eulerian. The other *lines* which have at least one *red* vertex are remeshed using cubic *splines*

Figure 5 shows the progress of the simulation. Of course, the proposed mesh management technique entails some issues. First of all, it is mandatory to roughly know the direction of the material flow when setting up the model. Moreover, seeing that the number of finite elements is initially fixed in the auxiliary regions of the mesh, these elements become larger and larger as the simulation progresses and, consequently, the geometry of the extruded parts becomes crudely discretised. Finally, it is not possible to extrude all of the material. The mesh of the billet must always be made up of the same number of elements, but its thickness continuously decreases. The crushed quadrangles, which lie either in the auxiliary region at the beginning of the simulation or in the main region of the mesh at the end of the simulation, lead to some convergence difficulties. On the one hand these finite elements are poorly conditioned for the Lagrangian steps of the ALE algorithm and, on the other hand, the stability criterion of the explicit data-transfer scheme of the Eulerian step is very restrictive concerning the maximum allowable punch displacement during a single time step. As a result, a very small time step has to be used at the beginning and at the end of the simulation (Fig. 6). Figure 7 shows the calculated force on the moving punch during the extrusion operation. The curve obtained by the current implementation is compared to the former results of Ponthot [21]. The trends of both curves are very similar and the final values are identical. The discrepancy between the curves may be explained by the differences in the ALE management of friction.

Despite these limitations, this particular mesh management technique is very attractive for modelling extrusion or any other process for which the material flow is predictable. For example, Gadala et al. [10] used the same ALE method to compute the shape of a metallic chip of a cutting operation.

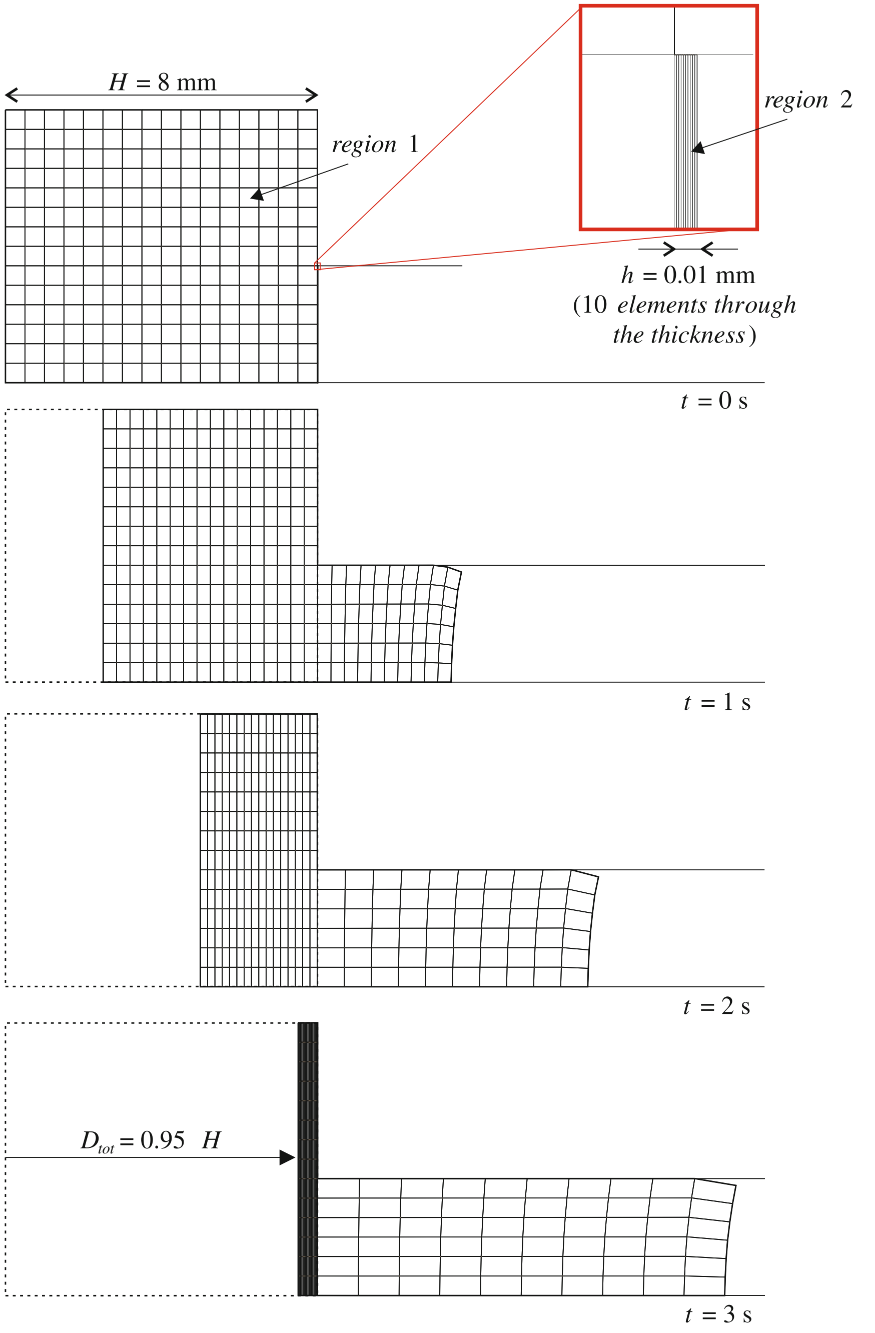


Fig. 5 Results of the extrusion test of Ponthot [21] for a punch stroke up to 95 % of the initial thickness of the cylinder (H)

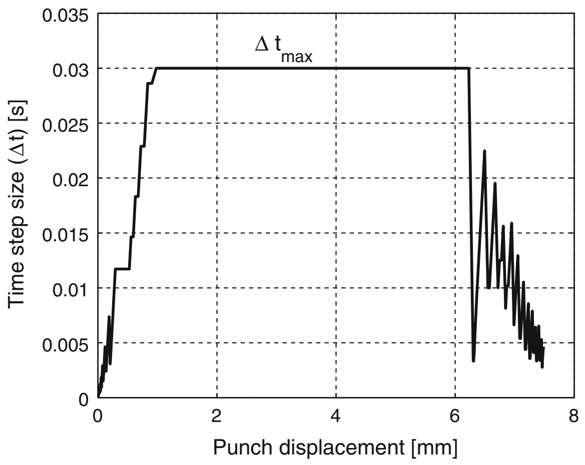


Fig. 6 Size of the time increment Δt along the simulation

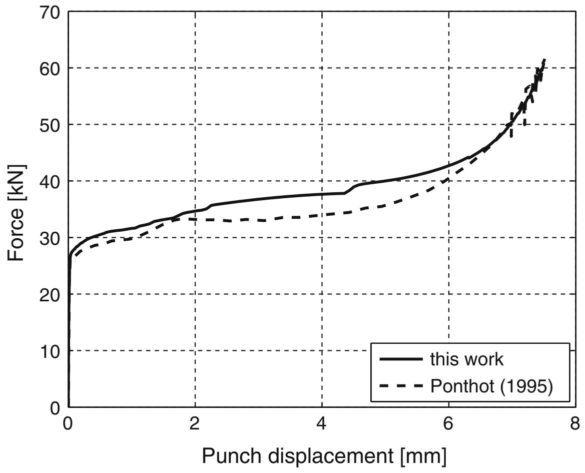


Fig. 7 Extrusion force as a function of the punch displacement and comparison with the results of Ponthot [21]

4 ALE Model of DCET

4.1 Geometry and Parameters

The previous mesh management technique is now applied to a Double Cup Extrusion Test. The chosen geometry was developed at the Engineering Research Center (ERC) of the Ohio State University in order to assess the properties of various lubricants [7]. The exact punch geometry is described in Fig. 8 and the corresponding numerical

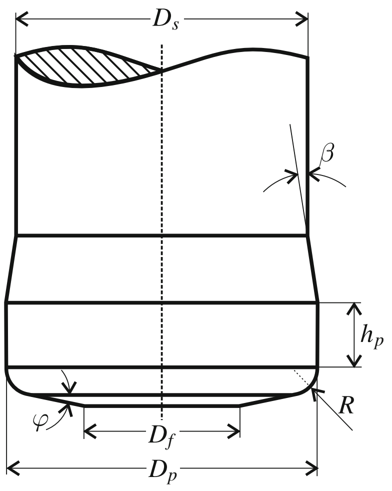


Fig. 8 Punch geometry used by Tan et al. [26] and Schrader [23]

Table 1 Geometry of the extrusion test studied by [23]. The parameters are related to Fig. 8

Punch							Billet	
D_p [mm]	D_f [mm]	D_s [mm]	R [mm]	h_p [mm]	φ [^{o}]	β [^{o}]	d_0 [mm]	h_0 [mm]
15.88	9.53	15.72	1.17	1.57	10.0	5.0	31.75	31.75

values are listed in Table 1. They are related to the work of [23], which will be the main reference in the remaining part of this chapter. The geometry of the punch is far more complex than the one used by Ponthot. Even if Buschhausen et al. [7] claim that the shape of the punch does not play a significant role on the results (its shape is actually optimised to favour the radial flow of the lubricant, which is not modelled here), this complex shape and all its geometrical details are retained in the model in order to demonstrate the capabilities of our ALE node-relocation algorithm.

The initial diameter of the cylindrical billet d_0 is equal to its height h_0 and to the internal diameter of the container. The *extrusion ratio* is defined as the ratio of the surface of the punch nose and the upper surface of the billet ($r = D_p^2/d_0^2$). This value, deduced from Table 1, is equal to $r = 0.25$. According to Schrader et al. [23], this particular value of r is ideal to observe large variations in the results due to friction conditions.

The material is an AISI 1018 steel with classical elastic properties: Young's modulus $E = 200\,\text{GPa}$ and Poisson's ratio $\nu = 0.3$. The nonlinear hardening is modelled by the following law:

$$\sigma_Y = K\,(\bar{\varepsilon}^p)^n \tag{6}$$

where $K = 735\,\text{MPa}$ and $n = 0.17$. This law has been identified from a standard tensile test the elastic part of which has been neglected. It is employed here in this form, despite the fact that the initial yield stress is zero.

A Tresca's law models the frictional contact: Eq. (1) may be rewritten as $\tau \leq m\,\sigma_Y/\sqrt{3}$. The extrusion test is supposed to provide the value of this friction coefficient m by identification of an experimental curve and a series of numerical curves obtained with a range of m values. In the following simulations the default value is $m = 0.05$. In practise, the yield stress σ_Y appearing in the Tresca's law is usually chosen as the initial yield stress (see for example the DCET simulations of Tan et al. [26]). In the work of Schrader et al., this numerical value is zero. Consequently, a first possibility is to use the updated local yield stress. However, this value is only defined at the Gauss points of the neighbouring elements of the contact nodes, and not directly at these nodes. The yield stress should thus be extrapolated from the Gauss points to the contact nodes. As a consequence, the friction force evaluated at a given node depends on all the positions of the nodes of the neighbouring elements and a new and more complete stiffness matrix must be computed in order to keep a quadratic convergence rate. The second possibility is to choose a mean value of the yield stress of the material. The cold-drawn AISI 1018 steel is listed on `matweb` [18] with an initial yield stress of 370 MPa. When this particular value is chosen, numerical results close to the ones of Schrader et al. [23] are obtained. The first choice leads to sensibly different results, which show that some uncertainties remain in the numerical parameters used in the reference work.

The model is axisymmetric and integrated in time by a implicit quasi-static solver (the speed of the punch is about 10 mm/s, which is largely insufficient to produce some inertia phenomena). The mesh is made up of Selective Reduced Integration (SRI) quadrangles. Both punches and the container are assumed rigid. The contact is modelled by the penalty method. The normal and tangent penalty coefficients, p_N and p_T respectively, are determined by trial and error: $p_N = 6\,10^4$ MPa/mm and $p_T = 6\,10^3$ MPa/mm for the container wall, and $p_N = 2\,10^4$ MPa/mm and $p_T = 2\,10^3$ MPa/mm for the punches. The billet is regularly meshed with elements of 1 mm along the extrusion direction and 0.3 mm along the radial direction for a total of 31×52 elements). As in Ponthot's model presented in the Sect. 4.1, the material flows are anticipated by adding two very thin auxiliary meshes close to both punches (15 elements though the thickness $\varepsilon = 0.2$ mm—see Fig. 10a).

The data-transfer step of the ALE algorithm consists in updating the stresses (pressure p, and deviatoric stress components s_{rr}, s_{rz}, s_{zz}) and the equivalent plastic strain $\bar{\varepsilon}^p$. These five fields are processed by a first-order Godunov scheme [6].

4.2 ALE Mesh Motion

The mesh motion definition is obviously more complex than in the case of the former example. The main difficulty is to define the motion of the red line highlighted in Fig. 9 which represents the surface of the billet under the punch nose and its extension up to the container wall. Unlike its counterpart in Fig. 4, this curve may not be Eulerian because the punch is not stationary anymore. Furthermore, given its slightly convex geometry, the punch is not entirely in contact with the surface of the billet at the

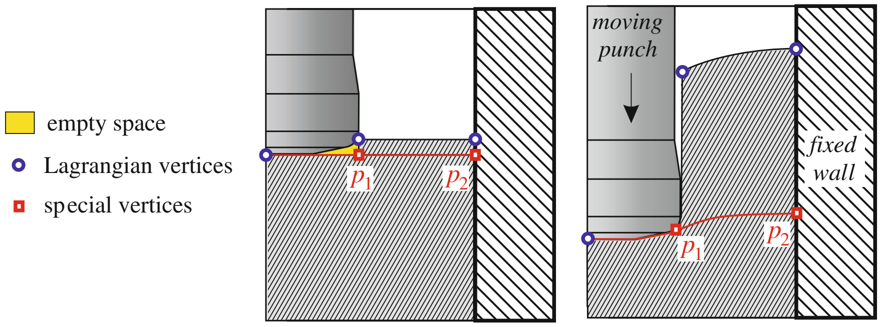
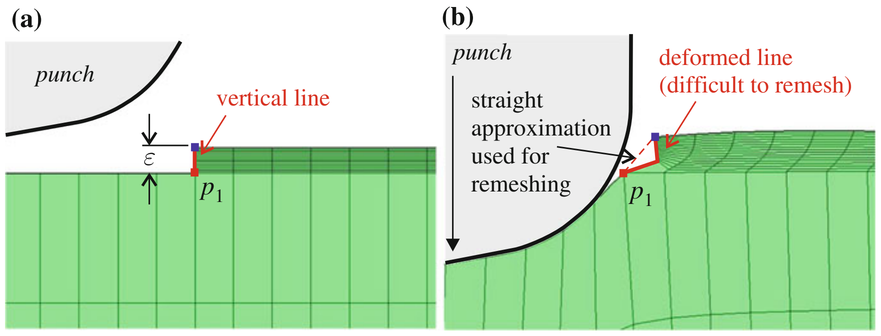


Fig. 9 Node relocation of the DCET model. Unlike the case of Fig. 4, the *red line* may not be Eulerian anymore



(a)
punch
vertical line
ε
p_1
(b)
punch
straight approximation used for remeshing
deformed line (difficult to remesh)
p_1

Fig. 10 **a** Zoom on the upper finely-meshed auxiliary region at the beginning of the simulation—**b** Excessive distorsion of the elements of the auxiliary domain if the nodes of the *red line* follow the real motion of the material boundary

beginning of the computation. A small gap, which is initially empty, should be filled during the first moments of the process.

One could imagine to simply prevent the radial motion of node p_1 of Fig. 9. The position of node p_2 would be such that both nodes would have continuously the same Y-coordinate. The other vertices would be Lagrangian. Unfortunately, this solution does not work because an unavoidable material flux is observed between the two regions of the mesh and the thin auxiliary mesh becomes rapidly distorted during the first steps of the simulation. Figures 10a and b explain this issue and the proposed solution. Initially, the vertical line above p_1 is very short and very finely meshed. During the first steps, this line is deformed because the first element of the auxiliary region receives some spurious fluxes related to the relocation of p_1 on the piecewise-linear boundary of the mesh. These fluxes are very small but, compared to the very small area of the elements, they are large enough to highly deteriorate the auxiliary mesh and to make the line impossible to remesh. Consequently this line is remeshed as if it was a straight line until the contact of p_1 with the punch is established.

The simulation is performed in two successive steps. The first one aims at filling up the empty gap between the billet and the punch. At the end of this step, the punch nose is entirely in contact with the billet and the situation becomes similar to the simple extrusion problem of the previous section. During this first step, the radial displacement of node p_1 (Fig. 9) is set to zero and node p_2 is Lagrangian. The vertical line above p_1 is remeshed as if it was straight in order to prevent any distortion problem of the boundary. Doing so, a small inward spurious material flux is tolerated through this boundary.

The second step begins when node p_1 hits the punch surface. At this precise moment, the vertical displacement of node p_2, as well as those of all the nodes of line (p_1, p_2), are equaled to the one of p_1. This line follows thus the vertical motion of the punch. Concerning the former problematic line, it is remeshed at that stage using a cubic spline in order to precisely follow the boundary of the extruded material.

This two-step strategy is symmetrically applied to the lower part of the billet. If some friction is modelled between the billet and the tools, the process is not symmetrical and the transition from the first step to the second does not occur at the same time for the upper and the lower part of the model. This is not a problem in practise.

Finally and more classically, all the remaining curves defining the boundaries of the billet are continuously remeshed using cubic splines. The internal nodes of the main meshed region are relocated thanks to Giuliani's smoothing method Giuliani [14]. This method has been chosen among many others because it produces the most regular mesh in this very case. This iterative smoothing requires five iterations with a overrelaxation coefficient $\omega = 1.5$. Eventually, both auxiliary meshes are continuously remeshed by transfinite mapping.

4.3 Model Validation for Limited Punch Strokes

The ALE model of the previous section is compared with two other models: the first one is a classical Lagrangian model and the second one is an alternative ALE model which simply consists in smoothing the mesh of the billet without defining any auxiliary region. This comparison enables us to validate the proposed ALE node relocation technique.

The punch displacement s (also called *punch stroke*) is limited to 8 mm so that the three models can converge and produce results. Figure 11a shows the Lagrangian solution. The mesh is highly distorted close to the punch nose. Having a closer look at this problematic spot (Fig. 11b), it can be noticed that the mesh boundary highly penetrates the punch. The node-to-surface formulation of the contact with the rigid tool yields erroneous results: the surface of the billet is subjected to very large local elongations and the surface mesh stretches so widely that the curvature of the punch radius is not well described anymore. Since the contact detection only involves the nodes of the boundary, the edges are free to cross the punch analytical surface producing a very large geometrical error.

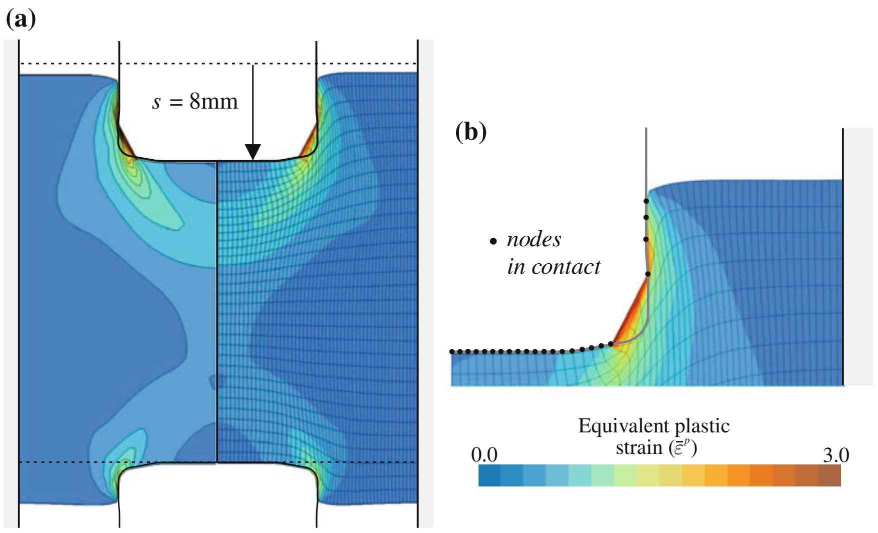


Fig. 11 **a** Lagrangian solution for a 8-mm stroke. These results validate the ALE model for the beginning of the process—**b** Zoom on the Lagrangian solution for which the contact is very badly taken into account

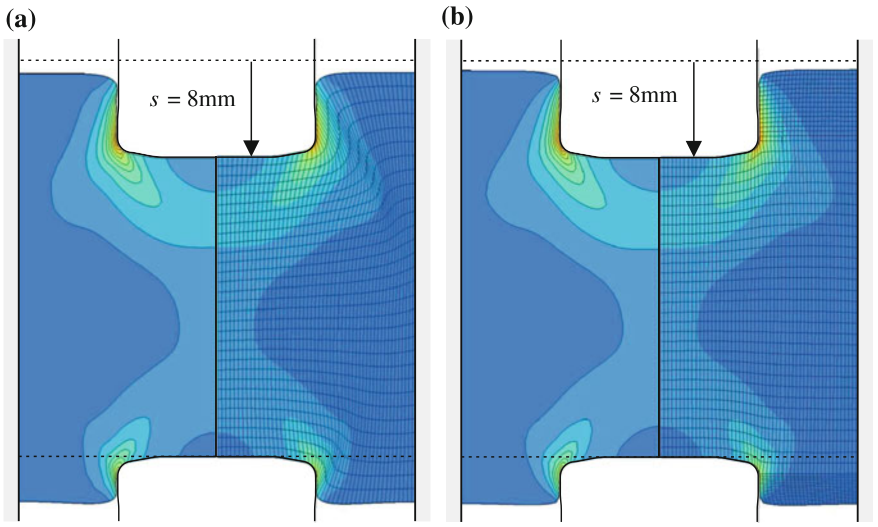


Fig. 12 **a** ALE solution for a 8-mm stroke (simple model). A single region is meshed and a tricky smoothing operator is used—**b** ALE solution for a 8-mm stroke (two-region model)

Figure 12a presents the results obtained by the simple ALE model of DCET without adding any auxiliary meshed regions. The initial mesh is identical to the Lagrangian one. All the boundary nodes are relocated using cubic splines in order

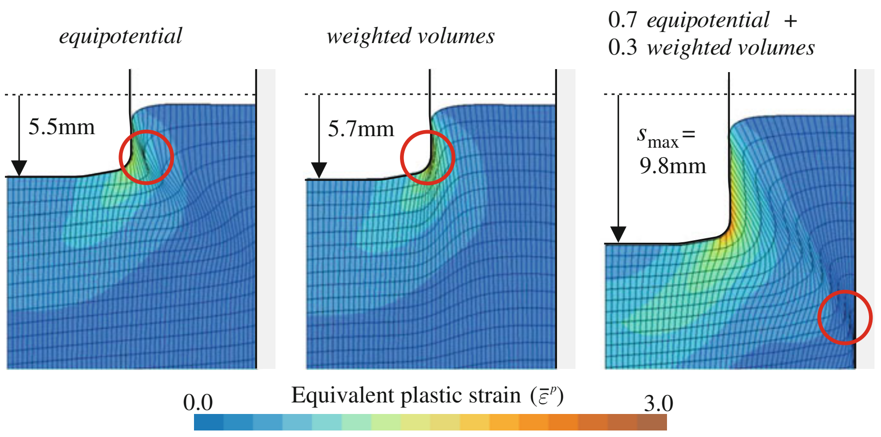


Fig. 13 Comparison of the efficiency of the relocation methods. For each case, the *red circle* indicates the most critical zone of the mesh where the elements are highly distorted

to avoid the excessive stretching of the element edges on the boundary, which was previously discussed. The inner nodes are relocated, after many trials and errors, by a very peculiar combination of two smoothing operators: 70 % of equipotential smoothing and 30 % of weighted volume smoothing Boman and Ponthot [5]. The equipotential part helps to keep the mesh lines almost perpendicular to each other. The weighted-volume part tries to equalise the volumes of the neighbouring quadrangles. Used alone, each of these methods does not permit the computation to converge so far. Figure 13 shows that it is possible to simulate a stoke of $s_{max} = 5.5$ mm with an equipotential smoother and a stoke of $s_{max} = 5.7$ mm with a volume-weighted smoother. An appropriate combination of both methods enables the simulation to converge up to 9.8 mm. Nevertheless, these values are much smaller than the experimental stroke value of 27 mm. Even if this stroke could be reached, it is important to notice that the combination factors of the smoothing methods are case-dependant (which means that they are related to a particular value of the friction coefficient m) and very tricky to guess. This simple ALE model is thus useless except for the validation of the two-region ALE model.

Figure 12b shows the results when using the more sophisticated ALE model including the finely meshed auxiliary region for a punch displacement of 8 mm. This time, the quality of the mesh is very good. The equivalent plastic strain distribution is very similar to the one computed by the simple ALE model. Of course, the extruded heights are slightly different because the two-region model starts at $t = t_0$ with nonzero heights ($h_1(t_0) = h_2(t_0) = \varepsilon$). In order to discard this error, the extruded heights are measured without taking into account the initial small heights of the auxiliary meshes (see Fig. 14).

It is possible to compare more precisely these three models. Figure 15 shows the evolution of h_1 and h_2 as a function of the punch stroke. Both ALE models give very close numerical results that follow the trend of the Lagrangian model at the beginning

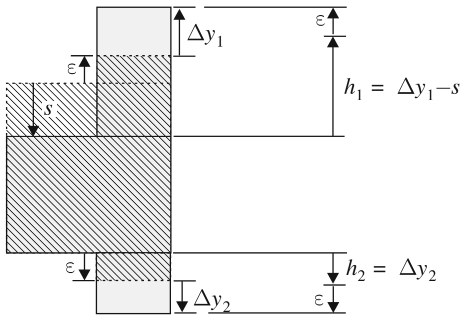


Fig. 14 Computation of the extruded cup heights h_1 and h_2 in the case of the ALE model using the additional auxiliary meshes. The initial heights ε are subtracted

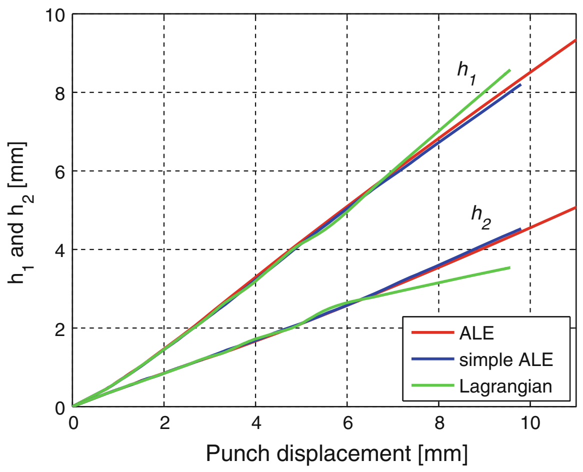


Fig. 15 Comparison of the cup heights h_1 and h_2 for the three models

of the computation. Beyond $s = 5$ mm, the Lagrangian model withdraws from the ALE models because of the penetration of the mesh inside the punch surface (see Fig. 11b). Despite the correction on the computed cup heights, the sophisticated ALE model generates slightly different results from the simple ALE model. This small error ($\Delta h_1 = 0.14$ mm and $\Delta h_2 = -0.08$ mm for $s = 9.8$ mm) certainly comes from the contact length of the billet on the container wall that is not the same in the two cases. The 2-region model is thus subjected to slightly more friction than the simple model. This fact directly affects the corresponding curve of Fig. 16. However, the global trend is quite satisfactory.

In Fig. 17, the curves representing the vertical forces measured on the tools for the three models are very similar at the beginning of the simulation. Starting from $s = 5$ mm, the Lagrangian solution does not model the real process anymore because of the excessive material penetration into the punch. Yet the simple ALE model provides force levels that are very close to the more sophisticated ALE model until it ceases to converge. Finally, as it was expected, the forces of the 3-region ALE model are slightly higher than the ones obtained by the two other models (+1.3 %

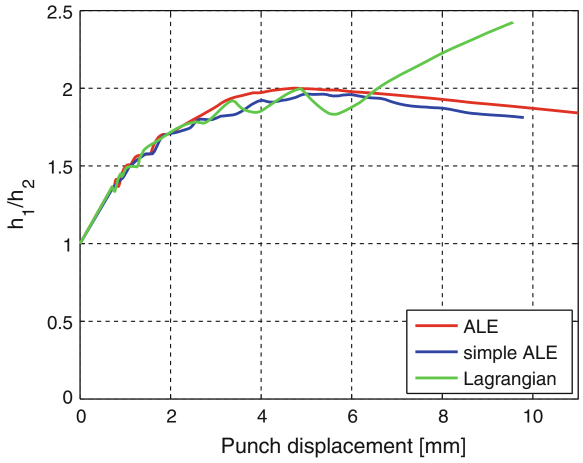


Fig. 16 Comparison of the cup height ratios h_1/h_2 during the process

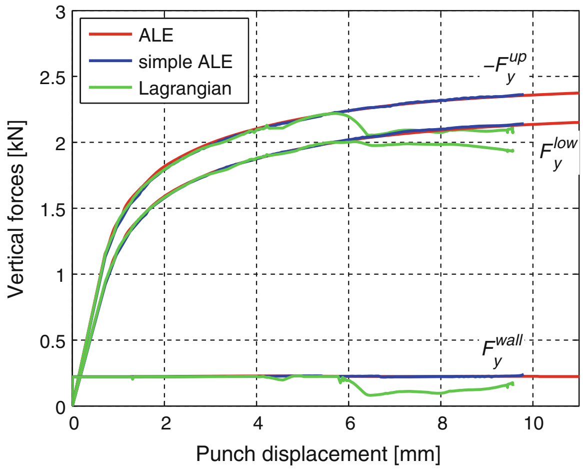


Fig. 17 Vertical forces computed on the *upper* punch F_y^{up}, on the *lower* punch F_y^{low} and on the container wall F_y^{wall} for the three models

for the force on the container wall). This difference, which is barely visible in the Figure, could be further reduced by decreasing the value of ε, at the cost of a slower convergence rate and thus an increase of the total computational time.

This preliminary study proves that the implemented ALE treatment of friction is correct because the same results are obtained independently of the chosen formalism.

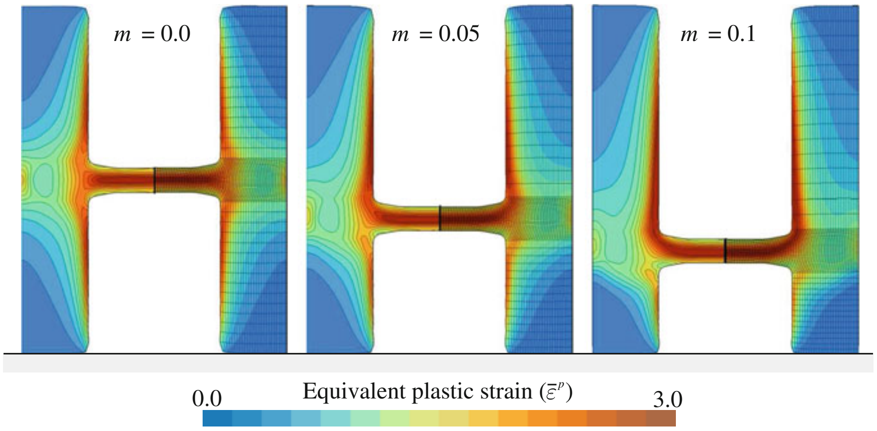


Fig. 18 Deformed billets which have been obtained for several friction values m

4.4 Study of the Whole Process

The whole process is now simulated by using the ALE model until the vertical displacement of the punch reaches $s = 29$ mm, which corresponds to 91 % of the initial height of the billet. When using $m = 0.05$, the problem is solved in 346 time increments corresponding to 422 Newton iterations. The total CPU time is $5'45''$ (single-threaded run on an AMD Opteron 254, 8 Ghz). This time does not depend much on the value of the friction coefficient m. Approximately half of this time (52 %) is spent in the Lagrangian step of the ALE algorithm. The remaining time splits into node relocation routines (7 %) and the data-transfer scheme (41 %).

The deformed shapes of the billet are presented in Fig. 18 for three values of the friction coefficient m ($m = 0, m = 0.05$ et $m = 0.1$). As expected, the upper cup height h_1 becomes larger when the friction coefficient increases. Since the elastic deformations are negligible and thanks to mass conservation, the opposite trend is observed for the lower cup height h_2. One must keep in mind that this volume conservation is not automatically verified in ALE formalism. A closer look at the volume variation of the mesh reveals that the total volume slightly increases during the ALE computations. Table 2 shows several interesting values: the added volume corresponding to the auxiliary meshes represents about 1 % of the exact initial volume of the experimental billet. At the end of the simulation, an increase of about 0.5 % of the total volume of the mesh is noticed instead of a slight loss of volume that could be intuitively expected from the elastic response of the crushed material. This variation mainly results from the spurious material fluxes that are generated during the remeshing of the boundaries of the mesh. A smaller fraction of this error might also come from the limited accuracy of the data-transfer scheme. Anyways, the observed volume variation of the ALE mesh is always positive and it slightly increases as the friction coefficient m increases.

Table 2 Variation of the volume of the mesh in the ALE simulations of DCET after a punch stroke of 29 mm ($m = 0.05$)

	V [mm^3]	V/V_0 $\times 100$ [%]
Initial volume of the experimental billet V_0	25137	100.00
Added volume (auxiliary meshes)	238	0.95
Initial volume of the model	25375	100.95
Final volume of the model ($m = 0.00$)	25492	101.41
Final volume of the model ($m = 0.05$)	25500	101.44
Final volume of the model ($m = 0.10$)	25525	101.54

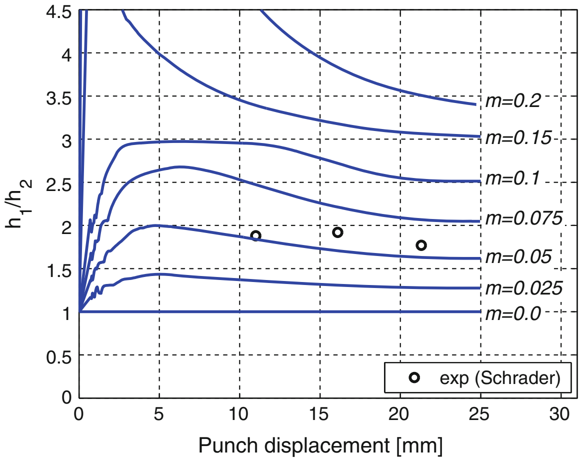


Fig. 19 Set of curves of cup height ratios numerically obtained for a range of friction coefficients m. A value for m may be deduced from the experimental measurements

According to its creator, Geiger [12], the main purpose of DCET is to numerically identify a friction coefficient which is directly related to the chosen lubricant. To reach this goal, a series of simulations are performed by considering a range of friction values m. As an example, Fig. 19 shows the resulting curves in the case of the studied model. The experimental values from Schrader et al. [23]—three punctual measurements—have been superimposed on the numerical curves. A friction value, which is marginally greater than $m = 0.05$, can then be deduced visually from these results.

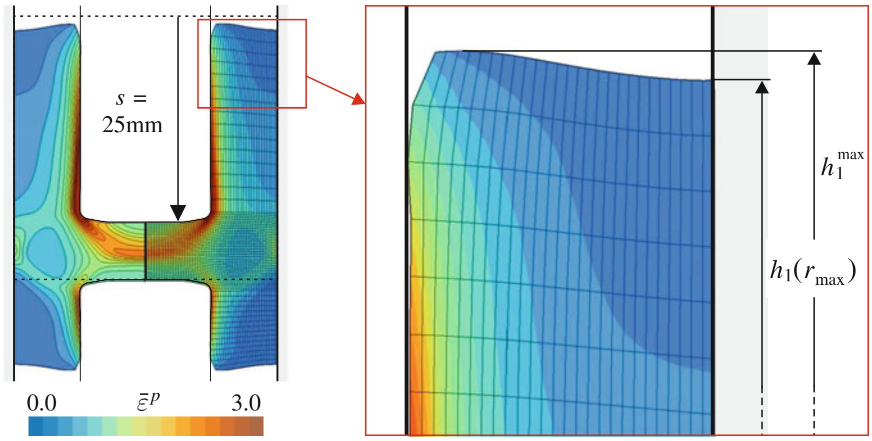


Fig. 20 Two different ways for measuring the height h_1 in the ALE model: either the largest height $h_1^{\max}$ (the measurement position r may vary during the simulation), or the height h_1 measured on the container wall (always at $r_{\max}$). These simulations correspond to $m = 0.05$ and $n = 0$

4.5 Comparison of Results Obtained by ALE and Remeshing

This section is devoted to a comparison between the ALE model and the numerical and experimental work of Schrader et al. [23]. These authors use DEFORM-2D, a FE code which is dedicated to the simulation of forging and extrusion processes. DEFORM-2D features a sophisticated automatic remeshing algorithm which is very useful to avoid critical distortions of the quadrangular finite elements during the computation. The numerical techniques, which are compared in this section, are thus completely different.

In their work, Schrader et al. study the influence of the hardening coefficient n of the material (see Eq. 6) on the cup height ratio and on the contact pressure when the friction value is $m = 0.05$. The cup height ratio is plotted in Fig. 21 for $n = 0.17$ (the reference value) and $n = 0.0$ (perfectly plastic material). As far as the ALE results are concerned, not one but two curves have been plotted for each hardening coefficient n. The first curve is obtained when the value of h is measured on the container wall (at $r = r_{\max} = d_0/2$). The second one is related to the largest value of h which could be measured at a variable radial position r during the simulation. As an example, Fig. 20 shows the final shape of the billet for a hardening coefficient of $n = 0$. The position of the largest value of h_1 is not located at the container wall. The cup height ratio may vary a lot according to the particular shape of the upper free surface of the material and to the measurement position of h_1. This is particularly the case when the punch stroke and h are small. For each value of n in Fig. 21, the two ALE curves nicely surround the one obtained by Schrader using DEFORM-2D. For a larger stroke, these three curves converge to an identical final value of the cup height ratio.

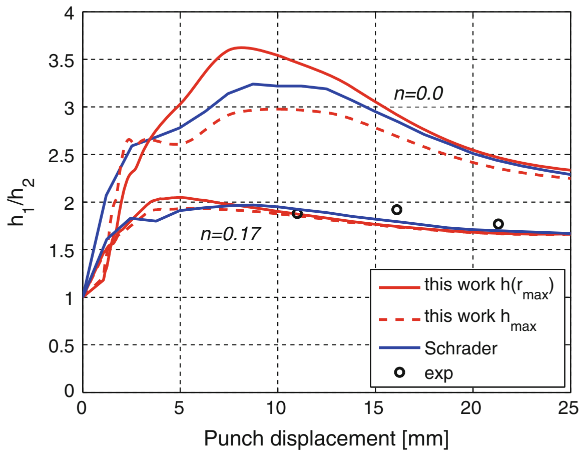


Fig. 21 Influence of the hardening coefficient n of the material on the cup height ratio ($m = 0.05$)

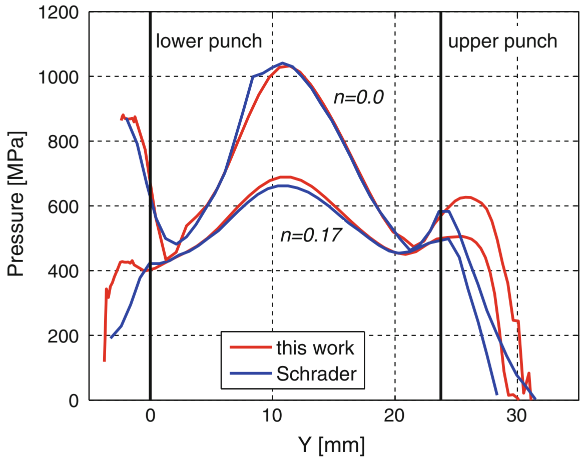


Fig. 22 Pressure field measured on the container wall for a 8-mm stroke and two different values of the hardening coefficient ($m = 0.05$)

The contact pressures on the container wall are plotted in Fig. 22 for a stroke limited to 8 mm and two different materials ($n = 0$ and $n = 0.17$). There is a very good agreement between the results of Schrader et al. and the ALE model. The curves are less close to each other at the level of both punches. Nevertheless, the global shapes of the pressure fields are quite similar.

Schrader et al. also studied the influence of the initial height h_0 of the cylindrical billet on the obtained results. The curves obtained for ratios h_0/d_0 of 0.75, 1.0 (reference value) and 1.25 are presented in Fig. 23. Once again, the ALE results are very close to the published results of Schrader. The largest difference is observed

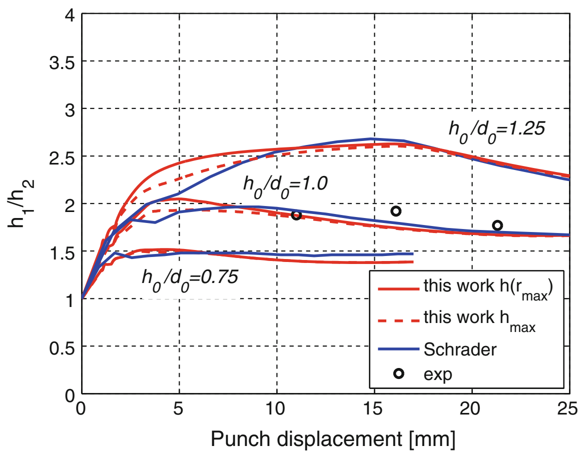


Fig. 23 Influence of the initial thickness h_0 of the cylindrical billet on the cup height ratio. The friction is kept constant ($m = 0.05$)

for the ratio $h_0/d_0 = 0.75$. In this case, the cup height ratio h_1/h_2 that is computed by the ALE model is a 5 % underestimate of the value computed by DEFORM-2D. Since some assumptions have been made about the supposed treatment of friction and the initial yield stress of the material, this difference may still be considered as rather small. Consequently it can concluded that the results of the ALE model are consistent with the ones obtained by a remeshing procedure.

5 Application to Thixoforming

In this section, the ALE model of the previous sections is used to simulate a semi-solid forming operation, also known as thixoforming. This kind of process relies on a specific behaviour, called *thixotropy*, of some alloys near their melting temperature. They behaves as solids at rest (a billet can sustain its own weight) but they react as liquids during shearing (for example, they can be cut easily).

A thermomechanical constitutive law which models a smooth transition between these two behaviours has been implemented in Metafor [16]. The numerical validation of this law is performed using the ALE model of DCET and the results of a campaign of experimental tests which was conducted at the University of Liège by Pierret, Vaneetveld and Rassili [19, 20] in collaboration with the industrial engineering and mechanical production laboratory of ENSAM (Ecole Nationale Supérieure d'Arts et Métiers, Metz, France).

The adapted numerical model exhibits several additional difficulties compared to the one presented previously: all the material parameters are temperature dependent and a coupled thermomechanical integration scheme is used. The heat transfer between the material (a 100Cr6 steel alloy heated up to 1370 °C) and the rigid tools (initially at 130 °C) is also taken into account. The upper punch velocity, which plays

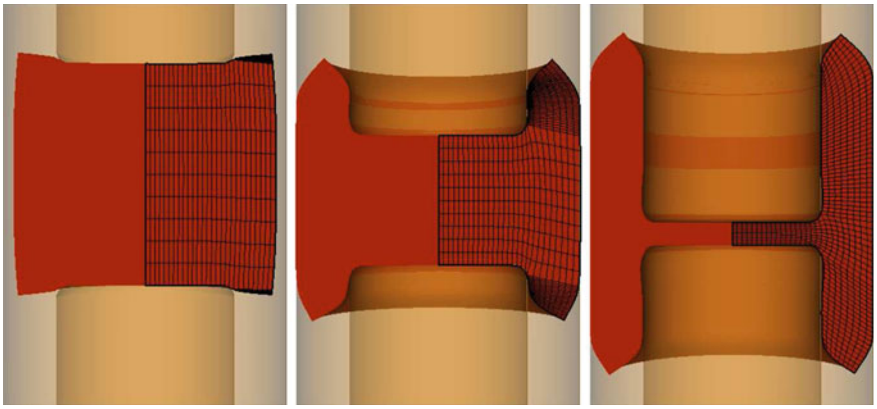


Fig. 24 Illustration of the mesh evolution during the extrusion test

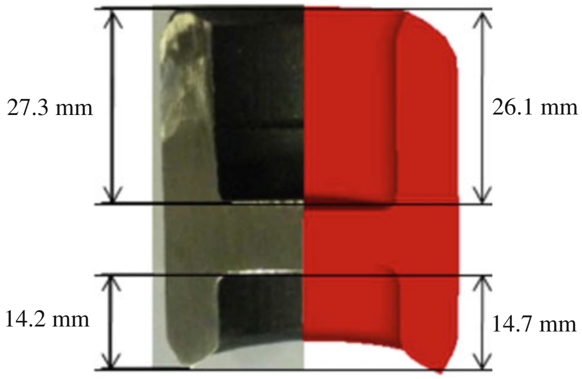


Fig. 25 Comparison of the final shape obtained experimentally (*left*) and the final deformed section resulting from the ALE simulation with a friction coefficient $m = 0.35$ (*right*)

a significant role on the process due to the variable viscosity of the thixotropic material, is not constant. Finally, there is an initial gap between the billet and the container wall at the beginning of the process (see Fig. 24). The filling of this gap requires the definition of an supplementary stage in the time-integration sequence.

Figure 25 presents the final shape of the billet obtained experimentally and numerically. Although the friction coefficient has been chosen to get almost the same cup heights in both cases, it is interesting to see that the simulated upper and lower boundaries of the cups are very similar to the experimental one. This simulation also proves that the mesh management technique presented in this chapter is able to deal with complex material flows.

6 Conclusions

An original 2D model of double cup extrusion test (DCET) has been presented in this chapter. This model efficiently uses the ALE formalism in order to avoid a series of complex and costly remeshing steps during the simulation. Since the DCET is a tribological test, the model is also very interesting to validate the contact treatment on an ALE mesh. An error in the ALE computation of the local friction force would be immediately reflected on the global final shape of the deformed billet.

In order to keep a constant mesh topology, which is a prerequisite condition to use the ALE formalism, it is necessary to add two very thin auxiliary material regions to the initial mesh of the billet. These regions are made up of flat elements which can inflate during the simulation when the billet is crushed between the punches and the material flows from one mesh to the other. Although this particular mesh management technique has been already used by [10, 21], it is the first time that this kind of method is applied to a geometrically-complex process. Indeed, the noses of the punches have not been simplified in the DCET model. They are not planar and their curvature adds a real difficulty to the definition of the mesh motion and the time-integration sequence.

The presented ALE model has been validated by two different means. It has been compared first with an equivalent Lagrangian model during the beginning of the simulations. Secondly, the ALE results have been compared to the ones computed by DEFORM-2D which makes use of an automatic remeshing procedure. A very good agreement has been observed between these two numerical techniques although they are radically different.

Finally, the ALE model of DCET has been used in the frame of a fully-coupled thermomechanical simulation of a semi-solid forming process. Once again, very good results have been obtained without any remeshing operations.

References

1. Atzema EH, Huétink J (1995) Finite element analysis of forward/backward extrusion using ALE techniques. In: Shen Dawson (ed) Simulation of materials processing: theory, methods and applications : proceedings of the 5th international conference NUMIFORM. New-York
2. Bay N (1994) The state of the art in cold forging lubrication. J Mater Process Technol 46(1–2):19–40. doi:10.1016/0924-0136(94)90100-7
3. Benson DJ (1989) An efficient, accurate, simple ale method for nonlinear finite element programs. Comput Methods Appl Mech Eng 72(3):305–350. doi:10.1016/0045-7825(89)90003-0
4. Benson DJ (1992) Computational methods in lagrangian and eulerian hydrocodes. Comput Methods Appl Mech Eng 99(2–3):235–394. doi:10.1016/0045-7825(92)90042-I
5. Boman R, Ponthot JP (2012) Efficient ale mesh management for 3d quasi-eulerian problems. Int J Numer Meth Eng 92(10):857–890. doi:10.1002/nme.4361
6. Boman R, Ponthot JP (2013) Enhanced ALE data transfer strategy for explicit and implicit thermomechanical simulations of high-speed processes. Int J Numer Meth Eng 53(0):62–73. doi: http://dx.doi.org/10.1016/j.ijimpeng.2012.08.007

7. Buschhausen A, Weinmann K, Lee JY, Altan T (1992) Evaluation of lubrication and friction in cold forging using a double backward-extrusion process. J Mater Process Technol 33(1–2):95–108. doi:10.1016/0924-0136(92)90313-H
8. Donéa J, Huerta A, Ponthot JP, Rodriguez-Ferran A (2004) Encyclopedia of computational mechanics, chap 14: arbitrary Lagrangian-Eulerian methods, Vol 1. Wiley, pp 413–437. doi:10.1002/0470091355.ecm009
9. Forcellese A, Gabrielli F, Barcellona A, Micari F (1994) Evaluation of friction in cold metal forming. J Mater Process Technol 45(1–4):619–624. doi:10.1016/0924-0136(94)90408-1
10. Gadala MS, Movahhedy MR, Wang J (2002) On the mesh motion for ale modeling of metal forming processes. Finite Elem Anal Des 38(5):435–459. doi:10.1016/S0168-874X(01)00080-4
11. Gariety M, Ngaile G, Altan T (2007) Evaluation of new cold forging lubricants without zinc phosphate precoat. Finite Elem Anal Des 47(3–4):673–681. doi:10.1016/j.ijmachtools.2006.04.016
12. Geiger R (1976) Der stofffluss beim kombinierten napffliesspressen - metal flow in combined can extrusion - (Berichte aus dem Institut fnr Umformtechnik, UniversitSt Stuttgart). 36, Girardet, Essen, Germany
13. Geijselaers HJM, Huétink J (2000) Semi implicit second order discontinuous Galerkin convection for ALE calculations. In: Onate E, Morgan K, Periaux J, Stein E (eds) (ECCOMAS) European congress on computational methods in applied sciences and engineering, Barcelona
14. Giuliani S (1982) An algorithm for continuous rezoning of the hydrodynamic grid in arbitrary lagrangian-eulerian computer codes. Nucl Eng Des 72(2):205–212. doi:10.1016/0029-5493(82)90216-3
15. Huétink J, Vreede PT, van der Lugt J (1990) Progress in mixed eulerian-lagrangian finite element simulation of forming processes. Int J Numer Meth Eng 30(8):1441–1457. doi:10.1002/nme.1620300808
16. Koeune R (2011) Semi-solid constitutive modeling for the numerical simulation of thixoforming processes. PhD thesis, University of Liège, Belgium
17. Male AT, Cockcroft MG (1965) A method for the determination of the coefficient of friction of metals under condition of bulk plastic deformation. J Inst Met 93:38–46
18. Matweb (2013) Online materials information resource. http://www.matweb.com/
19. Pierret J (2009) Quantification de la robustesse du procédé de thixoformage des aciers. PhD thesis, University of Liège, Belgium.
20. Pierret J, Rassili A, Vaneetveld G, Bigot R, Lecomte-Beckers J (2010) Friction coefficients evaluation for steel thixoforging. Int J Mater Form 3:763–766. doi:10.1007/s12289-010-0882-1
21. Ponthot JP (1995) Advances in Arbitrary Eulerian-Lagrangian finite element simulation of large deformation processes. In: Owen D, Oate E (eds) Computational plasticity: fundamentals and applications -proceedings of the 4th international conference. Pineridge Press Ltd, Barcelona
22. Ponthot JP (1995) Traitement unifié de la mécanique des milieux continus solides en grandes transformations par la méthode des éléments finis. PhD thesis, Université de Liège, Liège, Belgium.
23. Schrader T, Shirgaokar M, Altan T (2007) A critical evaluation of the double cup extrusion test for selection of cold forging lubricants. J Mater Process Technol 189(1–3):36–44. doi:10.1016/j.jmatprotec.2006.11.229
24. Scientific Forming Technologies Corporation (2013) DEFORM. http://www.deform.com/
25. Sofuoglu H, Rasty J (1999) On the measurement of friction coefficient utilizing the ring compression test. Tribol Int 32(6):327–335. doi:10.1016/S0301-679X(99)00055-9
26. Tan X, Bay N, Zhang W (1998) On parameters affecting metal flow and friction in the double cup extrusion test. Scand J Metall 27(6):246–252
27. Van Haaren MJ, Stoker HC, van den Boogaard AH, Huétink J (2000) The ALE-method with triangular elements: direct convection of integration point values. Int J Numer Meth Eng 49(5):697–720. doi:10.1002/1097-0207(20001020) 49:5 $\leq$697:AID-NME976$\geq$3.0.CO;2-U

Part II
Cardiovascular Fluid Mechanics

Simplified Fluid-Structure Interactions for Hemodynamics

Olivier Pironneau

Abstract Computing blood flows in a closed vascular system by isolating one section for simulation creates instabilities due to the time-periodic structure of the flow and possible non-physical back flow in the simplified geometry. We propose some solutions in the context of a simplified fluid structure interaction on a fixed geometry but with pressure dependent normal velocities at the compliant walls.The present analysis is based on the Surface Pressure model for the fluid-structure interactions.

Keywords Blood flow · Hemodynamics finite element method · Pressure boundary conditions · Primary: 91B28, 65L60 · Secondary: 82B31

1 Introduction

Mastering the simulation of blood flow is the key to proper design of by-passes, stents and heart valves (see Thiriet [17] for instance).

The problem was addressed by Charles Peskin in the nineties and his team have made impressive simulations since, using fictitious domains and immersed boundary techniques [1, 12, 13, 18].

Another approach, taken by Quarteroni et al. [5] and the REO project at INRIA [3, 4, 19] is to discretize the full fluid-structure coupled problem with solvers working in moving domains.

In a seminal paper [11], Nobile and Vergana showed that the problem is well posed and conserves energy. Nevertheless the numerical simulations are expensive [2] and there is room for simplifications.

O. Pironneau (✉)
Laboratoire Jacques-Louis Lions, Sorbonne Universités, UPMC,
Boite courrier 187, 75252 Paris Cedex 05, France
e-mail: olivier.pironneau@upmc.fr

In the special case of aortic flow the geometry does not change much. Typically the aorta has a radius of 1cm and a computational geometry deals with a section of length of 5–10 cm; the thickness of the aortic wall is around 0.1 cm; the heart pulse is about 1 Hz and the pressure drop roughly 6 KPa.

In principle arteries are deformable solids subject to large displacements and nonlinear elasticity (e.g. [7, 8, 10]). But when small displacement occurs only and linear elasticity applies, shell models like Koiter's can be used. It was shown in [11] that if lateral displacements are neglected, Koiter's model reduces to a scalar equation for the normal displacement η

$$\rho^s h\partial_{tt}\eta - \nabla\cdot(\mathbf{T}\nabla\eta) - \nabla\cdot(\mathbf{C}\nabla\partial_t\eta) + a\partial_t\eta + b\eta = f^s,\ \ \eta, \partial_t\eta \text{ given at } t=0 \tag{1}$$

on the mean position Σ of the vessel's wall; here h denotes the average thickness of the vessel and ρ^s its volumic mass; $\mathbf{T}$ is the pre-stress tensor (needed because at rest the vessel is blown up by the blood); $\mathbf{C}$ is a damping term, a, b are viscoelastic terms and f^s the external normal force, i.e. $-\sigma^{\mathbf{s}}{}_{nn}$ the normal component of the normal stress at the surface of the solid.

Notice however that the other components of the normal stress tensor cannot be matched with the fluid when the displacement is assumed normal.

Finally assume that $[h, T, C, a] << b$; then the *Surface Pressure Model* is obtained:

$$-\sigma^{\mathbf{s}}{}_{nn} = b\eta,\ \text{with } b = \frac{Eh\pi}{A(1-\xi^2)} \tag{2}$$

where A is the vessel's cross section, E the Young modulus, ξ the Poisson coefficient. Some typical values (MKSA):

$$E = 3MPa,\ \ \xi = 0.3,\ \ A = \pi R^2,\ \ R = 0.01,\ \ h = 0.001,\ \ \Rightarrow\ b = 3.310^7\mathrm{ms}^{-2} \tag{3}$$

2 Boundary Conditions

With simple toroidal coordinates $(r, \theta, \phi) \to (x = R\cos\phi,\ y = R\sin\phi,\ z = r\sin\theta)$ where $R = R_0 + r\cos\theta$,

$$\nabla\cdot u = h_r h_\theta h_\phi \left(\partial_r \frac{u_r}{h_\theta h_\phi} + \partial_\theta \frac{u_\theta}{h_\phi h_r} + \partial_\phi \frac{u_\phi}{h_r h_\theta}\right) \tag{4}$$

with $h_r = 1,\ h_\theta = \frac{1}{r},\ h_\phi = \frac{1}{R}$ because, by definition

$$\frac{1}{h_k^2} = (\partial_k x)^2 + (\partial_k y)^2 + (\partial_k z)^2, \quad k = r, \theta, \phi \tag{5}$$

So $\nabla \cdot u = 0$ and $u \times n = 0$ imply

$$\nabla \cdot u = \partial_r u_r + u_r \frac{R_0 + 2r\cos\theta}{r(R_0 + r\cos\theta)} = 0 \Rightarrow \partial_r u_r|_{\partial\Omega} = -\frac{u_r}{r}\frac{R_0 + 2r\cos\theta}{R_0 + r\cos\theta} \tag{6}$$

Similarly

$$\nabla u = \sum_i e^i h_i \otimes \partial_k \left(\sum_k e^k u_k \right), \quad i, k \in (r, \theta, \phi) \tag{7}$$

with

$$\begin{aligned} e^r &= (\cos\theta\cos\phi, \cos\theta\sin\phi, \sin\theta)^T, \\ e^\theta &= (-\sin\theta\cos\phi, -sin\theta\sin\phi, \cos\theta)^T, \quad e^\phi = (-\sin\phi, \cos\phi, 0)^T \end{aligned} \tag{8}$$

Thus

$$n^T(\nabla u)n = \partial_r u_r + \frac{u_r}{r}\left(1 + \frac{r}{R}\cos^2\theta\right) \Rightarrow \sigma^f_{nn} = p + 2\left(1 + \frac{r}{R}\cos^2\theta\right)\frac{\mu}{r}u \cdot n. \tag{9}$$

Hence the matching conditions at the fluid-structure interface on a torus of small radius r and big radius R are

$$\partial_t \eta = u \cdot n, \quad p = 2\left(1 + \frac{r}{R}\cos^2\theta\right)\frac{\mu}{r}\partial_t \eta + b\eta \tag{10}$$

Notice that (10) implies

$$\partial_t p = 2\left(1 + \frac{r}{R}\cos^2\theta\right)\frac{\mu}{r}\partial_t u \cdot n + bu \cdot n \tag{11}$$

3 Moving Fluid Domains Versus Fixed Domains

3.1 Energy Considerations

Assuming the fluid Newtonian and incompressible, the pressure p and the velocity u are given by the Navier-Stokes equations

$$\rho^f \left(\frac{\partial u}{\partial t} + u \cdot \nabla u\right) - \nabla \cdot \sigma^f = 0, \quad \nabla \cdot u = 0, \tag{12}$$

where ρ^f is the volumic mass of the fluid, μ the viscosity and $\sigma^f = -p\mathbf{I} + \mu(\nabla u + \nabla u^T)$ is the stress tensor.

To check the energy budget one multiplies (12) by u and integrates by parts:

$$\int_\Omega \left[\frac{\rho^f}{2} \partial_t |u|^2 + \frac{\mu}{2}(\nabla u + \nabla u^T) : (\nabla u + \nabla u^T) \right] + \int_{\partial\Omega} \frac{\rho^f}{2} |u|^2 u \cdot n = \int_{\partial\Omega} \sigma^s \cdot u \cdot n \tag{13}$$

The fluid velocity on $\partial\Omega$ is equal to the wall velocity, so (see [5])

$$\int_{\Omega(t)} \frac{1}{2} \partial_t |u|^2 + \int_{\partial\Omega} \frac{1}{2} |u|^2 u \cdot n = \partial_t \int_{\Omega(t)} \frac{1}{2} |u|^2 \tag{14}$$

This leads to the following energy identity

$$\int_{\Omega(T)} \frac{\rho^f}{2} |u|^2(T) + \int_{\Omega\times(0,T)} \frac{\mu}{2} |\nabla u + \nabla u^T|^2 = \int_{\Omega(0)} \frac{\rho^f}{2} |u|^2(0) + \int_{\partial\Omega\times(0,T)} \sigma^s \cdot u \cdot n \tag{15}$$

3.2 The Problem in Strong Form

Now if we consider (12) on a fixed domain with zero tangential velocities but non-zero normal velocities on the walls then to conserve energy we need to change $u \cdot \nabla u$ into $u \cdot \nabla u - \frac{1}{2}\nabla|u|^2$ which happens to be $-u \times \nabla \times u$ due to the identity

$$u \cdot \nabla u = \frac{1}{2} \nabla |u|^2 - u \times \nabla \times u. \tag{16}$$

Let us recall another identity:

$$-\Delta u = \nabla \times \nabla \times u + \nabla\nabla \cdot u \tag{17}$$

Therefore the modified Navier-Stokes system suited to flows in fixed domains with zero tangential components on the walls ($u \times n = 0$) is

$$\rho^f \left(\frac{\partial u}{\partial t} - u \times \nabla \times u \right) + \mu \nabla \times \nabla \times u + \nabla p = 0, \quad \nabla \cdot u = 0, \tag{18}$$

In a domain Ω with $u \cdot n = 0$ and p related by (11) on $\partial\Omega$, as shown below.

3.3 The Problem in Variational Form

Its variational formulation of is: find u, p such that $\forall \hat{u}$, $\hat{p}$ with $\hat{u} \times n|_{\partial\Omega} = 0$,

$$\int_\Omega \left[\rho^f \left(\frac{\partial u}{\partial t} - u \times \nabla \times u\right) \cdot \hat{u} + \mu \nabla \times u \cdot \nabla \times \hat{u} - p \nabla \cdot \hat{u} - \hat{p} \nabla \cdot u\right] + \int_{\partial\Omega} p\hat{u} \cdot n = 0. \tag{19}$$

with p related to $u \cdot n$ by (11).

Problem 1 Find u, p, η such that $\forall \hat{u}, \hat{p}, \hat{\eta}$ with $\hat{u} \times n|_{\partial\Omega} = 0$, u and η given at $t = 0$,

$$\int_\Omega [\rho^f \left(\frac{\partial u}{\partial t} - u \times \nabla \times u\right) \cdot \hat{u} + \mu \nabla \times u \cdot \nabla \times \hat{u} - p \nabla \cdot \hat{u} - \hat{p} \nabla \cdot u] + \int_{\partial\Omega} [(\alpha \partial_t \eta + b\eta)\hat{u} \cdot n + b\hat{\eta}(\partial_t \eta - u \cdot n)] = 0. \tag{20}$$

with $\alpha = 2\frac{\mu}{r}\left(1 + \frac{r}{R}\cos^2\theta\right)$. As (20) implies (10-a),
energy estimates derive by choosing $\hat{u} = u$, $\hat{p} = p$, $\hat{\eta} = \eta$

$$\int_\Omega \rho^f |u|^2(T) + \int_{\partial\Omega} b\eta^2(T) + \int_{\Omega\times(0,T)} 2\mu|\nabla \times u|^2 + 2\int_{\partial\Omega\times(0,T)} \alpha(\partial_t \eta)^2 = \int_\Omega \rho^f |u|^2(0) + \int_{\partial\Omega} b\eta^2(0) \tag{21}$$

3.4 Approximation with the Nedelec Edge Element

Boundary conditions like $u \times n$ are hard to enforce. Furthermore boundary conditions involving the pressure have their own difficulties (see [15, 16]). In [6] it is argued that finite element approximations of (24) requires edge elements. An error analysis is given with $P^k - P^{k-1}$ discontinuous elements with degrees of freedom being edge fluxes of degree k plus face fluxes of degree $k-1$ and volume fluxes of degree $k-2$ for the velocities.

Although the proof of convergence is done for $k \geq 2$, we tested the same idea with P^1 Raviart-Thomas elements (called RT^0) for the velocity and P^0 discontinuous elements for the pressure. In theory η should be P^0-discontinuous like the pressure; first we took it P^1-continuous to simplify the implementation because then we can add to the formulation a small regularization $-\epsilon\Delta\eta$ everywhere in Ω so as to avoid having degrees of freedom for η only on the boundary.

Then we tested also η approximated with the P^1 Raviart-Thomas element and formulated the laplacian of η in mixed form; this augments considerably the number of degree of freedom: $3*(n_v+n_e)+2*n_v$ for the $P^2-P^1-P^1$ element (tested in [14], see also below), $3*n_e+n_t+2*n_v$ for the $RT^0-P^1-P^1$ element and $6*n_e+2*n_t+2*n_v$ for the $RT^0-P^0-RT^0+P^0$ element, where n_v is the number of vertices, n_e the number of edges, n_t the number of elements. We tested these 3 sets of element on a simple geometry: a quarter of a torus with a pressure drop imposed from the top horizontal cross section to the right vertical one. The cross section of the torus is a circle of radius 1 cm. This circle is extruded on a greater circle of radius 4 cm. The pressure drop is $6\cos(\pi t)$, $b=200$ and $\nu=0.001$.

The time step is 0.05. The mesh has $n_v=1395$, $n_t=6120$, $n_e=1336$. The computation is stopped at $t=0.75$.

The results are shown on Fig. 4. On a core i7@2.3MHz it takes 17 s with the Nedelec-P^1-P^1 element to compute 16 time steps with the characteristic-Galerkin method for the non-linear terms (see [14]) and 22 s with the Nedelec/Raviart-Thomas element (see Fig. 1).

4 A Formulation Where the Displacement is Eliminated

Notice that η can be eliminated from (10), giving a formulation which contains $u\times n=0$ and

$$n\partial_t p=\alpha\partial_t u+bu \tag{22}$$

4.1 A Time Discretisation

Consider now (19) discretized in time :

$$\int_\Omega\left[\rho^f\left(\frac{u^{m+1}-u^m}{\delta t}-u^{m+\frac12}\times\nabla\times u^m\right)\cdot\hat u+\mu\nabla\times u^{m+\frac12}\cdot\nabla\times\hat u-p^{m+1}\nabla\cdot\hat u\right.$$
$$\left.-\hat p\nabla\cdot u^{m+\frac12}\right]+\int_{\partial\Omega}p^{m+1}\hat u\cdot n=0. \tag{23}$$

We use (22) discretized in time to compute $p^{m+1}|_{\partial\Omega}$ and so we consider

Problem 2 Find u, p such that $\forall\hat u$, $\hat p$ with u and $\partial_t p$ given at $t=0$,

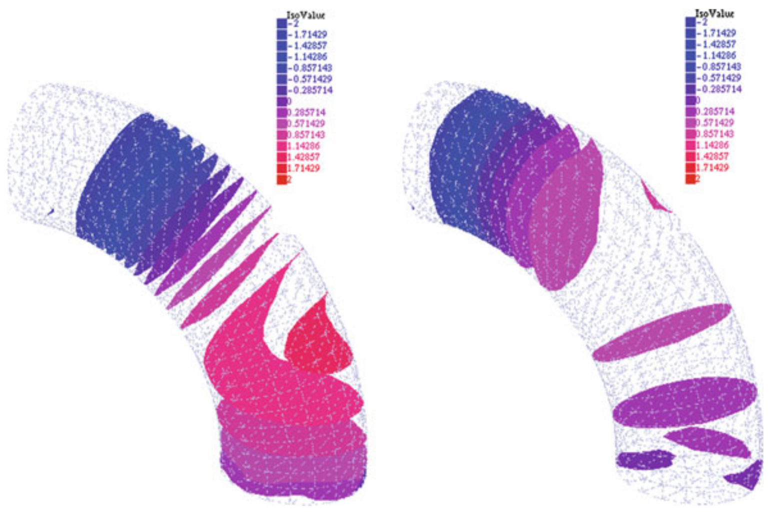


Fig. 1 *Left* Surfaces of constant pressure for a flow with $\nu = 10^{-3},\ b = 200$ in a quarter of a torus with $R = 4,\ r = 2$ discretized on a fixed geometry with the Nedelec edge element for the velocity, peacewise constant pressures and linear continuous deformation. *Right* same as *left* but with a mixte Raviart-Thomas element for the displacement

$$\int_\Omega \left[\rho^f \left(\frac{u^{m+1}-u^m}{\delta t} - u^{m+\frac{1}{2}} \times \nabla \times u^m\right) \cdot \hat{u} + \mu \nabla \times u^{m+\frac{1}{2}} \cdot \nabla \times \hat{u} - p^{m+1} \nabla \cdot \hat{u} \right.$$
$$\left. - \hat{p} \nabla \cdot u^{m+\frac{1}{2}} \right] + \int_{\partial\Omega} [\delta t b u^{m+\frac{1}{2}} + \alpha(u^{m+1} - u^m) + p^m n] \cdot \hat{u} = 0. \qquad (24)$$

Formulation (19) is valid only if $\hat{u} \times n = 0$. This condition has been removed from (24) to make it symmetric and easy to implement but the consequence is that by working the integrations by parts backward, it is found that this formulation implies (18) and on $\partial\Omega$:

$$[\delta t b u^{m+\frac{1}{2}} + \alpha(u^{m+1} - u^m)] \cdot n - (p^{m+1} - p^m), \qquad \nabla \times u^{m+\frac{1}{2}} \times n = 0 \quad (25)$$

The first condition no longer implies that $u \times n = 0$ and the second condition is like saying that the tangential stress is zero, which means that we match not only the normal components of the fluid and solid normal stress but all the components.

In summary Problem 2 is different from Problem 1; both of them have physically sound background but we need to test them numerically to see how different they are.

4.2 Discretization with a Finite Element Method

Let T_h be a triangulation with K tetraedra $\{T_k\}_1^K$ with the usual conformity hypotheses; let $\Omega := \cup_k T_k \subset \mathbb{R}^3$.

Consider the $P^2 - P^1$ element built from

$$\begin{aligned} V_h &= \{v \in C^0(\overline{\Omega})^3 \ : \ v_i|_{T_k} \in P^2, \ i = 1, 2, 3\} \\ Q_h &= \{q \in C^0(\overline{\Omega}) \ : \ q|_{T_k} \in P^1\} \end{aligned} \tag{26}$$

We assume that the boundary is made of two part, Σ which is the compliant wall and the input and output sections Γ on which p is given and $u \times n = 0$.

4.3 Discretization of Problem 1

For simplicity we assume that $r << R$, i.e. $\alpha = 1$. The momentum equation is also divided by ρ^f and $\nu = \mu/\rho^f$ and b is changed into b/ρ^f.

A feasible discretization of (24) is to find $[u^{m+1}, p^{m+1}, \eta^{m+1}] \in V_h \times Q_h \times Q_h$ with $u^{m+1} \times n|_\Gamma = 0$, $\eta^{m+1}|_\Gamma = 0$ and such that

$$\begin{aligned} &\int_\Omega \left[\hat{u} \cdot \left(\frac{u^{m+1} - u^m}{\delta t} - u^{m+\frac{1}{2}} \times \nabla \times u^m\right) - p^{m+1}\nabla \cdot \hat{u} - \hat{p}\nabla \cdot u^{m+\frac{1}{2}}\right] \\ &\quad + \int_\Omega \nu\nabla \times u^{m+\frac{1}{2}} \cdot \nabla \times \hat{u} + \varepsilon\nabla\eta^{m+\frac{1}{2}} \cdot \nabla\hat{\eta}] \\ &\quad + \int_\Sigma b\left[\eta^{m+\frac{1}{2}}\hat{u}_n - \hat{\eta}\left(u_n^{m+\frac{1}{2}} - \frac{1}{\delta t}(\eta^{m+1} - \eta^m)\right) + \frac{1}{\epsilon}(u^{m+\frac{1}{2}} \times n) \cdot (\hat{u} \times n)\right] \\ &= -\int_\Gamma p_\Gamma \hat{u}_n, \ \ \forall\, [\hat{u}, \hat{p}, \hat{\eta}] \in V_h \times Q_h \times Q_h \text{ with } \hat{u} \times n|_\Gamma = 0, \ \hat{\eta}|_\Gamma = 0. \end{aligned} \tag{27}$$

where ε is any small positive parameter.

When Ω is kept fixed, an energy consevation identity is found by choosing $\hat{u} = u^{m+\frac{1}{2}}$, $\hat{p} = -p^{m+1}$, $\hat{\eta} = \eta^{m+\frac{1}{2}}$:

$$\begin{aligned} &\int_\Omega \left[\frac{u^{m+1^2} - u^{m2}}{\delta t} + \nu|\nabla \times u^{m+\frac{1}{2}}|^2 + \varepsilon|\nabla\eta^{m+\frac{1}{2}}|^2\right] \\ &\quad + \int_\Sigma \frac{\eta^{m+1^2} - \eta^{m2}}{\delta t} + \frac{1}{\epsilon}\int_\Sigma |u^{m+\frac{1}{2}} \times n|^2 = -\int_\Gamma p_\Gamma \hat{u}_n^{m+\frac{1}{2}} \end{aligned} \tag{28}$$

As for the Navier-Stokes equations, when δt is small enough the problem has a unique solution because of the energy estimate and because of a general inf-sup condition is satisfied with p replaced by $[p, \eta]$.

4.4 Discretization of Problem 2

A feasible discretization of (24) is to find $u^{m+1} \in V_h,\ p^{m+1} \in Q_h$ such that

$$\begin{aligned}&\int_\Omega \left[\hat{u}\cdot\left(\frac{u^{m+1}-u^m}{\delta t} - u^{m+\frac{1}{2}} \times \nabla \times u^m\right) - p^{m+1}\nabla\cdot\hat{u} - \hat{p}\nabla\cdot u^{m+\frac{1}{2}}\right]\\ &\quad+\int_\Omega \nu\nabla\times u^{m+\frac{1}{2}}\cdot\nabla\times\hat{u}\\ &\quad+\int_\Sigma (u^{m+\frac{1}{2}}b\delta t + p^m n)\cdot\hat{u} = -\int_\Gamma p_\Gamma \hat{u}_n\\ &\quad\forall\hat{u}\in V_h,\ \hat{p}\in Q_h \text{ with } \hat{u}\times n|_\Gamma = 0 \end{aligned} \tag{29}$$

Notice that $u^{m+1} \times n|_\Sigma = 0$ is implied by the formulation. When Γ is flat that condition amounts to some component of the velocity being zero which is easy to implement.

Notice that the energy equality implies stability only so long a p remains bounded on Σ, which could possibly be derived from (29), but not so obviously:

$$\begin{aligned}\int_\Omega\left[\frac{u^{m+1^2}-u^{m2}}{\delta t} + \nu|\nabla\times u^{m+\frac{1}{2}}|^2\right] + \int_\Sigma b|u^{m+\frac{1}{2}}|^2\delta t\\ = -\int_\Sigma p^m u_n^{m+\frac{1}{2}} - \int_\Gamma p_\Gamma \hat{u}_n^{m+\frac{1}{2}}\end{aligned} \tag{30}$$

5 Numerical Tests

5.1 Moving the Geometry for Graphic Visualization

The full model requires that Σ be moved at every time step along its normal of a quantity $\delta t u^m \cdot n$. To preserve the triangulation we follow the literature [2] and solve an additional problem

$$-\Delta d^{m+1} = 0 \text{ in } \Omega,\quad d^{m+1}|_\Sigma = d^m + n\delta t u_n^m,\quad d^{m+1}|_\Gamma = 0 \tag{31}$$

and then move every vertex q^j of the triangulation $q^j \to q^j + \kappa d$. In theory $\kappa = 1$ but for graphic enhancement it can be adjusted. Note however that (31) is expensive.

5.2 Comparison of the Two Methods

On the problem described earlier both methods give very similar results as shown on Fig. 2. The geometry is updated for visualisation purpose with a multiplicative factor 100.

The geometry is a section of the aorta obtained from a MRI scan. It has 4991 vertices, giving 19964 degrees of freedom for each linear systems for $[u_1^{m+1}, u_2^{m+1}, u_3^{m+1}, p^{m+1}]$. The pressure drop from inflow section on the right to outflow section on the left is $p_{\Gamma_R} = 6\cos^2(\pi t)$ and the results are shown at $t = 0.8$. On the smaller cross sections a pressure drop equal to $p_{\Gamma_R}/2$ is imposed. Problem 1 and Problem 2 are solved for comparison with $\delta t = 0.05/\pi,\ \nu = 0.001,\ b = 200$. Results are shown on Fig. 3.

For Problem 1, the computation took $198''$ on a macbook pro $15''$, 2012, 2.3MHz core i7. For Problem 2 it took $180''$. The results are very similar with some difference on the pressure but very little on the velocities.

6 Inflow/Outflow Conditions by PML

We end this article with an idea to address the problem of loss of stability due to the creation of reverse flow in unwanted regions because of the boundary conditions on the artificial inflow and out flow sections.

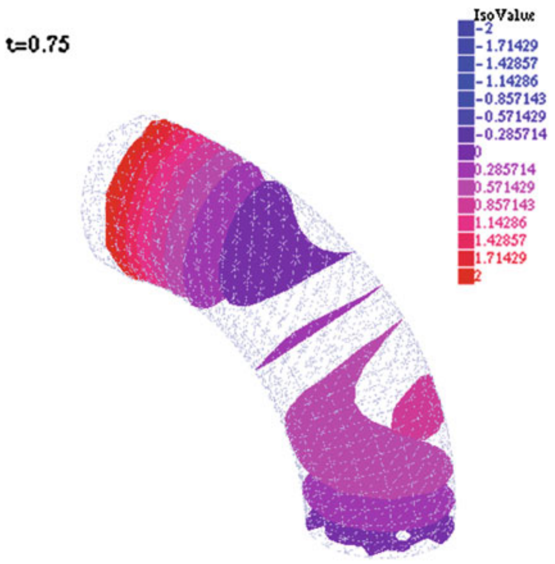
We borrow the idea from the PML literature (see for example [9]) and add to the artery geometry a viscous buffer after Γ_{out} where $\nu = \nu_1 >> \nu_{blood}$ (and similarly before Γ_{in} but we present the theory applied to the outflow section only).

Consider a geometry Ω where the exit section is $\Gamma_o = \{0\} \times [0, h]$ in 2D where pressure is set to p_0 while pressure is set to p_1 on entry. Assume that we impose a parabolic flow $u = Ky(h - y)$ at the exit of a viscous buffer $\mathbf{L} = [-L, 0) \times [0, h]$, i.e. on $\{-L\} \times [0, h]$. Now we solve the Navier-Stokes equations on $\Omega \cup \mathbf{L}$. The problem is to choose K so that the pressure on the inital outflow boundary Γ_o is unchanged in the mean, namely $\bar{p}_0 := h^{-1} \int p_0 \mathrm{d}y$.

Because at every time step the system to solve is linear we shall adjust K by superposition so that the mean pressure is $\bar{p}_0$ on Γ_{out}. Since, $p|_{\Gamma_{out}} \approx \bar{p}_1 + (\bar{p}_2 - \bar{p}_1)\frac{K-K_1}{K_2-K_1}$ where $\bar{p}_1$ is computed with $K = K_1$ and $\bar{p}_2$ the mean pressure when $K = K_2$, then

$$K = K_1 + (K_2 - K_1)\frac{\bar{p}_0 - \bar{p}_1}{\bar{p}_2 - \bar{p}_1} \tag{32}$$

This requires to solve the linear Stokes-like system at each time step 3 times. We can also add K to the unknowns of the Stokes-like linear system and add $\int_{\Gamma_{out}} p = |\Gamma_{out}| p_0$ to the equations; we used this second solution in the numerical tests because it is much less computer intensive.



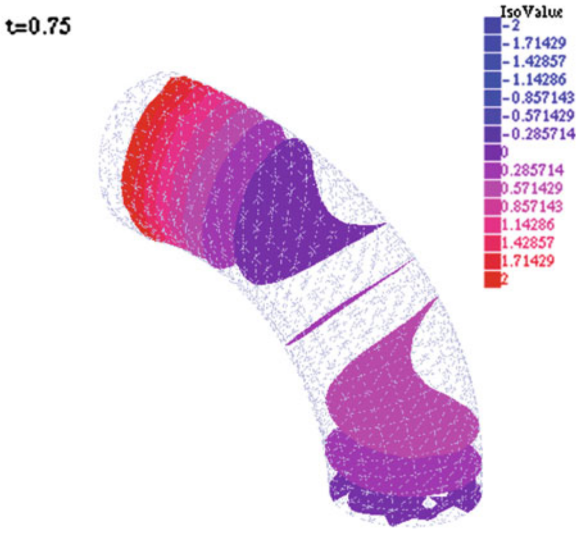


Fig. 2 *Left* surface of equal pressure at $t = 0.75$ computed by solving Problem 1 with $P^2 - P^1 - P^1$ elements and a penalization of the condition $u \times n = 0$. *Right* same as *left* but with Problem 2 and a $P^2 - P^1$ element

The idea is tested numerically on a quarter of a 2D-torus with radii 0.6 and 1 with $\nu = 0.002$ and a pressure drop equal to $\cos(t) + \cos(3t),\ t \in (0, 25)$. The PML viscosity is $\nu_1 = 0.2$. A PML region is added to both ends of the tube. Results are shown on Fig. 4.

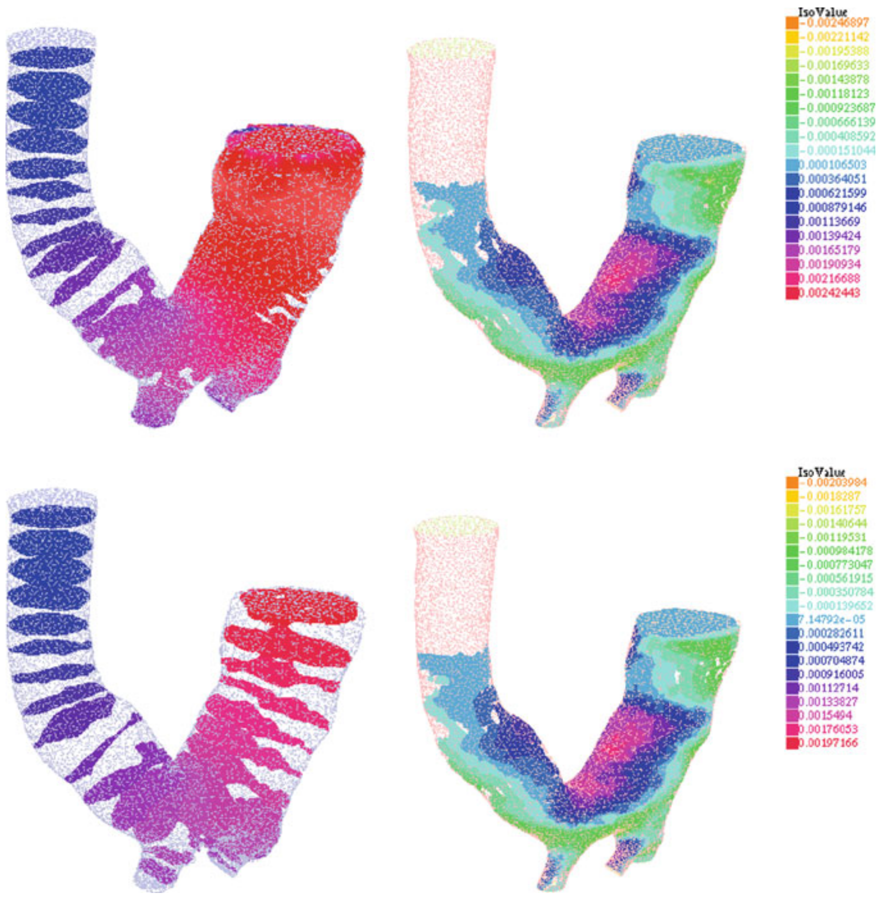


Fig. 3 Computation of $[u, p]$ for Problems 1 & 2 for a portion of an oarta (shown *upside down*). *Top* with Problem 1. the pressure is shown at $t = 0.8$ on the *left* on a geometry which has changed by η. On the *right* the third component of the velocity w is shown on the fixed geometry. *Bottom* same for Problem 2

The results look very different and that is because both computations do not have the same inflow and outflow conditions on the original inflow/outflow boundaries. In one case the pressure is imposed pointwise with $u \times n = 0$, in the PML case the mean pressure is imposed and no conditions are imposed on the velocity but parabolic velocity is imposed on the inflow/outflow of the PML boundaries.

The method will be tested in 3D and reported in a future publication.

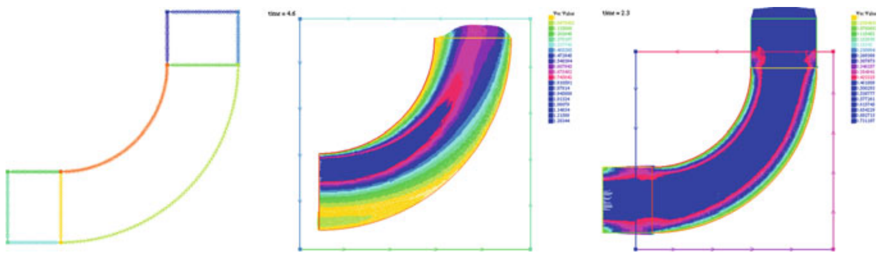


Fig. 4 *Left* Geometry for the flow with two PML regions added. *Center* the velocity vectors computed without the PML; notice the back flow in the *yellow* region. *Right* the same flow (velocity vectors) computed with the two PML regions. The pressure drop from the two inner boundaries (corresponding to the *top* and *left* boundaries of the geometry on the *center* figure) are the same as in the *center* figure

7 Conclusion

In this article we have presented problems and solutions encountered with fluid-structure interactions when a middle solution is seeked: neither the full problem with moving geometries because it is too expensive, nor rigid walls because it is not precise enough and it doesn't give the geometrical deformation.

The solution adopted here is to delay the geometrical deformations to the graphic diplay only. But in doing so we have to work with the Navier-Stokes equations with unusual boundary conditons which require unusual finite element discretizations.

For these intermediary problems we have shown that it is important to preserve energy. Furthermore we can choose either to match exactly the normal component of the solid and fluid normal stress tensor or to match approximately the 3 componenets of the normal stresses by relaxing slightly the no slip condition.

In all cases the problem of back flows in the pulsating cases remains. We have suggested a possible solution and made some preliminary tests.

References

1. Boffi D, Gastaldi L (2003) A fem for the immersed boundary method. Comput Struct 81:491–501
2. Deparis S, Fernandez MA, Formaggia L (2003) Acceleration of a fixed point algorithm for fluid-structure interaction using transpiration conditions. ESAIM:M2AN 37(4):601–616
3. Fernandez M (2013) Incremental displacement-correction schemes for incompressible fluid-structure interaction. Numer Math 123:21–65
4. Formaggia L, Gerbeau JF, Nobile F, Quarteroni A (2001) On the coupling of 3d and 1d navier-stokes equations for flow problems in compliant vessels. Comput Methods Appl Mech Eng 191:561–582
5. Formaggia L, Quarteroni A, Veneziani A (eds) (2009) Cardiovasuclar mathematics, MS and A series. Springer, Milano

6. Girault V (1988) Incompressible finite element methods for Navier-Stokes equations with nonstandard boundary conditions in R^3. Math Comp 51(183):55–74
7. Gonzalez O (2000) Exact energy and momentum conserving algorithms for general models in nonlinear elasticity. Comput Methods Appl Mech Eng 190:1763–1783
8. Gonzalez O, Simo JC (1996) On the stability of symplectic and energy-momentum algorithms for nonlinear Hamiltonian systems with symmetry. Comput Methods Appl Mech Eng 134:197–222
9. Hu Fang Q, Li XD, Lin DK (2008) Absorbing boundary conditions for nonlinear euler and navier-stokes equations based on the perfectly matched layer technique. J Comp Phys 227:4398–4424
10. Le Tallec P (2001) Fluid structure interaction with large structural displacements. Comput Methods Appl Mech Eng 190:3039–3067
11. Nobile F, Vergana C (2008) An effective fluid-structure interaction formulation for vascular dynamics by generalized robin conditions. SIAM J Sci Comp 30(2):731–763
12. Peskin C, McQueen D (1989) A three dimensional computational method for blood flow in the hearth-i. immersed elastic fibers in a viscous incompressible fluid. J Comput Phys 81:372–405
13. Peskin C (2002) The immersed boundary method. Acta Numerica 11:479–517
14. Pichon KG, Pironneau O (2014) Pressure boundary conditions for blood flows. Applied Math Conf in honnor of L. Tartar. Proc published in AIMS journal (to appear)
15. Pironneau O (1986) Conditions aux limites sur la pression pour les équations de Stokes et de Navier-Stokes. C R Acad Sci Paris Sér I Math 303(9):403–406
16. Pironneau O (1989) Finite element methods for fluids. Wiley, New York
17. Thiriet M (2011) Biomathematical and biomechanical modeling of the circulatory and ventilatory systems. Control of cell fate in the circulatory and ventilatory systems, vol 2. Springer, New York
18. Usabiaga F, Bell J, Buscalioni R, Donev A, Fai T, Griffith B, Peskin C (2012) Staggered schemes for fluctuating hydrodynamics. Multiscale Model Sim 10:1369–1408
19. Vignon-Clementel I, Figueroa A, Jansen K, Taylor CA (2006) Outflow boundary conditions for three-dimensional finite element modeling of blood flow and pressure in arteries. Comput Methods Appl Mech Eng 195:3776–3796

Patient-Specific Cardiovascular Fluid Mechanics Analysis with the ST and ALE-VMS Methods

Kenji Takizawa, Yuri Bazilevs, Tayfun E. Tezduyar, Christopher C. Long, Alison L. Marsden and Kathleen Schjodt

Abstract This chapter provides an overview of how patient-specific cardiovascular fluid mechanics analysis, including fluid–structure interaction (FSI), can be carried out with the space–time (ST) and Arbitrary Lagrangian–Eulerian (ALE) techniques developed by the first three authors' research teams. The core methods are the ALE-based variational multiscale (ALE-VMS) method, the Deforming-Spatial-Domain/Stabilized ST formulation, and the stabilized ST FSI technique. A good number of special techniques targeting cardiovascular fluid mechanics have been developed to be used with the core methods. These include (i) arterial-surface extraction and boundary condition techniques, (ii) techniques for using variable arterial wall thickness, (iii) methods for calculating an estimated zero-pressure arterial geometry, (iv) techniques for prestressing of the blood vessel wall, (v) mesh generation techniques for building layers of refined fluid mechanics mesh near the arterial walls, (vi) a special mapping technique for specifying the velocity profile at an inflow boundary with non-circular shape, (vii) a scaling technique for specifying a more realistic volumetric flow rate, (viii) techniques for the projection of fluid–structure interface

K. Takizawa (✉)
Department of Modern Mechanical Engineering and Waseda Institute for Advanced Study, Waseda University, 1-6-1 Nishi-Waseda, Shinjuku-ku, Tokyo 169-8050, Japan
e-mail: Kenji.Takizawa@tafsm.org

Y. Bazilevs
Structural Engineering, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093, USA

T. E. Tezduyar · K. Schjodt
Mechanical Engineering, Rice University, 6100 Main Street, Houston, TX 77005, USA

C. C. Long
T-3 Fluid Dynamics and Structural Mechanics, Los Alamos National Laboratory, Los Alamos, NM 87545, USA

A. L. Marsden
Mechanical and Aerospace Engineering, University of California,San Diego, 9500 Gilman Drive, La Jolla, CA 92093, USA

stresses, (ix) a recipe for pre-FSI computations that improve the convergence of the FSI computations, (x) the Sequentially-Coupled Arterial FSI technique and its multiscale versions, (xi) techniques for calculation of the wall shear stress (WSS) and oscillatory shear index (OSI), (xii) methods for stent modeling and mesh generation, (xiii) methods for calculation of the particle residence time, and (xiv) methods for an estimated element-based zero-stress state for the artery. Here we provide an overview of the special techniques for stent modeling and mesh generation and calculation of the residence time with application to pulsatile ventricular assist device (PVAD). We provide references for some of the other special techniques. With results from earlier computations, we show how the core and special techniques work.

1 Introduction

Cardiovascular fluid mechanics modeling even without fluid–structure interaction (FSI) poses formidable computational challenges in some cases. Still, with everything else being equal, taking the FSI between the blood flow and arterial walls into account makes the modeling far more challenging. The Reynolds number at the peak systole, which characterizes the flow regime, is of the order several hundreds in cerebral arteries and a few thousands in the aortic arch. This range of Reynolds numbers corresponds to complex 3D flows, which are, nevertheless, laminar. Mild turbulence is seen only under rare circumstances. Consequently, in general, state-of-the-art computational fluid dynamics (CFD) software can deliver stable and reasonably accurate solutions to cardiovascular fluid mechanics problems that do not require special methods. However, just CFD modeling of blood flow assumes that the blood vessel walls are rigid, which is not the case. Vascular walls can undergo large deformations due to hemodynamic forces. Wall deformations alter the blood flow patterns, which, in turn, alter the hemodynamics. Therefore, for the computational modeling to be realistic, the blood flow and wall deformation need to be treated in a coupled fashion, and that makes it an FSI problem. That is why, although we also present special methods for and computations from some challenging cardiovascular fluid mechanics modeling without FSI, we emphasize cardiovascular FSI modeling in this book chapter.

The blood flow is governed by the Navier–Stokes equations of incompressible flows. The vascular wall is typically assumed to behave as an elastic material that is allowed to undergo large deformations [1]. For simplicity, a hyperelastic framework is typically employed for this purpose (see [1, 2] and references therein). At the luminal surface kinematic and traction compatibility conditions are assumed to hold point-wise. That is, at every point on the luminal surface, the blood flow velocity must equal that of the wall, and the tractions associated with the fluid and structure domains must balance. These conditions are both physically meaningful and lead to a mathematically well-posed problem. To complete the formulation of the FSI problem, the motion of the blood vessel must be determined fully by point-wise computation of the time-dependent wall displacement. This added complexity of the FSI modeling relative to just CFD modeling creates more challenges for the computation, which are discussed here.

Methods for solving the fully-discretized coupled equations are of two kinds: loosely- and strongly-coupled. The strongly-coupled solution methods can further be categorized as "block-iterative," "direct" and "quasi-direct" coupling techniques (see [3, 4] for the terminology). The strongly-coupled solution methods include the monolithic methods, which typically imply matching discretizations at the fluid–structure interface. The quasi-direct and direct coupling methods are applicable to cases with nonmatching fluid and structure meshes at the interface, become equivalent to monolithic methods when the interface meshes are matching, and yield more robust algorithms than block-iterative coupling does, especially when the structure is light compared to the fluid masses involved in the FSI dynamics.

In loosely-coupled approaches, the equations for fluid, solid and mesh motion are solved sequentially, in an uncoupled fashion. Typically, within each time step, the increment in the fluid solution is computed on a fixed spatial domain, the fluid forces on the structure are collected, and the structural-solution increment is computed, which is followed by an update of the mesh position. This enables the use of existing fluid and structural solvers, a significant motivation for adopting this approach. Yet, difficulties in the form of lack of convergence have been noted in a number of cases, including some cardiovascular FSI cases. In strongly-coupled approaches, the equations for fluid, solid and mesh motion are solved in a fully coupled fashion, and in a more direct way in the case of quasi-direct and direct coupling. The main advantage is that monolithic and other quasi-direct and direct coupling techniques are more robust in that many of the convergence problems encountered with the loosely-coupled and block-iterative coupling approaches are completely avoided.

Although introducing FSI to blood flow increases modeling and simulation complexity, the computations produce physiologically more realistic results than those generated by just CFD. Effects of including wall elasticity in vascular simulations have been examined, for example, for carotid artery [5, 6], for cerebral aneurysms [7, 8] and for the total cavopulmonary connection [9], which was the first use of FSI for patient-specific pediatric cardiology applications. The rigid-wall assumption consistently shows an overestimation of the wall shear stress (WSS) compared to the flexible wall, in some cases by as much as 50 %. Some qualitative and quantitative differences between the rigid- and flexible-wall simulations were also observed for the blood flow patterns. We note that the blood vessels of young children are significantly more flexible that those of adults, making FSI modeling especially important for pediatric cardiology [10].

Unlike the rigid-wall assumption, FSI enables simulation of the complete mechanical environment of the vascular wall, both the loads acting on the wall due to blood flow and the loads acting within the wall. The latter loads are particularly important for they act on the cells that control wall structure and function, which in turn may change the bulk elastic properties of the wall and hence the hemodynamics. As a result, considering the associated wall mechanobiology and the importance of mechanics in understanding possible cellular responses presents an important future research direction (see [11] for preliminary results).

In recent years we have seen a rapid increase in the volume and level of research in cardiovascular fluid mechanics modeling (see, for example, [5, 6, 12–22]). Much of this has been in patient-specific modeling of arteries, especially those with aneurysm. The preferred method of handling the moving interfaces involved in FSI modeling has mostly been the Arbitrary Lagrangian–Eulerian (ALE) finite element formulation [23]. A residual-based variational multiscale (RBVMS) formulation in the context of ALE methods (herein referred to as ALE-VMS) was introduced in [24] and employed in cardiovascular FSI simulations of a number of patient-specific models of major blood vessels.

One of the earliest space–time (ST) methods targeting FSI modeling is the Deforming-Spatial-Domain/Stabilized ST (DSD/SST) formulation [25, 26]. It was introduced by the Team for Advanced Flow Simulation and Modeling (T★AFSM) in 1991 as a general-purpose interface-tracking (i.e. moving-mesh) technique for computation of flow problems with moving boundaries and interfaces. It is based on the Streamline-Upwind/Petrov–Galerkin (SUPG) [27] and Pressure-Stabilizing/Petrov–Galerkin (PSPG) [25, 28] methods. An earlier version of the pressure stabilization, for Stokes flows, was introduced in [29]. The stabilized ST formulations were introduced and tested earlier by other researchers in the context of problems with fixed spatial domains (see [30]).

Patient-specific arterial FSI modeling with the DSD/SST formulation was first reported by Torii et al. in a 2004 journal article [5] published by the Japan Society of Mechanical Engineers. Over the years following that, Torii et al. conducted one of the most extensive series of patient-specific arterial FSI modeling of cerebral aneurysms [6, 31]. The cases studied in these articles by Torii et al. were almost all for middle cerebral arteries, and the geometries were constructed from computed tomography (CT) images. In these arterial FSI computations the DSD/SST formulation was used together with the mesh update methods [32] developed by the T★AFSM and was implemented with block-iterative coupling [33]. The inflow boundary condition used in the computations is a pulsatile velocity profile, which closely represents the measured flow rate during a cardiac cycle.

New generation DSD/SST formulations, with increased scope, robustness and efficiency, were introduced by the T★AFSM in [4]. The stabilized ST FSI (SSTFSI) technique, which is based on the new-generation DSD/SST formulations, was also introduced in [4]. The ST-VMS method [34] is the the variational multiscale version of the DSD/SST method, which was originally called “DSD/SST-VMST” (i.e. the version with the VMS turbulence model) in [35]. The VMS components are from the RBVMS method given in [36–39]. The original DSD/SST formulation was named “DSD/SST-SUPS” in [35] (i.e. the version with the SUPG/PSPG stabilization), which was also called “ST-SUPS” in [17].

The SSTFSI technique was extended by the T★AFSM in [13] to arterial FSI modeling, with emphasis on arteries with aneurysm. The arterial geometries were approximations to patient-specific image-based geometries, mostly to those reported by Torii et al. A number of special techniques for arterial FSI were developed by the

T★AFSM in conjunction with the SSTFSI technique. These include techniques for calculating an estimated zero-pressure (EZP) arterial geometry [40, 41], a special mapping technique for specifying the velocity profile at an inflow boundary with non-circular shape [14], techniques for using variable arterial wall thickness [14, 41], mesh generation techniques for building layers of refined fluid mechanics mesh near the arterial walls [14, 41, 42], a recipe for pre-FSI computations that improve the convergence of the FSI computations [13], the Sequentially-Coupled Arterial FSI (SCAFSI) technique [43, 44] and its multiscale versions [44], and techniques [41] for the projection of fluid–structure interface stresses, calculation of the WSS and calculation of the oscillatory shear index (OSI). In FSI modeling of three cerebral artery segments with aneurysm reported by the T★AFSM in [45], the arterial geometries came from 3D rotational angiography (3DRA). In [45], the T★AFSM also addressed the computational challenges related to extraction of the arterial-lumen geometry from 3DRA, generation of a mesh for that geometry, and building a good starting point for the FSI computations. In addition to these computational challenges common to all three cases, the computational challenges encountered in some of these cases individually were addressed in [45]. In [46, 47], new techniques were presented for determining the shrinking amount in the EZP process, the arterial wall thickness, and the thickness of the layers of refined fluid mechanics volume mesh near the arterial walls. These techniques were originally proposed in Remark 3 of [45], but the description was very brief. In [46, 47], also a new scaling technique was introduced for specifying a more realistic volumetric flow rate. In [9], a technique was proposed to use the Laplace's equation for specifying a variable vessel wall thickness, and in [8, 48] a prestressing technique was developed for blood vessels. The former addressed the challenge of how to specify spatially varying vessel wall thickness. It inspired the idea of using the Laplace's equation over the surface mesh covering the lumen to specify a variable vessel wall thickness [45–47]. The latter addressed the challenge presented by the fact that patient-specific blood vessel geometry data comes from a configuration that is not stress-free, the same fact that motivated earlier the development of methods for calculating an EZP arterial geometry [40]. Both techniques are quite general and, most importantly, are independent of the details of the patient-specific blood vessel geometry. Related to that, recently, methods for an estimated element-based zero-stress state for the artery were introduced in [21].

While modeling the FSI between the blood flow and arterial walls is one of the most challenging problems in cardiovascular fluid mechanics, there are other complex problems that are comparably challenging. Patient-specific computation of unsteady blood flow in an artery with aneurysm and stent is one of them. Special methods targeting that class of cardiovascular fluid mechanics problems were introduced in [49], and a large set of computations were reported in [18, 49].

Thrombus formation (i.e., blood clotting) is a major problem in ventricular assist devices (VADs), especially in pulsatile VADs (PVADs). Long residence times and areas of recirculation or stagnation may lead to increased risk of thrombosis in PVADs [19, 50]. A method was presented in [51] for computing the particle residence time, which is known to correlate with an increased risk of thrombogenesis.

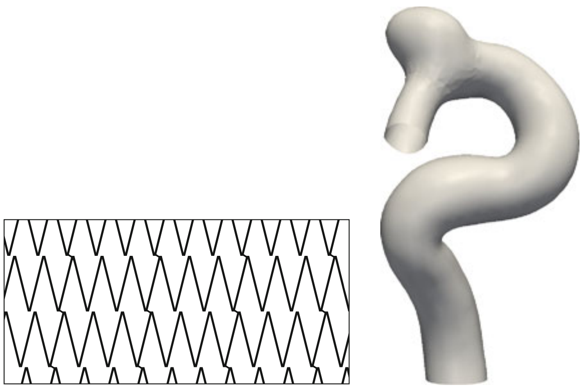


Fig. 1 Flat-stent geometry (*left*) and arterial lumen geometry (*right*)

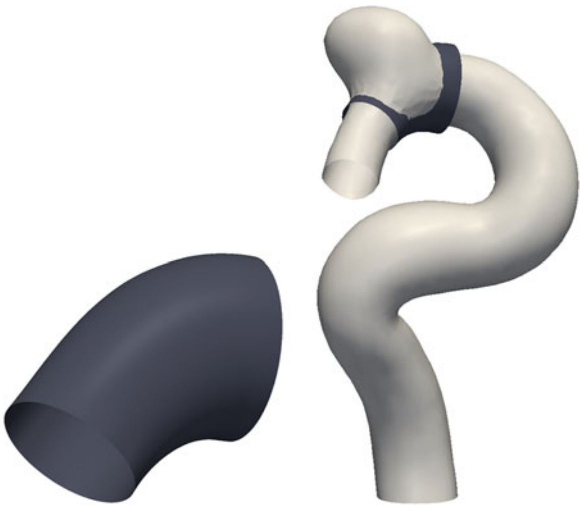


Fig. 2 Deformed stent (*left*) and split lumen geometry with the stent (*right*)

The method was developed in an ALE [17, 23] framework (the Eulerian case was investigated in [52]), and is suitable for flows with moving boundaries and interfaces, including FSI. In [53], the recently-developed residence time formulation was employed in the definition of the objective function for the FSI-based shape optimization study of a current PVAD design.

In this book chapter we provide an overview of how patient-specific cardiovascular fluid mechanics analysis, including FSI, can be very effectively carried out with the core and special ST and ALE technique. For the governing equations and the finite element formulations, including the ALE-VMS, DSD/SST and SSTFSI techniques, we refer the interested reader to [17, 46, 54]. Special techniques for stent

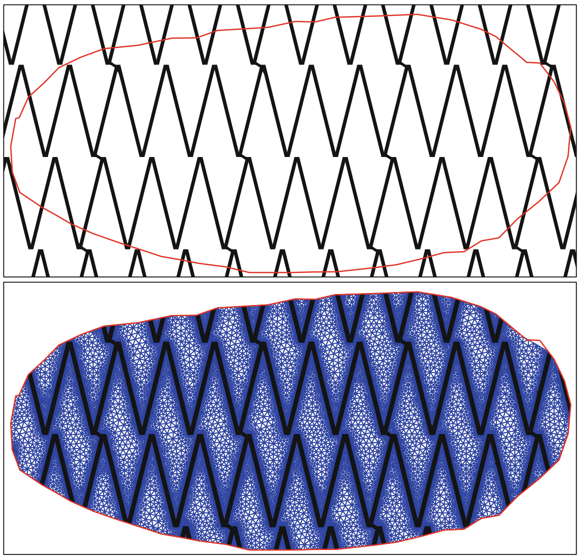


Fig. 3 Flat stent with the periphery of the interior-boundary geometry (*top*) stent mesh (*bottom*)

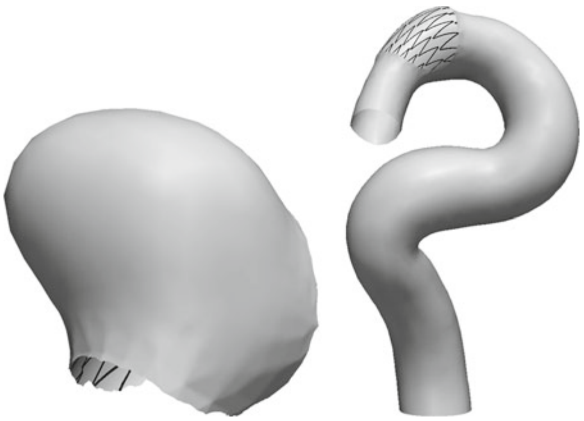


Fig. 4 Aneurysm (*left*) and parent (*right*) artery segments, separated by the stent

modeling and mesh generation and particle residence time calculation are described in Sects. 2 and 3. In Sect. 4, we give references for some of the other special techniques. The fluid (blood) and structure (blood vessel wall) properties and boundary conditions are given in Sect. 5. We present the computations in Sects. 6 and 7, and our concluding remarks in Sect. 8.

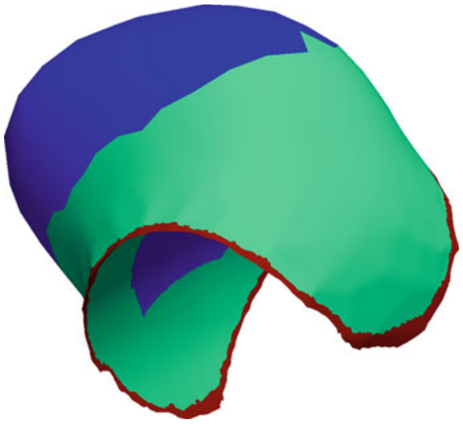


Fig. 5 Aneurysm artery segment showing regions of different thickness for the layers of refined mesh

2 Stent Modeling and Mesh Generation

This section is from [49]. Mesh generation of the cerebral artery with aneurysm and stent requires numerous steps that include taking the flat-stent design and lumen geometry and generating a fluid volume mesh representative of a stented artery with aneurysm. We begin by mapping the flat stent to the deformed stent, which fits across the neck of the aneurysm. The artery is separated into two segments, parent and aneurysm. Layers of refined mesh are generated at the stent and arterial walls in both segments. After the remaining volume mesh in each segment is generated, the two segments are merged on the interior-boundary mesh containing the stent.

1. Prepare the lumen geometry and flat-stent model as shown in Fig. 1. We extract the arterial surface geometry from medical images and generate a lumen geometry reflective of the inflated arterial-wall structure through the process reported in [45]. The flat-stent model was generated using the geometry of a Cordis Precise Pro Rx nitinol self-expanding stent (PC0630RXC) with a wire diameter of about 0.1 mm.
2. Generate a NURBS surface slightly larger than the artery such that the surface intersects the lumen geometry as shown in Fig. 2. We swept a NURBS surface following the curvature of the parent artery and extending slightly beyond the aneurysm neck. To simplify the mesh generation process, we only model the portion of the stent crossing the neck of the aneurysm. The intersection of the NURBS surface and lumen geometry is the periphery of the interior boundary containing the stent.
3. Map the periphery of the interior boundary, described above, to the flat stent and mesh that as shown in Fig. 3. We generate a triangular surface mesh using ANSYS ICEM CFD meshing software ("ICEM") and the geometry defined by the flat stent and interior-boundary periphery. The maximum element size specified in ICEM mesh generation leads to the width of the stent wire being

meshed with 3–4 elements. This ensures sufficient refinement to resolve the flow on the stent. The flat-stent mesh is then mapped from the flat NURBS surface to the deformed NURBS surface to form the interior-boundary mesh positioned across the neck of the aneurysm.

4. Use the periphery of the interior-boundary mesh as a predefined set of element edges, splitting the lumen geometry into parent and aneurysm segments as shown in Fig. 4. This reduces complexity in mesh generation. We use ICEM to generate the triangular surface meshes on the parent and aneurysm segments.
5. Using the surface meshes for the parent and aneurysm segments, we generate layers of refined mesh on either side of the stent and near the arterial walls. We use the process reported in [45] to generate the layers in the parent segment. In generating the layers in the aneurysm segment, we first start by separating the surface mesh into different regions as shown in Fig. 5. Due to the sharp angle of the geometry, no layers are explicitly generated in the red region. We specify a uniform thickness for the layers of refined mesh in the blue regions. The thickness of the first layer is approximately equal to the first layer of refined mesh in the parent segment. There are a total of four layers, each increasing in thickness using a progression ratio of 1.75 (the same number of layers and progression ratio used in [45]). To prevent elements tangling, the Laplace's equation is solved over the green region of the surface mesh to determine the thickness growth from essentially zero at the boundary with the red region to the desired layer thickness at the blue region boundary. We generate each of the four layers in the aneurysm segment separately and merge the layers (see Remark 2).
6. The rest of the fluid volume mesh is generated using ICEM. The innermost surface of the layers of refined mesh is extracted from the volume mesh and used as the surface mesh for generating the volume mesh in both the parent and aneurysm segments. The inner volume mesh is then merged to the refined layers.
7. The parent and aneurysm fluid volume mesh segments are merged on the interior-boundary mesh containing the stent. For the no-stent cases, all nodes are merged on that interior-boundary mesh. For the single- and double-stent cases, the nodes on the stent portion of the interior-boundary mesh are not merged and instead colocated.

Remark 1 We generate the double stent by overlaying two single flat-stent geometries and translating one of them in two directions. We map the intersection of the deformed NURBS surface and lumen geometry, which is again the periphery of the interior boundary, to the flat double-stent geometry and mesh the double stent as one mesh. The double-stent mesh is treated the same as the single-stent mesh in the remaining mesh generation steps. Figure 6 shows the full single and double stents.

Remark 2 The mesh generation process for the layers of refined mesh in the aneurysm segment presents challenges regarding tangled elements. With the mesh refinement required by the problem, building the layers into the artery has the potential to create elements with negative Jacobians. Each layer must be checked for the Jacobian values before generating the next layer.

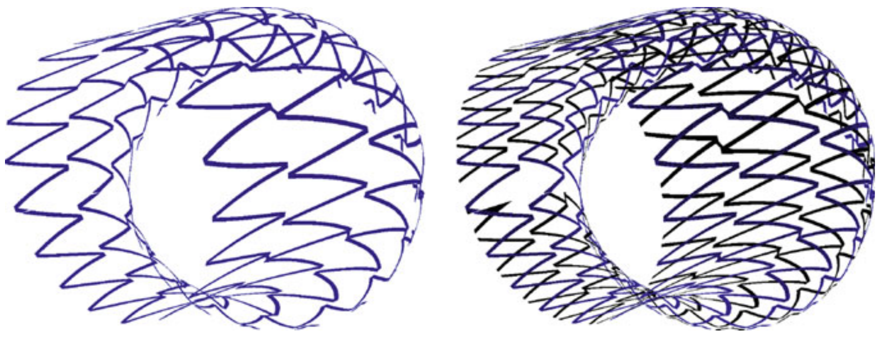


Fig. 6 Surface for single (*left*) and double (*right*) stents

3 Particle Residence Time Formulation

Consider the spatial domain Ω, and the subdomain $V \subset \Omega$. It may be of interest to know how long the material particles moving inside and through the domain Ω are "residing" in the subdomain V. (Hence, the term, "residence time".) We can formulate this as an initial-value problem:

$$\left.\frac{\partial \tau}{\partial t}\right|_{\mathbf{X}} = H(\mathbf{x}), \tag{1}$$

where

$$H(\mathbf{x}) = \begin{cases} 1 \text{ if } \mathbf{x} \in V \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

is the Heaviside function. The solution τ of Eq. (1) has the interpretation of the total time that a particle, occupying a spatial position $\mathbf{x}$ at time t, spent in the subdomain V. The source term given by the Heaviside function ensures that the time is accumulated only when the particle is inside the subdomain. In this framework, both Ω and V may be time-dependent, which is the case here.

For applications involving flows in stationary or moving domains it is convenient to re-write Eq. (1) with respect to the Eulerian (or spatial) frame as

$$\left.\frac{\partial \tau}{\partial t}\right|_{\mathbf{x}} + \mathbf{u} \cdot \nabla \tau = H(\mathbf{x}), \tag{3}$$

or the ALE frame as

$$\left.\frac{\partial \tau}{\partial t}\right|_{\hat{\mathbf{x}}} + (\mathbf{u} - \mathbf{v}) \cdot \nabla \tau = H(\mathbf{x}). \tag{4}$$

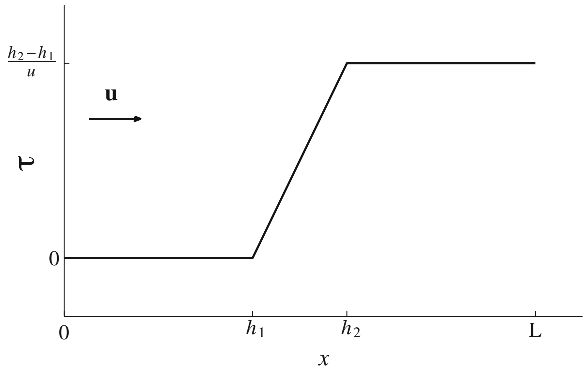


Fig. 7 Solution of the 1D model problem illustrating the particle residence time method

The following simple 1D example illustrates how the technique works. Let $\Omega = (0, L)$, $V = (h_1, h_2) \subset (0, L)$, and the flow velocity u is a positive constant (see Fig. 7). In this setting, Eq. (3) reduces to

$$\frac{\partial \tau}{\partial t} + u\frac{\partial \tau}{\partial x} = H(x). \tag{5}$$

Because the flow is from left to right and the region V is located downstream of the inlet, particles at the inlet could not have spent any time inside V. As a result, we set $\tau = 0$ at $x = 0$. Since this is a pure advection problem, boundary condition at the outlet is left unspecified. Equation (5) has a steady-state solution:

$$\tau(x) = \begin{cases} 0 & \text{if } x \in [0, h_1] \\ \frac{x-h_1}{u} & \text{if } x \in [h_1, h_2] \\ \frac{h_2-h_1}{u} & \text{if } x \in [h_2, L]. \end{cases} \tag{6}$$

Equation (6) implies that, once the transient response settles, prior to the interval of interest, the residence time is zero. As particles enter the interval of interest moving at constant speed the residence time is proportional to the distance from the leftmost edge of the interval, and inversely proportional to flow speed. When particles exit the interval of interest, the residence time stays constant. The analytical solution of the differential equation given by Eq. (6) is in complete agreement with the expected particle residence time for this simple case.

Remark 3 Besides illustrating how the method works, this example also shows that the proposed technique for calculating particle residence time is meaningful when the solution for τ reaches a steady state. The time-periodic solution for τ, which is often the case with cardiovascular applications, also presents a situation from which meaningful conclusions may be drawn.

Two scalar measures of residence time proposed in [51, 52] are:

$$RT_1 = \frac{1}{T|V|} \int_T \int_\Omega H(\mathbf{x}) \tau(\mathbf{x}, t) \, \mathrm{d}\Omega \, \mathrm{d}t, \tag{7}$$

where

$$|V| = \frac{1}{T} \int_T \int_\Omega H(\mathbf{x}) \, \mathrm{d}\Omega \, \mathrm{d}t \tag{8}$$

is the time-averaged volume of V, and

$$RT_2 = \frac{1}{T|Q|} \int_T \int_{\partial V} \tau \, (\mathbf{u} - \mathbf{v}) \cdot \mathbf{n} \, \mathrm{d}\partial V \, \mathrm{d}t, \tag{9}$$

where

$$|Q| = \frac{1}{T} \int_T \int_{\partial V^+} (\mathbf{u} - \mathbf{v}) \cdot \mathbf{n} \, \mathrm{d}\partial V \, \mathrm{d}t \tag{10}$$

is the time-averaged flow rate through the outflow boundary of V, denoted by ∂V. In Eq. (9), the expression for RT_2 may be simplified further as (see [51])

$$RT_2 = \frac{|V|}{|Q|}. \tag{11}$$

The most attractive feature of this measure of residence time is that it requires no information about τ.

Remark 4 Applied to the 1D model problem with solution given by Eq. (6) the two residence time measures produce $RT_1 = \frac{h_2 - h_1}{2u}$ and $RT_2 = \frac{h_2 - h_1}{u}$, which are the average and maximum particle residence time in V, respectively. For such a simple flow situation it is clear that the maximum residence time occurs at the outflow. However, in multiple dimensions, in the presence of complex, recirculating flow this may not be the case. In the case of blood pumps, one can in fact assess the device efficiency (meaning throughput efficiency) by comparing the residence time at the outlet with that in the interior of the domain. Residence time maxima occurring in the interior suggest the presence of persistent flow recirculation or stagnation zones that tend to "trap" the material and thus lower the pump efficiency.

An alternative approach to computing residence time is based on the dye injection technique. We use an advection equation as before, set the source term to zero, and apply a Dirichlet boundary condition on the scalar field τ at the inlet such that:

$$\tau = \begin{cases} 1 & \text{if } n = 1 \\ 0 & \text{otherwise} \end{cases}, \tag{12}$$

where n is the cycle number. (Note that τ is no longer particle residence time, but rather dye concentration.) This has the effect of "injecting" a dye for one cycle, which can be visualized as it moves through the domain. The volume of dye in the domain at any given time can be written as $\int_V \tau \, dV$. The dye is then ejected over subsequent cycles, and remaining dye volume is computed at the end of each cycle. The cycles are repeated until at least 95 % of the dye is removed from the system.

4 Other Special Tecniques

For interested reader, in this section we provide references to articles where special techniques related to different components of arterial-geometry construction and mesh generation can be found. These components include (a) arterial-surface extraction from medical images [45]; (b) arterial-wall-thickness construction based on "patches" [46] and based on solution of the Laplace's equation over the arterial volume [9] or lumen [46]; (c) fluid mechanics volume mesh generation, including layers of refined mesh near the arterial walls [46]; (d) calculating the EZP arterial geometry [46]; (e) calculating the blood vessel tissue prestress [8, 48]; and (f) calculating an estimated element-based zero-stress state for the artery [21]. We also provide references to articles where some additional special techniques can be found. These special techniques include (g) a mapping technique for inflow boundaries [14]; (h) boundary condition techniques for inclined inflow and outflow planes [45]; (i) a scaling technique for specifying realistic inflow rates [47]; (j) techniques [41] for the projection of fluid–structure interface stresses; (k) a recipe for pre-FSI computations that improve the convergence of the FSI computations [13]; (l) the SCAFSI technique [55]; (m) methods for WSS and OSI calculations [41]; and (n) a preconditioning technique [42].

5 Fluid and Structure Properties and Boundary Conditions

5.1 Fluid and Structure Properties

As it was done for the computations reported in [5], the blood is assumed to behave like a Newtonian fluid. The density and kinematic viscosity are set to 1000 kg/m^3 and 4.0×10^{-6} m^2/s. The material density of the arterial wall is known to be close to that of the blood and therefore set to 1000 kg/m^3. The arterial wall is modeled with the continuum element made of hyperelastic (Fung) material. The Fung material constants D_1 and D_2 (from [56]) are 2.6447×10^3 N/m^2 and 8.365, and the penalty Poisson's ratio is 0.45. Cerebral arteries are surrounded by cerebrospinal fluid, and we expect that to have a damping effect on the arterial-wall dynamics. Therefore we add a mass-proportional damping, which also helps in removing the high-frequency modes of the structural deformation. The damping coefficient η is chosen in such a way that the structural mechanics computations remain stable at the time-step size used. It is 1.5×10^4 s^{-1}.

5.2 Boundary Conditions

On the arterial walls, we specify no-slip boundary conditions for the flow. In the structural mechanics part, as boundary condition at the ends of the arteries, we set the normal component of the displacement to zero, and for one of those nodes we also set to zero the tangential displacement component that needs to be specified to preclude rigid-body motion. At the inflow boundary, we specify the velocity profile as a function of time, using the technique introduced in [14]. We use two types of conditions at the outflow boundaries. In the explicit version, at all outflow boundaries of an artery segment we specify the same traction boundary condition. The traction boundary condition is based on a pressure profile computed as described in [14]. In the implicit version, we consider a class of outflow boundary conditions in which the outlet traction is a function of the flow rate there (see [57] for details).

Remark 5 In the current T★AFSM computations, the volumetric flow rate at the inflow (calculated based on a velocity waveform representing the cross-sectional maximum velocity) is scaled by a factor. The factor is determined in such a way that the scaled flow rate, when averaged over the cardiac cycle, yields a target WSS for Poiseuille flow over an equivalent cross-sectional area. The target WSS is 10 dyn/cm^2 in the current computations. This technique was introduced in [46, 47].

6 Computations with the ST Methods

All computations were carried out in a parallel computing environment. In FSI modeling, the fully-discretized, coupled fluid and structural mechanics and mesh-moving equations are solved with the quasi-direct coupling technique (see Sect. 5.2 in [4]), and the computations were completed without any remeshing. In solving the linear equation systems involved at every nonlinear iteration, the GMRES search technique [58] is used with a diagonal preconditioner.

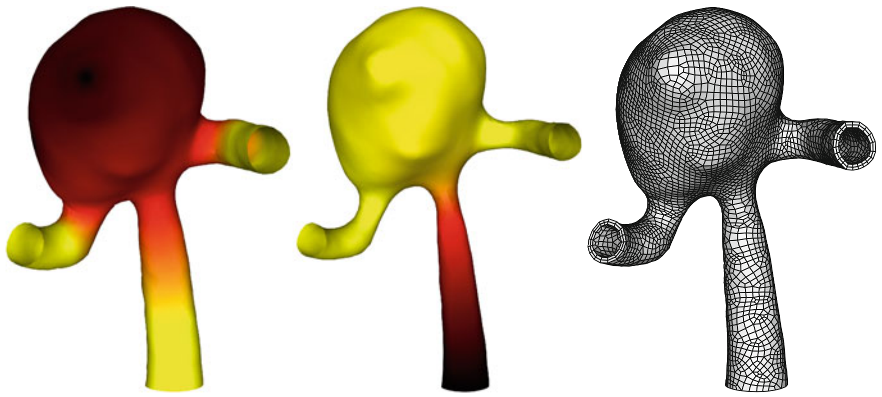


Fig. 8 Model-M6Acom. EZP shrinking amount over the surface (lumen) extracted from the medical image (*left*), wall thickness over the shrunk lumen (*middle*), and structure mesh at zero pressure (*right*). The *color range* represents a value range that increases from *light* to *dark*

6.1 FSI Modeling of a Cerebral Artery with Aneurysm

A sample was presented in [46] from a wide set of patient-specific cerebral-aneurysm models computed recently [47], where the shrinking amount in the EZP process, the arterial wall thickness, and the thickness of the layers of refined fluid mechanics mesh are determined based on the solution of the Laplace's equation over the surface mesh covering the lumen (see Sect. 4). We present that sample also here. The length scales used in conjunction with the trial ratios for the inflow and outflow boundaries are the lumen diameters at those ends. The value specified for the thickness of the first layer of elements at the inflow and outflow boundaries is 0.007 × (lumen diameter at those ends). In these computations, the volumetric flow rate is specified by using the scaling technique described in Remark 5. Figure 8 shows the EZP shrinking amount, wall thickness, and structure mesh for the arterial model, which we call Model-M6Acom. The diameter of the arterial lumen is 3.13 mm at the inflow end, and 2.12 mm at both outflow ends. The hexahedral structure mesh has two layers of elements across the arterial wall. For the layers of refined fluid mechanics mesh near the arterial wall, the progression factor is 1.75. Figure 9 shows the tetrahedral fluid mesh at the lumen, thickness of the first layer of elements near the arterial wall, and the mesh at the inflow plane. The structure mesh has 17,574 nodes and 11,650 elements, with 5,858 nodes and 5,825 element faces at the interface. The fluid mesh has 33,040 nodes and 192,112 elements, with 3,528 nodes and 6,996 element faces at the interface. The Womersley parameter is 1.96 and the peak volumetric flow rate is 1.2 ml/s. This is based on the duration of one cardiac cycle (1 s) and the representative diameter is calculated from the inflow area corresponding to the shape when inflated to the average pressure.

The computations are carried out with the SSTFSI-TIP1 technique (see [17, 46, 54] for details of the computational method). The (full) "SSP" option is used (see

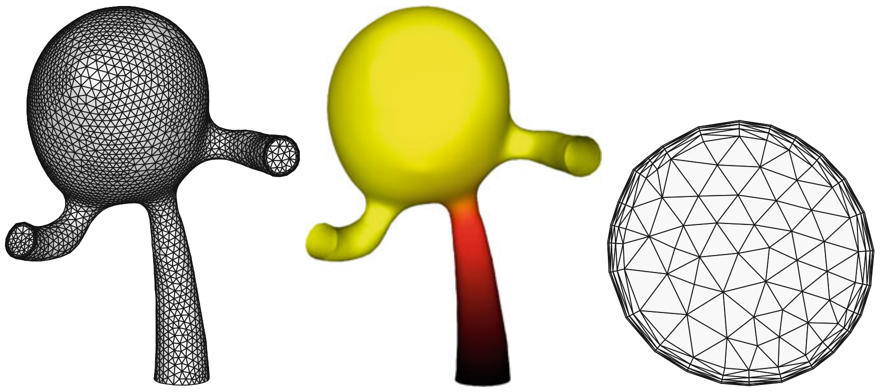


Fig. 9 Model-M6Acom. Fluid mechanics mesh at the lumen and outflow planes (*left*), thickness of the first layer of elements near the arterial wall (*middle*), and the mesh at the inflow plane (*right*). All pictures are from the starting point of our computation cycle. The *color range* represents a value range that increases from *light* to *dark*

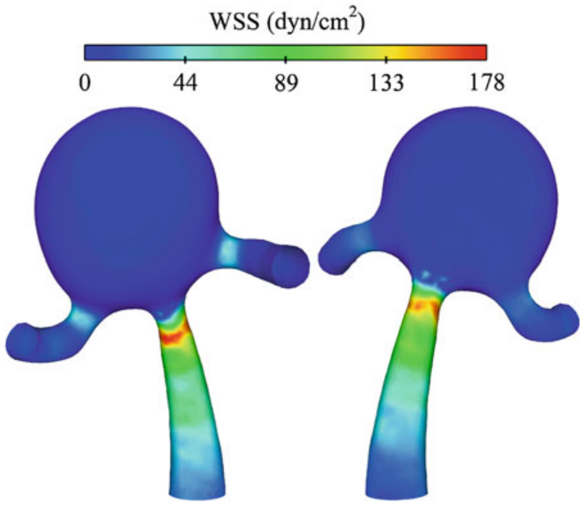


Fig. 10 Model-M6Acom. WSS when the volumetric flow rate is maximum

Remarks 21 and 22 in [46]). The time-step size is 3.333×10^{-3} s. The number of nonlinear iterations per time step is 6. The number of GMRES iterations per nonlinear iteration for the fluid + structure block was chosen such that mass balance is satisfied to within at most 5 % for each case. The number of GMRES iterations is 300, and this was sufficient for obtaining good mass balance. For all six nonlinear iterations the fluid scale is 1.0 and the structure scale is 100. For the mesh moving block the number of GMRES iterations is 30. Figure 10 shows the WSS when the volumetric

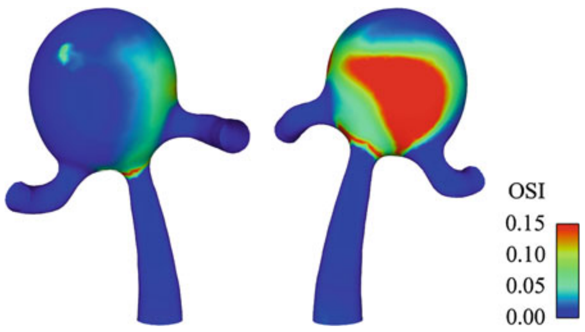


Fig. 11 Model-M6Acom. OSI

flow rate is maximum. Figure 11 shows the OSI, calculated with the technique that excludes rigid-body rotations from the calculation (see [17, 41, 46, 54]).

6.2 *Fluid Mechanics Modeling of a Cerebral Artery with Aneurysm and Stent*

This subsection is from [49]. Endovascular stent placement across the neck of an intracranial aneurysm can lead to aneurysm occlusion and thrombosis. We compare the flow field of arterial geometries before and after virtual "stenting" to assess the changes. Select aneurysms require treatment using two or more stents to sufficiently alter the flow field allowing for thrombosis. The test computations include a before-stenting case and after-stenting cases for both single- and double-stent treatments to compare the effectiveness of stenting with multiple stents. Section 6.2.1 details the parameters for the arterial geometry used in the computations. In Sect. 6.2.2 we compare hemodynamic values before and after stenting.

6.2.1 Computational Model

A patient-specific cerebral artery with aneurysm is studied at three states: before stenting, after stenting with a single stent, and after stenting with two stents. The inlet and outlet diameters, peak volumetric flow rate, and the Womersley number are 3.7 mm, 2.9 mm, 2.05 ml/s, and 2.33, respectively. The lumen geometry and the fluid mechanics mesh for the single-stent case are shown in Fig. 12. The cross-section view shows the refined mesh at the aneurysm neck on either side of the boundary separating the aneurysm from the parent artery. The number of nodes for the no-stent, single-stent and double-stent meshes are 527,323, 566,049 and 662,431, and the number of elements are 3,168,305, 3,300,182 and 3,736,603. All computations presented in Sect. 6.2.2 are for zero-thickness representation of the stent.

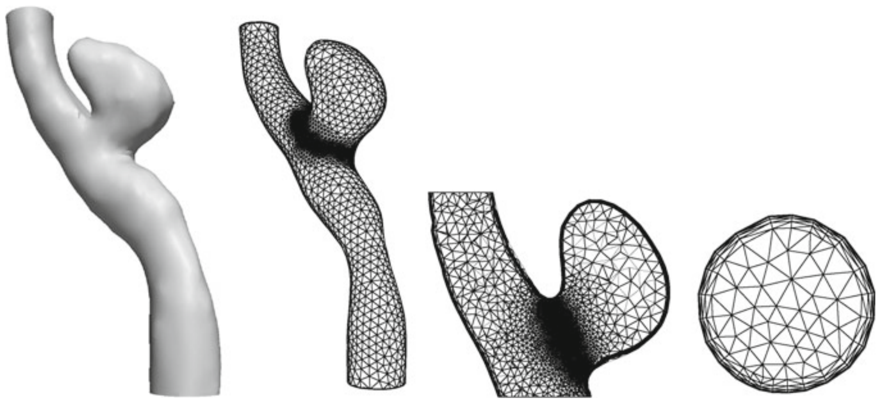


Fig. 12 Arterial lumen geometry obtained from voxel data (*left*) and the fluid mechanics mesh for the single-stent case, with cross-section and inflow plane views

The ST-VMS method is used, with the stabilization parameters as given by Eq. (7) in [4] for τ_{M} $(= \tau_{\mathrm{SUPS}}) = \tau_{\mathrm{SUPG}}$ and Eq. (37) in [59] for ν_{C} $(=\nu_{\mathrm{LSIC}}) = \nu_{\mathrm{HRGN}}$. The time step size is 3.333×10^{-3} s. The number of nonlinear iterations per time step is 4, and the number of GMRES iterations per nonlinear iteration for the no-stent, single-stent and double-stent cases is 1,000, 1,500 and 1,500.

6.2.2 Comparative Study

Inducing thrombosis in an aneurysm requires altering the hemodynamics at the aneurysm. Inserting a stent changes the pattern and amount of blood flow from the parent artery to the aneurysm, influencing stasis within the aneurysm. The stent free area at the neck of the aneurysm is reduced to approximately 85 and 71 % in the single- and double-stent cases. We compare the fluid mechanics before and after stenting by analyzing the ratio of the aneurysm-inflow rate to the time-averaged parent-artery-inflow rate $\frac{Q_{\mathrm{A}}}{\overline{Q}_{\mathrm{P}}}$, the spatially averaged kinetic energy and vorticity in the aneurysm, and OSI. The aneurysm-inflow rate is calculated by integrating the magnitude of the normal component of the velocity over the interior-boundary mesh containing the stent and dividing that by 2. We divide by 2, because the integral of the magnitude of the normal component of the velocity measures twice the inflow rate. The effectiveness of stenting using either the single or double stent depends on the degree to which the flow characteristics were altered and also the arterial geometry and size of the aneurysm. The higher OSI observed in stent cases follows the belief that regions with increased OSI prompt thrombus formation [60, 61].

The aneurysm has a volume of 0.10 cm^3 and approximate neck area of 0.47 cm^2. The total area in the neck blocked by the stent in the single- and double-stent cases is 0.07 and 0.13 cm^2. Figure 13 shows the aneurysm velocity magnitude at peak flow into the aneurysm. The parent artery has an average inflow rate of 0.62 ml/s. The

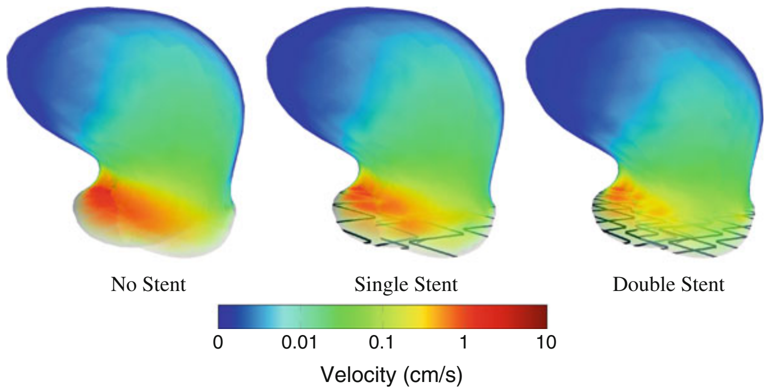


Fig. 13 Aneurysm velocity magnitude at peak flow into the aneurysm

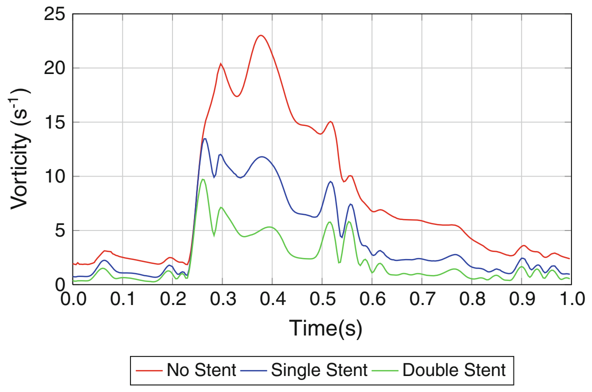


Fig. 14 Comparison of spatially averaged vorticity magnitude in the aneurysm

peak blood flow into and within the aneurysm occurs approximately 0.02 s before peak inflow rate in the parent artery. The time-averaged $\frac{Q_A}{Q_P}$ decreases by 22 and 78 % in the single- and double-stent cases. Similarly, the kinetic energy averaged in space and time decreases by 72 % in the single-stent case and 92 % in the double-stent case. The reduction in vorticity in the aneurysm caused by stenting is shown in Figs. 14 and 15. The vorticity, averaged in space and time, is reduced by 47 and 72 % in the single- and double-stent cases.

7 Computations with the ALE-VMS FSI Method

In this section we present two examples of cardiovascular FSI computations, which include patient-specific cerebral aneurysms and the PVAD. The cerebral aneurysm results are taken from [48], while the PVAD computations are reported in [19, 51].

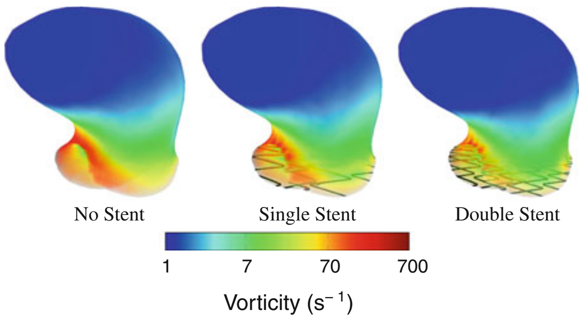


Fig. 15 Aneurysm vorticity magnitude at peak flow into the aneurysm

Computations were carried out in a parallel computing environment. The FSI equations are advanced in time using the generalized-α time integrator proposed in [62] for structural dynamics, and developed for fluid mechanics in [63] and FSI in [24].

A quasi-direct solution strategy is used, where the increments of the fluid and structural mechanics variables are obtained simultaneously [3, 4, 17, 64]. A Jacobian-based mesh stiffening technique [17, 32, 65, 66] is used to move the fluid mechanics mesh. The effect of the mesh motion on the fluid equations is omitted from the tangent matrix for efficiency, as advocated in [57] for cardiovascular FSI applications. In solving the linear equation systems involved at every nonlinear iteration, the GMRES search technique is used with a block-diagonal preconditioner.

7.1 Cerebral Aneurysms: Tissue Prestress

In this section we focus on the importance of prestressing of arterial tissue and its effect on the key quantities of interest in FSI computations. For other results obtained in ALE-VMS FSI simulations of cerebral aneurysms the reader is referred to [8, 67–69]. A Neo-Hookean material with dilatational penalty is employed here to model the response of the vascular wall. We apply the prestress procedure proposed in [48] to two cerebral aneurysm models shown in Fig. 16. The inflow and outflow branches are labeled as M1 and M2, respectively, in the same figure. Both models come from patient-specific imaging data and exhibit significant geometrical differences. Model 1 has a relatively small aneurysm dome and an inlet branch of large radius. The situation is reversed for Model 2. Meshing techniques developed by Zhang et al. [68] are used for generating linear tetrahedral elements for both models. The meshes contain both the blood flow volume and solid vessel wall. Fine meshes with boundary layer resolution are employed.

Figure 17 shows the final prestressed state for both models, which also demonstrates the applicability of the method to different vascular geometries. The models

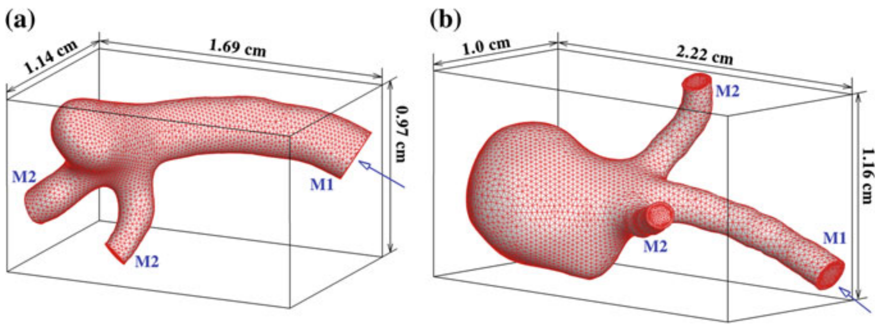


Fig. 16 Tetrahedral finite element mesh of the middle cerebral artery (MCA) bifurcation with aneurysm. Both patient-specific models are discretized using approximately 165K elements and 30K nodes. Inlet branches are labeled M1 and outlet branches are labeled M2 for both models. The *arrows* point in the direction of inflow velocity. The inlet cross-sectional areas Models 1 and 2 are 4.962×10^{-2} and 2.102×10^{-2} cm^2, respectively. **a** Model 1; **b** Model 2

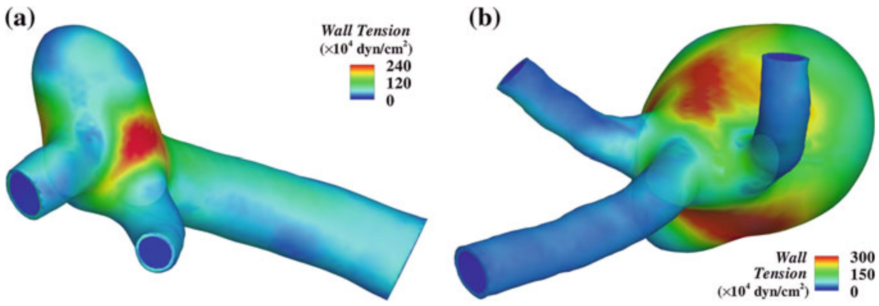


Fig. 17 Final prestressed state for Model 1 and 2. The models are colored by the isocontours of wall tension, which is defined as the absolute value of the first principal in-plane stress of $\mathbf{S}_0$. **a** Model 1; **b** Model 2

are colored by the isocontours of wall tension, which is defined as the absolute value of the first principal in-plane stress.

To assess the influence of the prestress, we perform a coupled FSI simulation of both models and compare the results with and without prestress. Figure 18 shows the relative wall displacement between the deformed configuration and reference configuration coming from imaging data. The deformed configuration corresponds to the time instant when the fluid traction vector is closest to the averaged traction vector used for the prestress problem. Almost no difference between the reference and deformed configurations is seen in the case of the prestressed-artery simulation, as expected. However, in the case of non-prestressed simulation, the differences between the two configurations are significant. This indicates that the FSI problem is not being solved on the correct geometry. Furthermore, the relative geometry error is larger for Model 2, which has a larger aneurysm dome and a thinner wall. Figure 19 shows the relative wall displacement between the deformed configuration

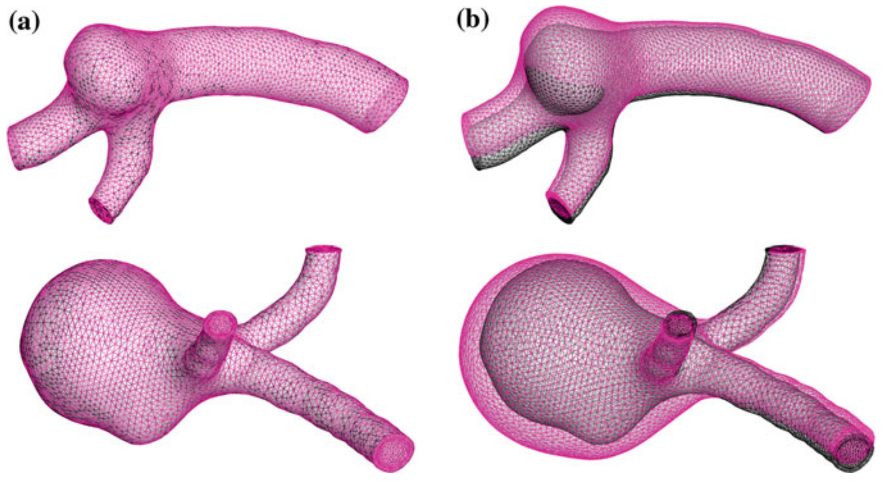


Fig. 18 Relative wall displacement between the deformed configuration and reference configuration coming from imaging data. *Top* Model 1; *Bottom* Model 2. The deformed configuration corresponds to the time instant when the fluid traction vector is closest to the averaged traction vector used for the prestress computation. **a** With prestress; **b** Without prestress

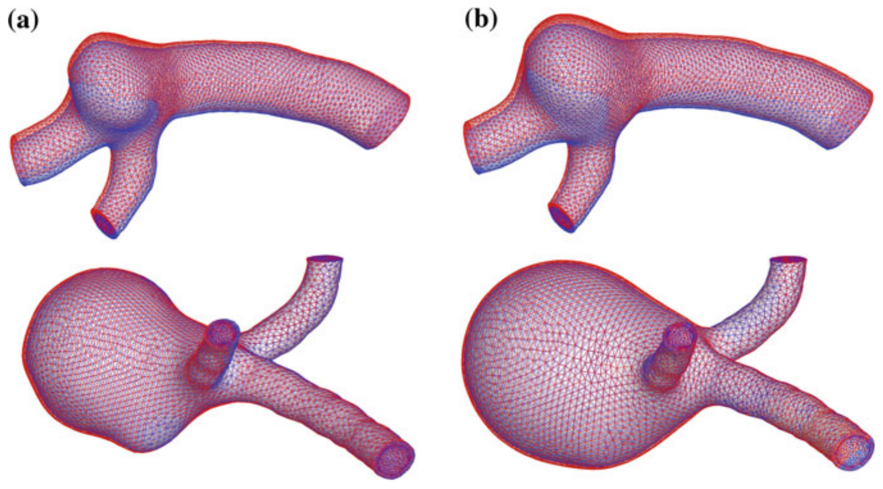


Fig. 19 Relative wall displacement between the deformed configuration at peak systole and low diastole. *Top* Model 1; *Bottom* Model 2. **a** With prestress; **b** Without prestress

at peak systole and low diastole. In both the prestressed and non-prestressed cases the relative displacement is fairly small, yet non-negligible. The non-prestressed case, however, makes use of the geometry that is significantly more "inflated" compared to the prestressed case and the imaging data.

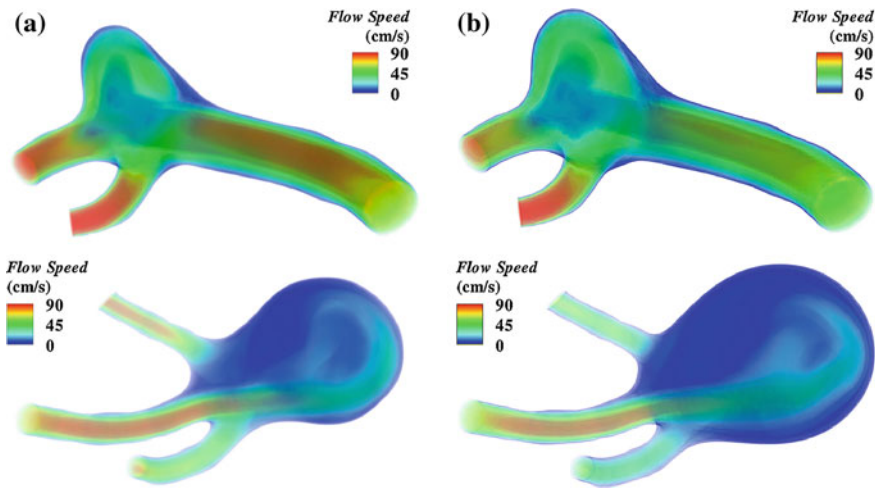


Fig. 20 Volume-rendered blood flow velocity magnitude near peak systole. *Top* Model 1; *Bottom* Model 2. **a** With prestress; **b** Without prestress

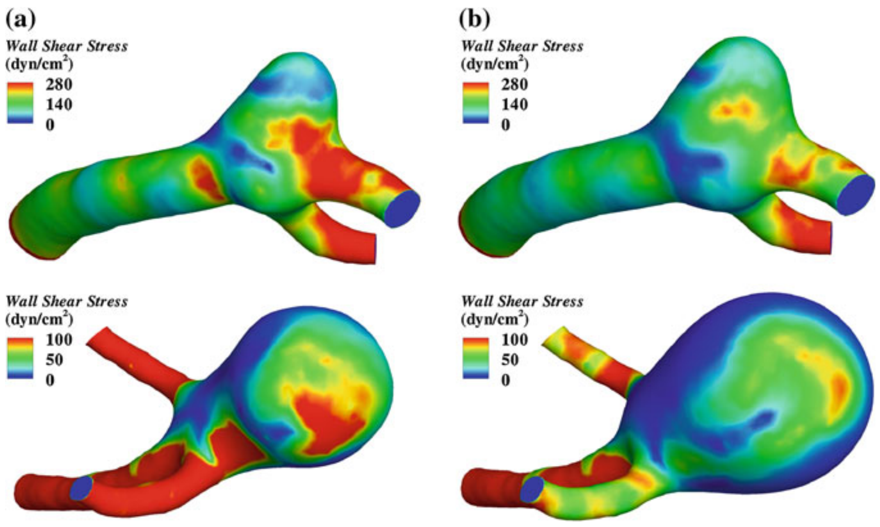

(a)
Wall Shear Stress (dyn/cm^2)
280
140
0
(b)
Wall Shear Stress (dyn/cm^2)
280
140
0
Wall Shear Stress (dyn/cm^2)
100
50
0
Wall Shear Stress (dyn/cm^2)
100
50
0

Fig. 21 Wall shear stress near peak systole. *Top* Model 1; *Bottom* Model 2. **a** With prestress; **b** Without prestress

Figure 20 shows a comparison of the blood flow speed near peak systole for the simulations with and without prestress. The results between the two cases are very similar, although some differences in the flow structures are visible, especially for Model 2. Figure 21 shows a comparison of the WSS near peak systole for both cases. The WSS, unlike blood flow velocity, exhibits significant differences in magnitude

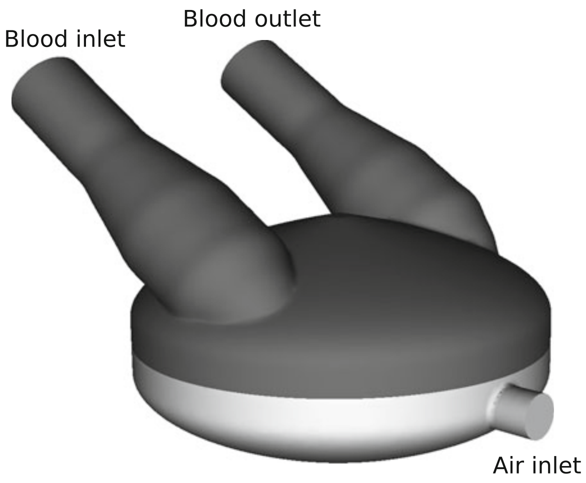


Fig. 22 PVAD geometry including blood and air chambers as well as inlet and outlet boundaries. *Darker* and *lighter shades* are used to denote the blood and air chambers, respectively

and spatial distribution. The simulations presented clearly show the importance of tissue prestress in patient-specific vascular FSI modeling for accurate prediction of hemodynamic phenomena and vessel wall mechanics.

7.2 PVAD: Residence Time Computations

The PVAD operates as follows. The pneumatically driven device is comprised of the blood and air chambers and a thin membrane separating the two subdomains (see Fig. 22). The blood chamber is connected to the circulatory system. During the fill stage, the air is pumped out of the air chamber, which displaces the membrane downward and the blood chamber is filled with blood from one of the heart ventricles. During the ejection stage, air is pumped into the air chamber, which displaces the membrane upward and blood is ejected into the aorta. We simulate the process, exactly as described, using the FSI techniques summarized in what follows.

The computational FSI methodology for for PVADs was presented in detail in [19]. The linear FEM-based ALE-VMS technique [12, 24, 54, 70] is used for the computation of blood and air flow in the respective chambers of the device. The NURBS-based isogeometric Kirchhoff–Love shell formulation [71–77] is employed to model the structural mechanics behavior of the thin membrane separating the air and blood chambers. The St. Venant–Kirchhoff constitutive law is used to model the material response of the thin membrane. A nonmatching interface FSI formulation [4, 14, 17, 34, 35, 42, 44–46, 54, 78–80] is employed. In an effort to enhance computational efficiency, a combination of a sparse-matrix-based and a matrix-free approach was used in the implementation of the quasi-direct nonmatching-interface FSI coupling technique (see [19] for details). Due to the large deformations of the membrane, the blood and air domain meshes need to be periodically regenerated. Five-to-six remeshing steps typically occur during one cycle.

During the ejection stage the inlet is closed and the outlet is open to a resistance boundary condition [81]. During the fill stage, the outlet is closed and the inlet is open to the same resistance BCs. The latter setup presents a flow regime that is often numerically unstable due to the significant amount of reversed flow coming into the computational domain through the open boundary. The outflow stabilization technique developed in [24] and further studied in [82] is employed here, which renders this setup stable and enables simulating PVAD operation as described in the first paragraph of this section.

The discretization of the time-dependent advection equations describing particle residence time and dye injection makes use of the ALE-based, SUPG-stabilized [27] formulation augmented with a YZβ discontinuity-capturing operator [83, 84]. The advection equations are only solved in the blood chamber using the same meshes of linear tetrahedral elements as the fluid mechanics problem. Because the FSI equations do not depend on the residence-time solution, we first compute the FSI problem for three cycles to achieve a nearly-time-periodic solution. The fluid and mesh velocity, and mesh position solutions in the last cycle are used to drive the residence time computations.

For the FSI simulation, a stroke volume of 73 ml is chosen for this device, which yields an ejection fraction of 68 %. A beat frequency of 80 bpm is used, for a pump output of 5.8 l/min. Each pump cycle may be broken up into two components: the fill stage and the ejection stage. We impose the fill period of 0.45 s, and the ejection period of 0.3 s, and we also enforce that each stage must fill or eject the same volume, 73 ml. For simplicity, the flow is assumed to behave sinusoidally during each stage.

The number of elements in each domain fluctuates with remeshing. The device is initialized with 557,416 elements in the blood chamber and 264,597 elements in the air chamber. Each remeshing step respects the initial length scale of the elements. The membrane is discretized with 1,024 C^1-continuous quadratic NURBS elements. A time step size of 0.001 s is used. The FSI problem is computed for three time cycles in order to reach a time-periodic solution. The FSI solution in the last cycle is used to drive the residence time computations. See Fig. 23 for the snapshots of flow solution and membrane deformation at various times during the cycle.

In the residence time simulations the scalar field τ is initialized to zero. The subdomain V in Eq. (2) is assumed to be the whole blood chamber, including the inlet and outlet arms. Thus, τ is a measure of particle residence time in the whole blood chamber. At the inlet branch, τ is set to zero and at the outlet branch its value is left unspecified. Five cycles were required to reach a time-periodic solution for τ. Figure 23 shows the snapshots of residence time distribution in the blood chamber during the cycle. While a local maximum in the residence time occurs near the apex of the blood chamber, which is due to a large vortical flow structure in the device, the maximum residence time concentrates near the outlet branch for most of the cycle.

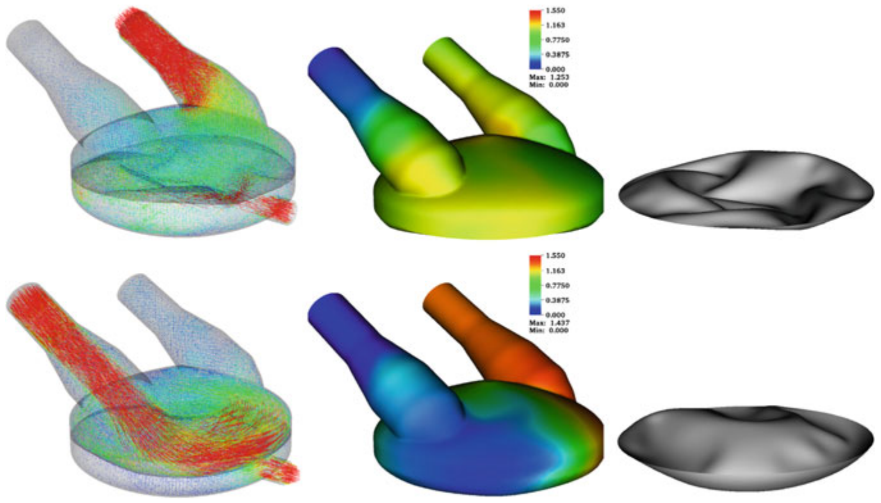


Fig. 23 PVAD flow solution (*left*), residence time (*middle*), and membrane deformation (*right*) at several instances during the cycle: *Top to bottom*, $t = 0.15$ s and $t = 0.525$ s

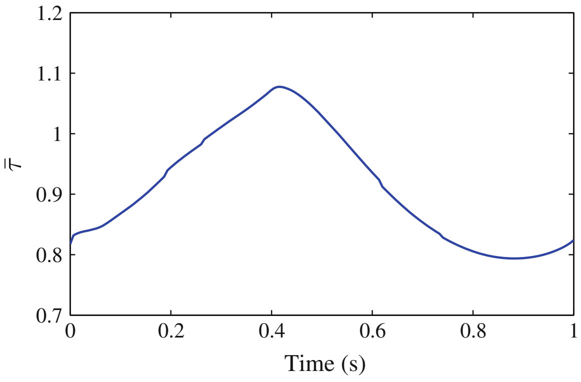


Fig. 24 Plot of $\bar{\tau}$ as a function of time. The fill stage is given by $t \in [0, 0.45]$, and the ejection stage as $t \in (0.45, 0.75]$

This suggests that the old material does not accumulate in the interior of the device, and is ejected by the pump in a fairly efficient manner.

The time history of $\bar{\tau}$, the spatial average of the blood chamber residence time, is shown in Fig. 24. The average residence time rises uniformly during the ejection stage. This is because no new material is entering the blood chamber and the average is expected to rise with time. The fill stage shows a brief and rapid decrease in $\bar{\tau}$, as new material with $\tau = 0$ enters the blood chamber. The trend again reverses and $\bar{\tau}$ begins to increase again in the later part of the fill stage as the influx of new material slows and the blood chamber volume grows. Using Eqs. (7) and (11), we find that

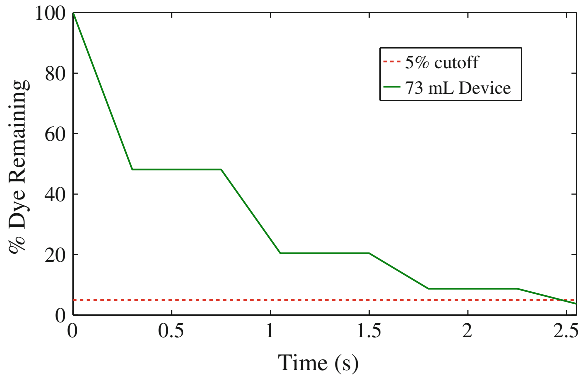


Fig. 25 Plot of the percentage of dye remaining in the blood chamber versus time

$RT_1 = 0.893$ s and $RT_2 = 1.031$ s for this device, meaning the blood particles remain in the chamber on average for about 1 s. Note that the difference between RT_1 and RT_2 is very minor, suggesting that the bulk of the residence time comes from the particles circulating the chamber rather than directly traversing the length of the device from the inlet to outlet.

Results from the dye injection analysis can be seen in Fig. 25. The figure shows a percentage of dye remaining in the blood chamber after the chamber is filled for one cycle. Note that in the first cycle over 50 % of the dye is removed. Furthermore, it takes 2.47 s to remove 95 % of the dye subsequent to the initial “fill cycle”.

8 Concluding Remarks

We presented a review of how cardiovascular fluid mechanics analysis, including FSI, can be very effectively carried out with the core and special ST and ALE techniques. We presented several challenging computations that were successfully carried with these techniques. The core techniques are the ALE-VMS, DSD/SST and SSTFSI methods. The special techniques reviewed were those for stent modeling and mesh generation and for calculation of the particle residence time. These were only a few examples of the large number of special techniques developed for cardiovascular fluid mechanics modeling in conjunction with the core techniques, ranging from arterial-surface extraction techniques to methods for an estimated element-based zero-stress state for the artery. We provided references for some of the other special techniques. This article shows that the core and special ST and ALE techniques developed can successfully address the computational challenges encountered in patient-specific cardiovascular fluid mechanics modeling.

References

1. Humphrey JD (2002) Cardiovascular solid mechanics. Springer, New York
2. Holzapfel GA (2000) Nonlinear solid mechanics, a continuum approach for engineering. Wiley, Chichester
3. Tezduyar TE, Sathe S, Keedy R, Stein K (2006) Space–time finite element techniques for computation of fluid–structure interactions. Comput Methods Appl Mech Eng 195:2002–2027. doi:10.1016/j.cma.2004.09.014
4. Tezduyar TE, Sathe S (2007) Modeling of fluid–structure interactions with the space–time finite elements: solution techniques. Int J Numer Meth Fluids 54:855–900. doi:10.1002/fld.1430
5. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2004) Influence of wall elasticity on image-based blood flow simulation. Jpn Soc Mech Eng J Ser A 70:1224–1231 (in Japanese)
6. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2006) Computer modeling of cardiovascular fluid–structure interactions with the deforming-spatial-domain/stabilized space–time formulation. Comput Meth Appl Mech Eng 195:1885–1895. doi:10.1016/j.cma.2005.05.050
7. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2007) Influence of wall elasticity in patient-specific hemodynamic simulations. Comput Fluids 36:160–168. doi:10.1016/j.compfluid.2005.07.014
8. Bazilevs Y, Hsu M-C, Zhang Y, Wang W, Kvamsdal T, Hentschel S, Isaksen J (2010) Computational fluid–structure interaction: methods and application to cerebral aneurysms. Biomech Model Mechanobiol 9:481–498
9. Bazilevs Y, Hsu M-C, Benson D, Sankaran S, Marsden A (2009) Computational fluid–structure interaction: methods and application to a total cavopulmonary connection. Comput Mech 45:77–89
10. Bazilevs Y, del Alamo JC, Humphrey JD (2010) From imaging to prediction: emerging noninvasive methods in pediatric cardiology. Prog Pediatr Cardiol 30:81–89
11. Figueroa CA, Baek S, Taylor CA, Humphrey JD (2009) A computational framework for fluid-solid-growth modeling in cardiovascular simulations. Comput Meth Appl Mech Eng 199:3583–3602
12. Bazilevs Y, Calo VM, Zhang Y, Hughes TJR (2006) Isogeometric fluid–structure interaction analysis with applications to arterial blood flow. Comput Mech 38:310–322
13. Tezduyar TE, Sathe S, Cragin T, Nanna B, Conklin BS, Pausewang J, Schwaab M (2007) Modeling of fluid–structure interactions with the space–time finite elements: arterial fluid mechanics. Int J Numer Meth Fluids 54:901–922. doi:10.1002/fld.1443
14. Takizawa K, Christopher J, Tezduyar TE, Sathe S (2010) Space–time finite element computation of arterial fluid–structure interactions with patient-specific data. Int J Numer Meth Biomed Eng 26:101–116. doi:10.1002/cnm.1241
15. Sugiyama K, Ii S, Takeuchi S, Takagi S, Matsumoto Y (2010) Full eulerian simulations of biconcave Neo-Hookean particles in a Poiseuille flow. Comput Mech 46:147–157
16. Manguoglu M, Takizawa K, Sameh AH, Tezduyar TE (2011) A parallel sparse algorithm targeting arterial fluid mechanics computations. Comput Mech 48:377–384. doi:10.1007/s00466-011-0619-0
17. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Computational fluid–structure interaction: methods and applications. Wiley, Hoboken
18. Takizawa K, Schjodt K, Puntel A, Kostov N, Tezduyar TE (2013) Patient-specific computational analysis of the influence of a stent on the unsteady flow in cerebral aneurysms. Comput Mech 51:1061–1073. doi:10.1007/s00466-012-0790-y
19. Long CC, Marsden AL, Bazilevs Y (2013) Fluid–structure interaction simulation of pulsatile ventricular assist devices. Comput Mech. doi:10.1007/s00466-013-0858-3 (published online)
20. Esmaily-Moghadam M, Bazilevs Y, Marsden AL (2013) A new preconditioning technique for implicitly coupled multidomain simulations with applications to hemodynamics. Comput Mech , doi:10.1007/s00466-013-0868-1 (published online)

21. Takizawa K, Takagi H, Tezduyar TE, Torii R (2013) Estimation of element-based zero-stress state for arterial FSI computations. Comput Mech. doi:10.1007/s00466-013-0919-7 (published online)
22. Takizawa K, Tezduyar TE, Buscher A, Asada S (2013) Space–time interface-tracking with topology change (ST-TC). Comput Mech. doi:10.1007/s00466-013-0935-7 (published online)
23. Hughes TJR, Liu WK, Zimmermann TK (1981) Lagrangian–Eulerian finite element formulation for incompressible viscous flows. Comput Meth Appl Mech Eng 29:329–349
24. Bazilevs Y, Calo VM, Hughes TJR, Zhang Y (2008) Isogeometric fluid–structure interaction: theory, algorithms, and computations. Comput Mech 43:3–37
25. Tezduyar TE (1992) Stabilized finite element formulations for incompressible flow computations. Adv Appl Mech 28:1–44. doi:10.1016/S0065-2156(08)70153-4
26. Tezduyar TE (2003) Computation of moving boundaries and interfaces and stabilization parameters. Int J Numer Meth Fluids 43:555–575. doi:10.1002/fld.505
27. Brooks AN, Hughes TJR (1982) Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. Comput Meth Appl Mech Eng 32:199–259
28. Tezduyar TE, Mittal S, Ray SE, Shih R (1992) Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements. Comput Meth Appl Mech Eng 95:221–242. doi:10.1016/0045-7825(92)90141-6
29. Hughes TJR, Franca LP, Balestra M (1986) A new finite element formulation for computational fluid dynamics: v. circumventing the Babuška–Brezzi condition: a stable Petrov–Galerkin formulation of the stokes problem accommodating equal-order interpolations. Comput Meth Appl Mech Eng 59:85–99
30. Hughes TJR, Hulbert GM (1988) Space–time finite element methods for elastodynamics: formulations and error estimates. Comput Meth Appl Mech Eng 66:339–363
31. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2011) Influencing factors in image-based fluid–structure interaction computation of cerebral aneurysms. Int J Numer Meth Fluids 65:324–340. doi:10.1002/fld.2448
32. Tezduyar T, Aliabadi S, Behr M, Johnson A, Mittal S (1993) Parallel finite-element computation of 3D flows. Computer 26:27–36. doi:10.1109/2.237441
33. Tezduyar TE (2004) Finite element methods for fluid dynamics with moving boundaries and interfaces. In: Stein E, Borst RD, Hughes TJR (eds) Encyclopedia of computational mechanics. Fluids, vol 3. Wiley, Hoboken (Chapter 17)
34. Takizawa K, Tezduyar TE (2012) Space–time fluid–structure interaction methods. Math Models Math Appl Sci 22:1230001. doi:10.1142/S0218202512300013
35. Takizawa K, Tezduyar TE (2011) Multiscale space–time fluid–structure interaction techniques. Comput Mech 48:247–267. doi:10.1007/s00466-011-0571-z
36. Hughes TJR (1995) Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles, and the origins of stabilized methods. Comput Meth Appl Mech Eng 127:387–401
37. Hughes TJR, Oberai AA, Mazzei L (2001) Large eddy simulation of turbulent channel flows by the variational multiscale method. Phys Fluids 13:1784–1799
38. Bazilevs Y, Calo VM, Cottrell JA, Hughes TJR, Reali A, Scovazzi G (2007) Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. Comput Meth Appl Mech Eng 197:173–201
39. Bazilevs Y, Akkerman I (2010) Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual-based variational multiscale method. J Comput Phys 229:3402–3414
40. Tezduyar TE, Cragin T, Sathe S, Nanna B (2007) FSI computations in arterial fluid mechanics with estimated zero-pressure arterial geometry. In: Onate E, Garcia J, Bergan P, Kvamsdal T (eds) Marine 2007. CIMNE, Barcelona, Spain
41. Takizawa K, Moorman C, Wright S, Christopher J, Tezduyar TE (2010) Wall shear stress calculations in space–time finite element computation of arterial fluid–structure interactions. Comput Mech 46:31–41. doi:10.1007/s00466-009-0425-0

42. Tezduyar TE, Schwaab M, Sathe S (2009) Sequentially-coupled arterial fluid–structure interaction (SCAFSI) technique. Comput Meth Appl Mech Eng 198:3524–3533. doi:10.1016/j.cma.2008.05.024
43. Tezduyar TE, Schwaab M, Sathe S (2007) Arterial fluid mechanics with the sequentially-coupled arterial FSI technique. In: Onate E, Papadrakakis M, Schrefler B (eds) Coupled Problems 2007. CIMNE, Barcelona, Spain
44. Tezduyar TE, Takizawa K, Moorman C, Wright S, Christopher J (2010) Multiscale sequentially-coupled arterial FSI technique. Comput Mec 46:17–29. doi:10.1007/s00466-009-0423-2
45. Takizawa K, Moorman C, Wright S, Purdue J, McPhail T, Chen PR, Warren J, Tezduyar TE (2011) Patient-specific arterial fluid–structure interaction modeling of cerebral aneurysms. Int J Numer Meth Fluids 65:308–323. doi:10.1002/fld.2360
46. Tezduyar TE, Takizawa K, Brummer T, Chen PR (2011) Space–time fluid–structure interaction modeling of patient-specific cerebral aneurysms. Int J Numer Meth Biomed Eng 27:1665–1710. doi:10.1002/cnm.1433
47. Takizawa K, Brummer T, Tezduyar TE, Chen PR (2012) A comparative study based on patient-specific fluid–structure interaction modeling of cerebral aneurysms. J Appl Mech 79:010908. doi:10.1115/1.4005071
48. Hsu M-C, Bazilevs Y (2011) Blood vessel tissue prestress modeling for vascular fluid–structure interaction simulations. Finite Elem Anal Des 47:593–599
49. Takizawa K, Schjodt K, Puntel A, Kostov N, Tezduyar TE (2012) Patient-specific computer modeling of blood flow in cerebral arteries with aneurysm and stent. Comput Mech 50:675–686. doi:10.1007/s00466-012-0760-4
50. Bluestein D, Niu L, Schoephoerster R, Dewanjee M (1997) Fluid mechanics of arterial stenosis: relationship to the development of mural thrombus. Ann Biomed Eng 25:344–356
51. Long CC, Esmaily-Moghadam M, Marsden AL, Bazilevs Y (2013) Computation of residence time in the simulation of pulsatile ventricular assist devices. Comput Mech. doi:10.1007/s00466-013-0931-y (published online)
52. Esmaily-Moghadam M, Hsia T-Y, Marsden A (2014) A non-discrete method for computation of residence time in fluid mechanics simulations. Phys Fluids. doi:10.1063/1.4819142
53. Long CC, Marsden AL, Bazilevs Y (2014) Shape optimization of pulsatile ventricular assist devices using FSI to minimize thrombotic risk. Comput Mech. (accepted for publication)
54. Takizawa K, Bazilevs Y, Tezduyar TE (2012) Space–time and ALE-VMS techniques for patient-specific cardiovascular fluid–structure interaction modeling. Arch Comput Meth Eng 19:171–225. doi:10.1007/s11831-012-9071-3
55. Tezduyar TE, Sathe S, Schwaab M, Conklin BS (2008) Arterial fluid mechanics modeling with the stabilized space–time fluid–structure interaction technique. Int J Numer Meth Fluids 57:601–629. doi:10.1002/fld.1633
56. Huang H, Virmani R, Younis H, Burke AP, Kamm RD, Lee RT (2001) The impact of calcification on the biomechanical stability of atherosclerotic plaques. Circulation 103:1051–1056
57. Bazilevs Y, Gohean JR, Hughes TJR, Moser RD, Zhang Y (2009) Patient-specific isogeometric fluid–structure interaction analysis of thoracic aortic blood flow due to implantation of the Jarvik (2000) left ventricular assist device. Comput Meth Appl Mech Eng 198:3534–3550
58. Saad Y, Schultz M (1986) GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comput 7:856–869
59. Takizawa K, Henicke B, Puntel A, Spielman T, Tezduyar TE (2012) Space–time computational techniques for the aerodynamics of flapping wings. J Appl Mech 79:010903. doi:10.1115/1.4005073
60. Rhee K, Han MH, Cha SH, Khang G (2001) The changes of flow characteristics caused by a stent in fusiform aneurysm models. Engineering in Medicine and Biology Society, 2001. In: Proceedings of the 23rd annual international conference of the IEEE, vol 1, pp 86–88. doi:10.1109/IEMBS.2001.1018852
61. Jou L-D, Mawad ME (2011) Hemodynamic effect of neuroform stent on intimal hyperplasia and thrombus formation in a carotid aneurysm. Med Eng Phys 33:573–580. doi:10.1016/j.medengphy.2010.12.013

62. Chung J, Hulbert GM (1993) A time integration algorithm for structural dynamics withimproved numerical dissipation: the generalized-α method. J Appl Mech 60:371–375
63. Jansen KE, Whiting CH, Hulbert GM (2000) A generalized-α method for integrating the filtered Navier–Stokes equations with a stabilized finite element method. Comput Meth Appl Mech Eng 190:305–319
64. Tezduyar TE, Sathe S, Keedy R, Stein K (2004) Space–time techniques for finite element computation of flows with moving boundaries and interfaces. In: Gallegos S, Herrera I, Botello S, Zarate F, Ayala G (eds) Proceedings of the III international congress on numerical methods in engineering and applied science. CD-ROM, Monterrey, Mexico
65. Tezduyar TE, Behr M, Mittal S, Johnson AA (1992) Computation of unsteady incompressible flows with the finite element methods—space–time formulations, iterative strategies and massively parallel implementations. In: New methods in transient analysis, PVP-Vol. 246/AMD, vol 143. ASME, New York, pp 7–24
66. Johnson AA, Tezduyar TE (1994) Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. Comput Meth Appl Mech Eng 119:73–94. doi:10.1016/0045-7825(94)00077-8
67. Isaksen JG, Bazilevs Y, Kvamsdal T, Zhang Y, Kaspersen JH, Waterloo K, Romner B, Ingebrigtsen T (2008) Determination of wall tension in cerebral artery aneurysms by numerical simulation. Stroke 39:3172–3178
68. Zhang Y, Wang W, Liang X, Bazilevs Y, Hsu M-C, Kvamsdal T, Brekken R, Isaksen J (2009) High-fidelity tetrahedral mesh generation from medical imaging data for fluid–structure interaction analysis of cerebral aneurysms. Comput Model Eng Sci 42:131–150
69. Bazilevs Y, Hsu M-C, Zhang Y, Wang W, Liang X, Kvamsdal T, Brekken R, Isaksen J (2010) A fully-coupled fluid–structure interaction simulation of cerebral aneurysms. Comput Mech 46:3–16
70. Bazilevs Y, Hsu M-C, Takizawa K, Tezduyar TE (2012) ALE-VMS and ST-VMS methods for computer modeling of wind-turbine rotor aerodynamics and fluid–structure interaction. Math Models Meth Appl Sci 22:1230002. doi:10.1142/S0218202512300025
71. Hughes TJR, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: cad, finite elements, nurbs, exact geometry, and mesh refinement. Compu Meth Appl Mech Eng 194:4135–4195
72. Cottrell JA, Hughes TJR, Bazilevs Y (2009) Isogeometric analysis. Wiley, Toward Integration of CAD and FEA
73. Kiendl J, Bletzinger K-U, Linhard J, Wüchner R (2009) Isogeometric shell analysis with Kirchhoff–Love elements. Comput Meth Appl Mech Eng 198:3902–3914
74. Kiendl J, Bazilevs Y, Hsu M-C, Wüchner R, Bletzinger K-U (2010) The bending strip method for isogeometric analysis of Kirchhoff–Love shell structures comprised of multiple patches. Comput Meth Appl Mech Eng 199:2403–2416
75. Bazilevs Y, Hsu M-C, Kiendl J, Wüchner R, Bletzinger K-U (2011) 3D simulation of wind turbine rotors at full scale. Part II: Fluid–structure interaction modeling with composite blades. Int J Numer Meth Fluids 65:236–253
76. Benson DJ, Bazilevs Y, De Luycker E, Hsu M-C, Scott M, Hughes TJR, Belytschko T (2010) A generalized finite element formulation for arbitrary basis functions: from isogeometric analysis to xfem. Int J Numer Meth Eng 83:765–785
77. Benson DJ, Bazilevs Y, Hsu M-C, Hughes TJR (2011) A large deformation, rotation-free, isogeometric shell. Comput Meth Appl Mech Eng 200:1367–1378
78. Tezduyar TE, Sathe S, Pausewang J, Schwaab M, Christopher J, Crabtree J (2008) Interface projection techniques for fluid–structure interaction modeling with moving-mesh methods. Comput Mech 43:39–49. doi:10.1007/s00466-008-0261-7
79. Tezduyar TE, Takizawa K, Moorman C, Wright S, Christopher J (2010) Space–time finite element computation of complex fluid–structure interactions. Int J Numer Meth Fluids 64:1201–1218. doi:10.1002/fld.2221
80. Bazilevs Y, Hsu M-C, Scott MA (2012) Isogeometric fluid–structure interaction analysis with emphasis on non-matching discretizations, and with application to wind turbines. Comput Meth Appl Mech Eng 249–252:28–41

81. Vignon-Clementel IE, Figueroa CA, Jansen KE, Taylor CA (2006) Outflow boundary conditions for three-dimensional finite element modeling of blood flow and pressure in arteries. Comput Meth Appl Mech Eng 195:3776–3796
82. Moghadam ME, Bazilevs Y, Hsia T-Y, Vignon-Clementel IE, Marsden AL (2011) The modeling of congenital hearts alliance (MOCHA), "a comparison of outlet boundary treatments for prevention of backflow divergence with relevance to blood flow simulations". Comput Mech 48:277–291 doi:10.1007/s00466-011-0599-0
83. Tezduyar TE, Senga M (2006) Stabilization and shock-capturing parameters in SUPG formulation of compressible flows. Comput Meth Appl Mech Eng 195:1621–1632. doi:10.1016/j.cma.2005.05.032
84. Bazilevs Y, Calo VM, Tezduyar TE, Hughes TJR (2007) YZβ discontinuity-capturing for advection-dominated processes with application to arterial drug delivery. Int J Numer Meth Fluids 54:593–608. doi:10.1002/fld.1484

Part III
Particle Methods in Coupled Problems

Direct Numerical Simulation of Particulate Flows Using a Fictitious Domain Method

Bircan Avci and Peter Wriggers

Abstract Multiphase flows consisting of a continuous fluid phase and a dispersed phase of macroscopic particles are present in many engineering applications. In general, a main task in the study of the particle-laden fluid flow of an application is to make predictions about the system's nature for various boundary conditions, since, depending on the volume fraction and mass concentration of the dispersed phase a fluid-particle system shows quite different flow properties. Unfortunately, often it is impossible to investigate such a system experimentally in detail or even at all. An option to capture and to predict its properties is performing a direct numerical simulation of the particulate fluid. For this purpose, a model approach based on a fictitious domain method is proposed in this contribution. Here, the fluid and the particle phase are treated, respectively, within the framework of the finite element method and the discrete element method. The coupling scheme, which accounts for the phase interaction, is realized at the particle scale. For the computation of the forces that the fluid exerts on a particle an approach is used in which they are determined directly from the flow field in the vicinity of its surface.

1 Introduction

Particulate flows are of great importance in very different industrial branches, e.g., in medical, process and chemical industries and also in geotechnical engineering and bioengineering. Examples include fluidized beds, sedimentation, fluvial erosion, sand production in oil wells, dust collection devices and aerosol transport in human respiratory airways. The characteristics involving particulate flows are up to now

B. Avci (✉) · P. Wriggers
Institute of Continuum Mechanics, Appelstrasse 11, 30167 Hannover, Germany
e-mail: avci@ikm.uni-hannover.de

P. Wriggers
e-mail: wriggers@ikm.uni-hannover.de

neither well-investigated nor well-understood, particularly with regard to flows with a dispersed phase composed of macroscopic particles. From the side of industry and applied sciences there exists an intense interest in understanding the processes taking place in these flows in order to predict their behavior, because a profound knowledge of the fluid-particle interaction would allow easily to improve the performance of an existing system or to design more efficient systems in the future; but the conduction of experiments on particulate flow systems—unless they can be carried out at all—is expensive and time-consuming. Performing virtual experiments on models via numerical simulations is of course an alternative way to gain some insight into the flow properties of a multiphase system. However, due to existing crucial limitations regarding the hardware technology and the scalability of algorithms, simulations cannot often totally replace real laboratory experiments. Thus, they are still required, but even so the knowledge extracted from representative small-scale virtual experiments can contribute in minimizing the sequence of real experiments. Consequently, the improvement of existing as well as the development of new frameworks for simulation of particulate flows is significant and demands more effort, as the simulation of problems with large number of particles is still a great challenging task.

In particle-laden multiphase flows the strength of the phase coupling is predominantly determined both by the local volume fraction of the dispersed phase and by its mass concentration (see, e.g., Crowe [11]). That means that in case of dense or locally highly concentrated flows the presence of particles in the fluid field can be a determining factor for the main characteristics inherent to an engineering system. To capture the mutual interaction of the phases in such flows, it is crucial to analyze the respective problem at the particle scale. This necessity implies a fully resolved model where the particles are described as having a body volume, and not just as point-particles, because in dense flows neighboring particles can interact not only via close range effects—like contact forces, adhesion or agglomeration—, but also via long range effects due to particle induced wakes, eddies and other local disturbances. Those effects can only be captured in the framework of a full 3D direct numerical simulation (DNS). However, a full 3D DNS approach requires very powerful techniques and is nontrivial, even when considering systems with only one immersed particle settling in a fluid, let alone systems with some thousands of dispersed particles in the fluid. This is due to the fact that besides the handling of the evolution of the time varying fluid domain, the motion of the fluid-particle boundaries needs to be continuously tracked during the flow process in order to account for the momentum exchanges taking place at the phase interfaces.

In the last two decades, great progress has been made in the development and improvement of DNS methods for particulate flow simulations. Basically, the proposed approaches may be classified into two groups: (i) adaptive grid methods and (ii) fixed grid methods. The first DNS approaches published in the literature are assigned to the category (i). Here, the fluid field is described using a body-fitted moving mesh whose elements follow locally the boundaries of the particles being in motion. Of course, depending on the particles' motion this can lead to large element distortions in a mesh, so that the grid needs to be re-meshed from time to time. Such a procedure is computationally very intensive, and one requires, in fact, very efficient and sophisticated mesh motion and re-meshing algorithms. The first 3D DNS computations of particulate flows belonging to this category were presented in Johnson and Tezduyar [23], which can be considered as a pioneering work in this area (see also Johnson and Tezduyar [24–26]). In these articles, the authors propose the deformable-spatial-domain/stabilized space–time (DSD/SST) finite element method (FEM) setting for the treatment of the fluid field. Further milestone contributions related to group (i) were published by Hu et al. [19–21]. These authors employ in their work the Arbitrary-Lagrangian-Eulerian method framework in order to describe the particulate fluid field. In terms of the approaches assigned to the fixed-grid category (ii), one can find in the literature a number of techniques suggested for the DNS analysis of fluid-particle interactions (see the review paper of Haeri and Shrimpton [17] and the references cited therein for an overview). Taken together the proposed approaches are known under the generic term fictitious domain (FD) method. The most widely-used ones are the immersed boundary method, the distributed Lagrange multiplier/fictitious domain method and the fictitious boundary method. They all have in common that the fluid flow is treated in the framework of an Eulerian setting, where a fixed mesh is employed. Here, a fluid mesh covers, compared to the approaches in group (i), the whole computational domain, also including the space of the particles—that means the solid domain is filled as well with fluid. The main idea on which the FD methods are based is to uncouple the particles from the mesh and to consider them as fictitious objects having the property to traverse through the grid without causing any element deformation. The crucial point is here to enforce the fluid enclosed by an embedded particle to adopt its solid body motion. In general, this is realized through imposing additional implicit constraints to the flow field.

A prominent method to simulate granular materials is the well-established discrete element method (DEM) approach, which was proposed originally by Cundall and Strack [12]. The application of a DEM solver to predict the behavior of the dispersed phase in a particulate flow can be found, e.g., in Wachs [41] and Avci and Wriggers [3]. In a DEM setting, usually a soft sphere approach based on repulsive force models is applied in order to describe the contact among particles, which are at the same time assumed being quasi-rigid.

In this work, a DNS approach is developed in the framework of a FD strategy for the numerical simulation of 3D particle-laden flows. The fluid-particle interactions are computed at the particle scale, with a fully resolved flow around the particles. As numerical solvers regarding the simulation of the fluid part and particle part, the FEM and DEM are used, respectively. Here, both methods are appropriately coupled by a staggered solution procedure to handle particulate flows.

2 Governing Equations

A multiphase domain $\Omega \in \mathbb{R}^3$ is considered to describe a particulate fluid that consists of a flow field $\Omega_f(t)$ and N particles. Therein, each particle $\mathscr{P}_i$ occupies the domain $\Omega^i_p(t)$. Hence, for Ω it follows: $\Omega = \Omega_f(t) \cup \{\Omega^i_p(t)\}_{i=1,N}$.

The flow of the fluid field is modeled by the non-stationary incompressible Navier–Stokes equations:

$$\rho_f\left(\frac{\partial \mathbf{u}_f}{\partial t} + \mathbf{u}_f \cdot \nabla \mathbf{u}_f\right) - \nabla \cdot \boldsymbol{\sigma} = \mathbf{0}, \quad \nabla \cdot \mathbf{u}_f = 0, \quad \forall \mathbf{x} \in \Omega_f. \tag{1}$$

Herein, $\mathbf{u}_f$ is the fluid velocity, ρ_f the fluid density and $\boldsymbol{\sigma}$ describes the Cauchy stress tensor. In the numerical studies the constitutive equation for a Newtonian flow is used:

$$\boldsymbol{\sigma} = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon} \quad \text{with} \quad \boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \mathbf{u}_f + (\nabla \mathbf{u}_f)^T\right), \tag{2}$$

where p is the pressure, $\mathbf{I}$ the identity tensor, μ the dynamic viscosity and $\boldsymbol{\varepsilon}$ the strain rate tensor.

The motion of a quasi rigid particle $\mathscr{P}$ can be deduced from the Newton–Euler equations. Consequently, its translational and angular velocities, $\mathbf{U} = \dot{\mathbf{X}}$ and $\boldsymbol{\omega}$, have to satisfy:

$$M\frac{\mathrm{d}^2\mathbf{X}}{\mathrm{d}t^2} = (\rho - \rho_f)V\mathbf{b} + \mathbf{F} + \mathbf{F}_f \tag{3}$$

$$\boldsymbol{\theta}\frac{\mathrm{d}\boldsymbol{\omega}}{\mathrm{d}t} + \boldsymbol{\omega} \times (\boldsymbol{\theta}\boldsymbol{\omega}) = \mathbf{T} + \mathbf{T}_f. \tag{4}$$

Therein, M is the mass, $\mathbf{X}$ the position vector to the center of mass $\mathscr{M}$, ρ the mass density, $\mathbf{b}$ the gravity and V denotes the volume of $\mathscr{P}$. The tensor of inertia is represented by $\boldsymbol{\theta}$. Furthermore, the sum of the contact forces is stated as $\mathbf{F}$ and the fluid force that acts upon the particle surface $\partial\Omega_p$ is considered by $\mathbf{F}_f$. The torques that are caused by $\mathbf{F}$ and $\mathbf{F}_f$ with respect to $\mathscr{M}$ are associated to the quantities $\mathbf{T}$ and $\mathbf{T}_f$, respectively. Hence, the fluid forces can be obtained by:

$$\mathbf{F}_f = \int_{\partial\Omega_p} \mathbf{t}\,\mathrm{d}A \quad \text{and} \quad \mathbf{T}_f = \int_{\partial\Omega_p} \mathbf{r} \times \mathbf{t}\,\mathrm{d}A. \tag{5}$$

The traction vector $\mathbf{t}$ on $\partial\Omega_p$ is defined as $\mathbf{t} = \boldsymbol{\sigma}\mathbf{n}_f$, where $\mathbf{n}_f$ is the unit outward normal vector and $\mathbf{r}$ is the position vector of a point at $\partial\Omega_p$ with respect to $\mathscr{M}$.

3 Constitutive Modeling of the Particle Phase

The particles, which are immersed in the fluid, are modeled as quasi rigid spheres. To describe their collision behavior, a force-displacement based approach relying on repulsive models is used, which allows to determine the inter-particle forces. In the sections below, the relevant concepts of the contact models are stated briefly.

3.1 Normal Contact Model

The normal contact forces acting between colliding particles and between particles and system boundaries are described by a constitutive viscoelastic model. For adhesive particles being in contact, the JKR theory, introduced by Johnson et al. [27], is used to determine the resultant attractive van der Waals force F_a^n in the contact area (see also Maugis [35] for a detailed description of this model). As shown by Loskofsky et al. [31], the JKR theory yields even in the case of underwater adhesion satisfying results. For the purpose of governing the elastic contact force F_e^n, the Hertzian law [18] constitutes a well-established model. If the particles to be treated have also viscous material properties, for this, a consistent phenomenological model was presented in Brilliantov et al. [5, 6], where the effect of viscosity is considered via an added dissipative force F_d^n. Thus, one obtains for the forces acting on a particle:

$$F^n = F_e^n - F_a^n + F_d^n. \tag{6}$$

The elastic repulsive force based on the Hertzian contact law is determined by:

$$F_e^n = \frac{4}{3} E \sqrt{R}\, \delta^{3/2}, \tag{7}$$

where $\delta = \delta_i + \delta_j$ is the total particle compression, R and E are the effective radius and the effective Young's modulus of the contact pair $\mathscr{P}_i$ and $\mathscr{P}_j$, respectively (see Hertz [18]). As a result of the mutual compression of the particles, a circular area is formed in the contact zone. In the Hertz model, the radius a of this area, which is often called contact radius, is related to the total deformation δ via $a^2 = R\delta$.

According to the JKR model, it is implied that the adhesive force acts only within the contact area. Here, the work of adhesion to separate a unit contact area of $\mathscr{P}_i$ and $\mathscr{P}_j$ in a liquid medium (l) is defined as $W = \gamma_{il} + \gamma_{jl} - \gamma_{ij}$, where γ describes the respective interfacial energy (see, e.g., Loskofsky et al. [31]). Since the adhesive force F_a^n is opposed to the elastic force F_e^n, it reduces the elastic deformation δ_e, and one obtains for the total deformation:

$$\delta := \delta_e - \delta_a = \frac{a^2}{R} - \sqrt{\frac{2\pi\, W a}{E}}, \tag{8}$$

where the second term δ_a is due to adhesion (see, e.g., Maugis [35] for details). Based on this model the difference between the elastic and the adhesive force is given by:

$$F_{ea}^n := F_e^n - F_a^n = \frac{4Ea^3}{3R} - 2\pi a^2 \sqrt{\frac{2WE}{\pi a}}. \tag{9}$$

A special case here is the situation when external forces are absent. If so, an equilibrium contact area with radius a_0 is formed in the contact zone due to the mutual compression δ_0 of the particles caused just by their adhesive attraction. These both quantities are defined as:

$$a_0 = \left(\frac{9\pi\, W R^2}{2E}\right)^{1/3} \quad \text{and} \quad \delta_0 = \left(\frac{3R}{4}\left(\frac{\pi\, W}{E}\right)^2\right)^{1/3}. \tag{10}$$

To separate the particles, one has to apply a traction force under which they suffer minute stretching deformations forming a connecting neck around the contact zone. Once the pulling force has reached a critical level, i.e., $F^n = -F^n_c$, the contact breaks. Here, the critical force is obtained by $F^n_c = 3\pi\, WR/2$ and the corresponding critical deformation of the particle pair is $\delta_c = a_0^2/(48^{1/3}R)$. That means, the pulling distance regarding their detachment is defined as $\delta = -\delta_c$. By incorporating these critical quantities, one yields for the displacement δ in (8) and the force F^n_{ea} in (9) the following dimensionless relationships (see Chokshi et al. [9]):

$$\frac{\delta}{\delta_c} = 6^{1/3}\left[2\left(\frac{a}{a_0}\right)^2 - \frac{4}{3}\left(\frac{a}{a_0}\right)^{1/2}\right] \tag{11}$$

$$\frac{F^n_{ea}}{F_c} = 4\left(\frac{a}{a_0}\right)^3 - 4\left(\frac{a}{a_0}\right)^{3/2}. \tag{12}$$

To consider the properties of material viscosity, a dissipative force is adopted according to Brilliantov et al. [5]. In that work, the definition of the force is given as $F^n_d = A\dot{a}\, \partial F^n_{ea}/\partial a$. From this definition, the viscous force can be written as follows:

$$F^n_d = A\dot{a}\left(\frac{4Ea^2}{R} - \frac{3}{2}\sqrt{8\pi\, WEa}\right), \tag{13}$$

where the dissipative factor A is related to a constant function of material viscosity, which can also be used as a fitting parameter.

In the present contribution, the force laws on which F^n in (6) is based are algorithmically treated as displacement driven (see, e.g., Wriggers [43]). Introducing the penetration measure $g^n = (R_i + R_j) - ||\mathbf{X}_i - \mathbf{X}_j|| > 0$ and equating $g^n \equiv \delta$ as the mutual compression of $\mathscr{P}_i$ and $\mathscr{P}_j$, the individual parts of the force F^n can be computed straightforward after having determined the contact radius a. To evaluate this quantity, one has to solve the nonlinear expression in (11) for the currently calculated penetration δ of the examined pair of particles. The direction of the contact force $\mathbf{F}^n = F^n\mathbf{n}$ of the respective particle is opposed to the direction of its displacement, where $\mathbf{n} = (\mathbf{X}_i - \mathbf{X}_j)/||\mathbf{X}_i - \mathbf{X}_j||$ is the normal unit vector pointing from $\mathscr{M}_j$ towards $\mathscr{M}_i$.

3.2 Tangential Contact Model

The constitutive relation of Coulomb's law couples the tangential force F^t via the coefficient of friction to the normal force F^n such that the relation $F^t = \mu_d F^n$ holds for sliding and $F^t \leq \mu_s F^n$ for sticking. Therein, the dynamic and the static friction coefficients are denoted by μ_d and μ_s, respectively, where $\mu_d \leq \mu_s$. For a constitutive treatment of F^t, a classical tangential (linear) spring-dashpot element with an incorporated slider is used in this work in order to model the tangential friction problem. For an overview and a discussion of different tangential contact models proposed in the literature in the context of the DEM see Kruggel-Emden et al. [28]. Here, a return mapping scheme is adopted for the computation of the tangential force (see Luding [32], Wriggers [43]). This projection method needs a tangential trail traction that takes the form:

$$\mathbf{F}^t_o = -(c^t \mathbf{g}^t + d^t \mathbf{v}^t). \tag{14}$$

Therein, $\mathbf{g}^t$ is the elongation of the tangential spring, c^t and d^t are the tangential spring stiffness and the tangential dissipation parameter, respectively. The tangential relative velocity at the contact point $\mathscr{C}$ is given by $\mathbf{v}^t = \mathbf{v}^s - (\mathbf{v}^s \cdot \mathbf{n})\,\mathbf{n}$ with $\mathbf{v}^s = \mathbf{v}^{\mathscr{C}}_i - \mathbf{v}^{\mathscr{C}}_j$ as the relative velocity at $\mathscr{C}$, where the corresponding local velocities are defined by $\mathbf{v}^{\mathscr{C}}_i = \mathbf{U}_i + \boldsymbol{\omega}_i \times \mathbf{r}_i$ and $\mathbf{v}^{\mathscr{C}}_j = \mathbf{U}_j + \boldsymbol{\omega}_j \times \mathbf{r}_j$. The vectors pointing from $\mathscr{M}_i$ and $\mathscr{M}_j$ to $\mathscr{C}$ are associated with $\mathbf{r}_i = R_i(-\mathbf{n})$ and $\mathbf{r}_j = R_j\mathbf{n}$, respectively. By introducing a trial function f^{tr}, the following relation can be stated for the tangential contact:

$$f^{\text{tr}} := ||\mathbf{F}^t_o|| - \mu_s ||\mathbf{F}^n|| \Rightarrow \begin{cases} \leq 0 : & \text{stick} \\ > 0 : & \text{slip.} \end{cases} \tag{15}$$

If $f^{\text{tr}} \leq 0$, the contact point $\mathscr{C}$ is in the stick region, and if $f^{\text{tr}} > 0$, it is in the slip region. In the latter case, sliding occurs in the contact area. Note that if $F^t_o < \mu_d F^n$ becomes valid during a sliding process, then the stick case comes into effect. If the contact point sticks within the current time step, the actual tangential spring $\mathbf{g}^t$ is incremented for the succeeding time step by the relation $\Delta\mathbf{g}^t = \mathbf{v}^t \Delta t_{\text{D}}$. Consequently, the new spring length is defined by $\overline{\mathbf{g}}^t = \mathbf{g}^t + \Delta\mathbf{g}^t$. Here, Δt_{D} denotes the time step size of the DEM. But if the contact point slides within the time step, then the tangential spring is adjusted by means of:

$$\overline{\mathbf{g}}^t = -\frac{1}{c^t}(\mu_d F^n \mathbf{t} + d^t \mathbf{v}^t) \tag{16}$$

in order to fulfill the Coulomb's slip condition. Therein, $\mathbf{t} = \mathbf{F}^t_o / ||\mathbf{F}^t_o||$ is the direction of the trial traction. In two subsequent time steps, the contact area might be slightly rotated. To take this into account, one can—as proposed in Luding [32]—project the tangential spring onto the current rotated contact area at the beginning of each new time step via $\mathbf{g}^t = \overline{\mathbf{g}}^t - (\overline{\mathbf{g}}^t \cdot \mathbf{n})\mathbf{n}$. Finally, in the context of the return mapping scheme, the tangential contact force is $F^t = ||\mathbf{F}_o^t||$ if $f^{\text{tr}} \leq 0$ holds and $F^t = \mu_d F^n$ if $f^{\text{tr}} > 0$. By computing F^t, one obtains $\mathbf{F}^t = F^t \mathbf{t}$ and $\mathbf{T}^t = \mathbf{r} \times \mathbf{F}^t$ that contributes to $\mathbf{F}$ and $\mathbf{T}$ in (3) and (4), respectively.

3.3 Rolling Resistance Model

During a rolling motion of two particles the leading part of the contact area is continuously compressed and the trailing part is decompressed with respect to the rolling direction. In case of an attractive van der Waals force in the contact area, the particles suffer an opposing torque that generates rolling resistance. Here, a model consisting of a rolling spring-dashpot-slider element is adopted (see Iwashita and Oda [22]). At this, the opposing torque is given by:

$$\mathbf{M}_o^r = -(c^\phi \boldsymbol{\phi} + d^\phi \dot{\boldsymbol{\phi}}). \tag{17}$$

Therein, c^ϕ is the rolling stiffness, d^ϕ the rolling viscosity coefficient, $\boldsymbol{\phi}$ the relative particle rotation and $\dot{\boldsymbol{\phi}}$ denotes the corresponding relative rotational velocity. By introducing a trial force $\mathbf{F}_o^r$ that induces an equivalent torque to $\mathbf{M}_o^r$, the problem of rolling resistance can be treated algorithmically like the model of tangential friction (see Luding [33]). In this regard, the equivalent formulation can be stated as follows:

$$\mathbf{M}_o^r = R\,\mathbf{n} \times \mathbf{F}_o^r, \tag{18}$$

with $\mathbf{F}_o^r = -(c^\xi\, \mathbf{g}^r + d^\xi\, \mathbf{v}^r)$, $c^\xi = c^\phi / R^2$ and $d^\xi = d^\phi / R^2$. Here, $\mathbf{g}^r$ is the elongation of the spring and $\mathbf{v}^r$ denotes the rolling velocity that can be computed according to Kuhn and Bagi [29] using:

$$\mathbf{v}^r = -R\Big[(\boldsymbol{\omega}_i - \boldsymbol{\omega}_j) \times \mathbf{n} - \frac{1}{2}\Big(\frac{1}{R_j} - \frac{1}{R_i}\Big)\mathbf{v}^t\Big]. \tag{19}$$

Assuming that the slider can sustain a certain critical rolling resistance torque M_c^r, one can write analog to (15):

$$f_r^{\text{tr}} := ||\mathbf{F}_o^r|| - \frac{M_c^r}{R} \Rightarrow \begin{cases} \leq 0 \ : \ \text{stick} \ \Rightarrow \ F^r = ||\mathbf{F}_o^r|| \\ > 0 \ : \ \text{slip} \ \Rightarrow \ F^r = M_c^r / R. \end{cases} \tag{20}$$

With regard to the numerical handling of the spring in this context, the respective relationships can be expressed in a summarized form as:

$$f_r^{\text{tr}} := \begin{cases} < 0 \ \Rightarrow \ \bar{\mathbf{g}}^r = \mathbf{g}^r + \Delta \mathbf{g}^r, & \Delta \mathbf{g}^r = \mathbf{v}^r \Delta t \\ > 0 \ \Rightarrow \ \bar{\mathbf{g}}^r = -\dfrac{1}{c^\xi}\Big[F_c^r\, \mathbf{t}^r + d^\xi\, \mathbf{v}^r\Big], & \mathbf{t}^r = \dfrac{\mathbf{F}_o^r}{||\mathbf{F}_o^r||}, \ F_c^r = \dfrac{M_c^r}{R}. \end{cases} \tag{21}$$

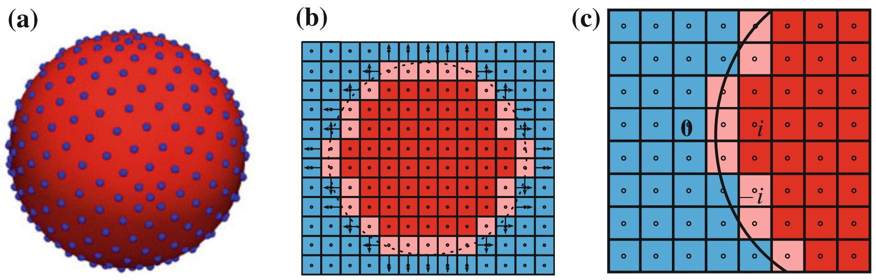


Fig. 1 **a** Distribution of the Lebedev quadrature points for the case of $N_L = 302$ points. **b** Steplike discretization of a particle for the evaluation of fluid forces. **c** A detail of the domain Ω showing the classification of computational elements

In this model, the projection direction of the spring relies on the common rolling direction of the pair of particles being in contact. Thus, the projection condition is defined as $\mathbf{g}^r = (\bar{\mathbf{g}}^r \cdot \tilde{\mathbf{t}})\, \tilde{\mathbf{t}}$, where $\tilde{\mathbf{t}} = \mathbf{v}^r / ||\mathbf{v}^r||$. By computing F^r, one yields the pseudo force $\mathbf{F}^r = F^r\, \mathbf{t}^r$ and respectively the rolling resistance torque $\mathbf{M}^r = R\mathbf{n} \times \mathbf{F}^r$. Finally, the torques for the examined pair of particles $\{\mathscr{P}_i, \mathscr{P}_j\}$ can be written as follows: $\mathbf{M}_i^r = -\mathbf{M}_j^r =: \mathbf{M}^r$.

4 Phase Coupling

In the fixed grid approach, the mesh of the flow field does not coincide with the boundaries of the particles. Hence, information between the Eulerian and Lagrangian description has to be transferred. A further challenging issue concerning the coupling of the phases is the computation of the fluid forces acting on the particle surfaces.

4.1 *Evaluation of the Fluid Forces*

A crucial point regarding the study of fluid-particle interactions in a fully resolved 3D DNS framework is the computation of the fluid forces to which the immersed particles are subjected. To carry out this task, two different strategies, as shown in Fig. 1a, b, have been considered in this work. In the following, the approach illustrated in (a) should be abbreviated as AP1 and the one in (b) as AP2.

Approach AP1 For the integration of the fluid forces acting on the surface of a particle a quadrature rule can be used that was developed in Lebedev and Laikov [30]. In Fig. 1a, the distribution of the Lebedev integration points, which can also be seen as Lagrangian force points, is displayed for the case of $N_L = 302$ points. This numerical integration applied to particle $\mathscr{P}_i$ yields:

$$\mathbf{F}_f^i = \int\limits_{\partial\Omega_p^i} \mathbf{t}\,\mathrm{dA} = \sum_{k=1}^{N_L} (d\mathbf{F})_k = J \sum_{k=1}^{N_L} w_k \mathbf{t}_k \tag{22}$$

$$\mathbf{T}_f^i = \int\limits_{\partial\Omega_p^i} \mathbf{r} \times \mathbf{t}\,\mathrm{dA} = \sum_{k=1}^{N_L} \mathbf{r}_k \times (\mathrm{d}\mathbf{F})_k = J \sum_{k=1}^{N_L} \mathbf{r}_k \times w_k \mathbf{t}_k. \tag{23}$$

Therein, $\mathbf{t}_k = (\langle\boldsymbol{\sigma}\rangle\mathbf{n})_k$ is the traction vector at the kth Lebedev point, w_k the corresponding integration weight and $\langle\boldsymbol{\sigma}\rangle = (1/V)\int \boldsymbol{\sigma}\,\mathrm{d}V$ specifies the averaged fluid stress tensor of the finite element with volume V in which the point k is located. The mapping from a unit sphere to $\mathscr{P}_i$ is performed via the relation $J = 4\pi R_i^2$, where R_i is the particle radius.

Approach AP2 In this approach to the evaluation of the fluid forces exerting on $\mathscr{P}_i$, the shape of the particle surface is reproduced on the basis of the computational grid, see Fig. 1b. Here, the calculation of $\mathbf{F}_f^i$ and $\mathbf{T}_f^i$ is carried out using the steplike reconstructed surface contour of $\mathscr{P}_i$. To determine these forces from this surface, one has to compute the averaged tractions on the respective element faces $\overline{\Gamma}_j$ $\{j = 1, \ldots, N_\Gamma\}$, and subsequently the corresponding forces can simply be summed up yielding:

$$\mathbf{F}_f^i = \int\limits_{\partial\Omega_p^i} \mathbf{t}\,\mathrm{d}\Gamma = \sum_{j=1}^{N_\Gamma} (\mathrm{d}\mathbf{F})_j = \sum_{j=1}^{N_\Gamma} \mathbf{t}_j \overline{\Gamma}_j \tag{24}$$

$$\mathbf{T}_f^i = \int\limits_{\partial\Omega_p^i} \mathbf{r} \times \mathbf{t}\,\mathrm{d}\Gamma = \sum_{j=1}^{N_\Gamma} \mathbf{r}_{\Gamma_j} \times (\mathrm{d}\mathbf{F})_j = \sum_{j=1}^{N_\Gamma} \mathbf{r}_{\Gamma_j} \times \mathbf{t}_j \overline{\Gamma}_j, \tag{25}$$

where $\mathbf{r}_{\Gamma_j}$ is the position vector of the center of the element surface $\overline{\Gamma}_j$ with respect to $\mathscr{M}_i$.

Remark To characterize the motion of the DEM particles sliding through the mesh, the mesh elements are labeled as depicted in Fig. 1c. By using the position of the element center point $\mathscr{E}$ as an assignment criteria, the elements that coincide with a particle domain Ω_p^i are marked at each time step by the particle number of $\mathscr{P}_i$. Here, the interior and boundary elements are defined by ‘ i ’ and ‘ $-i$ ’, respectively. Furthermore, fluid elements are denoted by ‘ 0 ’.

4.2 Coupling Constraints

For the coupling process, the rigid body motion of the particles is imposed on the flow field. The rigidity constraints due to $\mathscr{P}_i$ that are applied to the Navier–Stokes equations can be regarded as additional Dirichlet no-slip boundary conditions (see, e.g., Wan and Turek [42]). The constraint for an interior velocity node $\mathscr{V} \in \Omega_p^i$ of an boundary element ‘ $-i$ ’ is given by:

$$\mathbf{u}_f = \mathbf{U}_i + \boldsymbol{\omega}_i \times \mathbf{r}_p, \tag{26}$$

where $\mathbf{r}_p$ is the vector pointing from $\mathscr{M}_i$ to the considered velocity node. However, for a velocity node $\mathscr{V} \in \Omega_p^i$ that adjoins the fluid phase, the velocity constraint is defined as:

$$\mathbf{u}_f = (1 - \phi_A)\mathbf{u}_f + \phi_A(\mathbf{U}_i + \boldsymbol{\omega}_i \times \mathbf{r}_p), \tag{27}$$

where ϕ_A is the element face area fraction situated within Ω_p^i. Hence, ϕ_A acts as a weighting factor for $\mathscr{V}$.[1] But if $\mathscr{V} \notin \Omega_p^i$, then a nonlinear weighted strategy is applied according to Luo et al. [34], which reads:

$$\mathbf{u}_f = (1 - \phi_R)\mathbf{u}_f + \phi_R(\mathbf{U}_i + \boldsymbol{\omega}_i \times \mathbf{r}_p), \tag{28}$$

where the interpolation factor $\phi_R = e^{-Re_p|\alpha|}$ is a nonlinear function of the relative Reynolds number $Re_p = \rho_f||\mathbf{U}_i - \mathbf{u}_f||D_i/\mu_f$ and of the relative distance $\alpha = h/D_i$. Here, D_i and h denote the particle diameter and the distance from $\mathscr{V}$ to the surface of $\mathscr{P}_i$, respectively.

5 Solution Algorithms

5.1 FEM Solver for the Fluid Problem

A spatial and temporal discretization of (1) yields a set of nonlinear algebraic equations for the fluid velocity $\mathbf{u}_f$ and the pressure $\mathbf{p}$. The resulting coupled equation system, which has to be solved at each time step, can be written as follows:

$$\begin{bmatrix} \mathbf{M} + \theta_1 \Delta t \mathbf{N}(\mathbf{u}_f^{n+1}) & \theta_2 \Delta t \mathbf{G} \\ \mathbf{G}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{u}_f^{n+1} \\ \mathbf{p}^{n+1} \end{bmatrix} = \begin{bmatrix} (\mathbf{M} - \theta_3 \Delta t \mathbf{N}(\mathbf{u}_f^n))\, \mathbf{u}_f^n \\ \mathbf{0} \end{bmatrix}. \tag{29}$$

Therein, $\mathbf{M}$ is the mass matrix, $\mathbf{N}$ the matrix including the diffusive, convective and stabilization terms, $\mathbf{G}$ the gradient matrix, $\mathbf{G}^T$ the divergence matrix, Δt the current time-step size and θ_{1-3} are parameters of the fractional step θ-scheme, see Turek

[1] In the present work, the nonconforming rotated trilinear $\tilde{Q}_1/Q_0$ element pair is used where a nodal value at $\mathscr{V}$ is the mean value of the velocity vector over the respective element face area, see Turek [38] for details. The velocity nodes of this element are located at the midpoints of the element faces.

[38] for details. To solve (29), the multigrid FEM solver FEATFLOW [39] is applied in this contribution.

5.2 DEM Solver for the Particle Problem

The movement of the particles is governed by the equations of motions given in (3) and (4). With regard to the numerical integration of these equations in time, the finite difference based Gear predictor–corrector scheme of third and fourth order is applied. The Gear integration scheme is subdivided into three steps: (1) prediction of all the kinematic variables, (2) evaluation of the forces according to Sect. 3 by using the predicted variables and, subsequently, computation of the corresponding accelerations and, finally, (3) correction of the predicted kinematic variables based on the new accelerations. For the algorithmic details of the Gear scheme see Allen and Tildesley [1].
Remark An aspect that has also to be considered in the framework of developing a FEM-DEM coupling scheme is the widely differing computational time scale between the both numerical methods. Since, due to the displacement driven character of the DEM concerning the force computation, one usually has $\Delta t_{\mathrm{D}} \ll \Delta t$. To handle this problem of unequal time scales, a sub cycling strategy can be used as suggested by Feng et al. [13].

5.3 Search Algorithms

The computation of the contact forces is the most CPU time consuming part of a DEM simulation. Here, the evaluation of the contact detection has to be minimized to the neighbors of a particle, since they are its only relevant potential contact partners. For this purpose, the Verlet-List and the Linked-Cell is combined in order to yield a fast contact search algorithm (see, e.g., Allen and Tildesley [1], Pöschel and Schwager [36]).

In the Verlet-List procedure a list of neighboring particle indices is maintained for each particle in the system. By defining a Verlet distance threshold value v_d, a pair of particles can be considered as neighbor if $v_d > |g^n|$ holds. Once the Verlet-List is built, the contact detection needs only to be evaluated for the neighboring pairs. As a result, this task scales with respect to the corresponding computational effort with $O(N)$. Certainly, the list has to be updated at some intervals. In this regard, a possible rebuild criteria can be defined by $\Delta s_{\max} \geq 0.6 v_d$, where $\Delta s_{\max}$ is the largest displacement of a particle since the last list update (see Pöschel and Schwager [36]). But the construction of the list in a straightforward manner scales with $O(N^2)$, thus, one has to speed up this task in order to obtain a search algorithm that scales in toto with $O(N)$. For this purpose, the computational domain is divided into cubic cells of uniform edge lengths where the cells are slightly larger than the largest particle in the system. After assigning all particles to the cells relative to their center of mass, the relevant particles for the construction of the neighbor list of $\mathscr{P}_i$ are those who are referenced to the group of 27 cells consisting of the owner cell of $\mathscr{P}_i$ and of its direct 26 surrounding cells.

For a fast assignment of the element flags and, furthermore, in order to localize efficiently the elements containing the integration points for the computation of the fluid forces on the particles, the approach of the Linked-Cell method is used analogously. The Linked-Cell algorithm generates in this case an element list referring to the same cell structure as for the particles. Here, an element is referenced to a cell with respect to the position of the elements center point $\mathscr{E}$. Consequently, for the application of the velocity constraints related to particle $\mathscr{P}_i$ and for the computation of its fluid forces, only the elements are significant that are binned into the group of those 27 cells which are relevant for $\mathscr{P}_i$. In order to reduce the trial computations, some elements can also be excluded in advance from detailed considerations if the distance between $\mathscr{E}$ and $\partial\Omega_p^i$ is larger than a threshold value that can be chosen according to the largest element size in the computational domain.

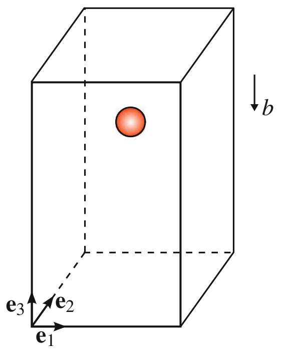
6 Numerical Simulations

The numerical results of three computed test problems obtained by the presented approach are discussed next. The first test problem is the sedimentation of one particle in a box. In the second simulation example, the sedimentation of two particles in a row is considered in order to mimic their *drafting-kissing-tumbling* effect, and the last example deals with a particle-laden flow through a tube with changing cross section.

6.1 Sedimentation of a Single Particle in a Box

In this example, the sedimentation of a single particle in a box filled with fluid is examined. The considered system is shown in Fig. 2. This system corresponds to the setup that was experimentally investigated in ten Cate et al. [37] where the authors measured the settling velocity of the immersed particle under the action of gravity in four test cases, each with a different fluid. In the following, the obtained simulation results for the cases with minimum and maximum terminal particle Reynolds numbers, $Re_p = 1.5$ and $Re_p = 31.9$, are presented. Previously, the sedimentation problem of ten Cate et al. [37] was computed by, among others, Veeramani et al. [40] and Feng and Michaelides [14].

To discretize the box in Fig. 2, a uniform mesh consisting of 819,200 $\tilde{Q}_1/Q_0$ elements ($80 \times 80 \times 128$ elements) is used. All the simulations were carried out by imposing no-slip velocity conditions at the box boundaries. The diagrams in Fig. 3 show the computed temporal evolutions of the settling velocity of the particle



Box dimension:	$10 \times 10 \times 16\,\text{cm}$
Particle position:	$(5/5/12.75)\,\text{cm}$
Particle radius:	$R = 0.75\,\text{cm}$
Particle density:	$\rho_P = 1.12\,\text{g/cm}^3$
Gravity:	$b = 980\,\text{cm/s}^2$

Fluid:

Case	$\rho_f\,[\text{g/cm}^3]$	$\nu_f\,[\text{cm}^2/\text{s}]$	$Re_P = U_\infty D/\nu_f$
C1	0.970	3.8454	1.5
C2	0.960	0.6042	31.9

Fig. 2 Sedimentation of a single particle in a box. Geometry and material data

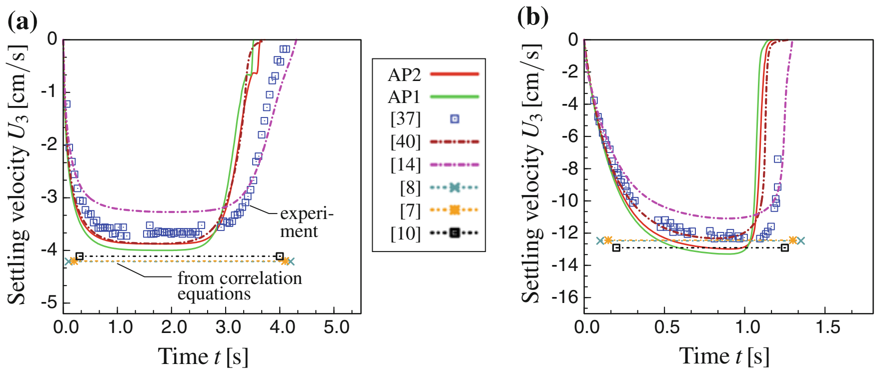


Fig. 3 Evolution of the settling velocity of the particle for **a** case C1 and **b** case C2

center in the direction of gravity for the considered two cases. As it can be seen, each case was simulated both by means of approach AP1 and AP2. Every diagram also includes the predicted terminal velocity of the particle based on the correlation equations[2] suggested in Clift et al. [10], Brown and Lawler [7] and Cheng [8], and furthermore, the numerical results of Veeramani et al. [40] and those of Feng and Michaelides [14]. At the beginning of the experiment, the particle is at rest, and it accelerates due to the action of gravity. It is observed that when the gravitational and the drag forces reach a state of equilibrium, the particle will sediment with a uniform velocity, which is called terminal velocity. The simulation results show that the presented model, both based on AP1 and AP2, is capable to predict the evolution of the particle's settling velocity. The maximum discrepancy of the predictions with

[2] In general, correlations for drag and terminal settling velocity are valid for a particle in an infinite domain, but they still provide reasonable results for a relatively large distance between a particle and system boundaries.

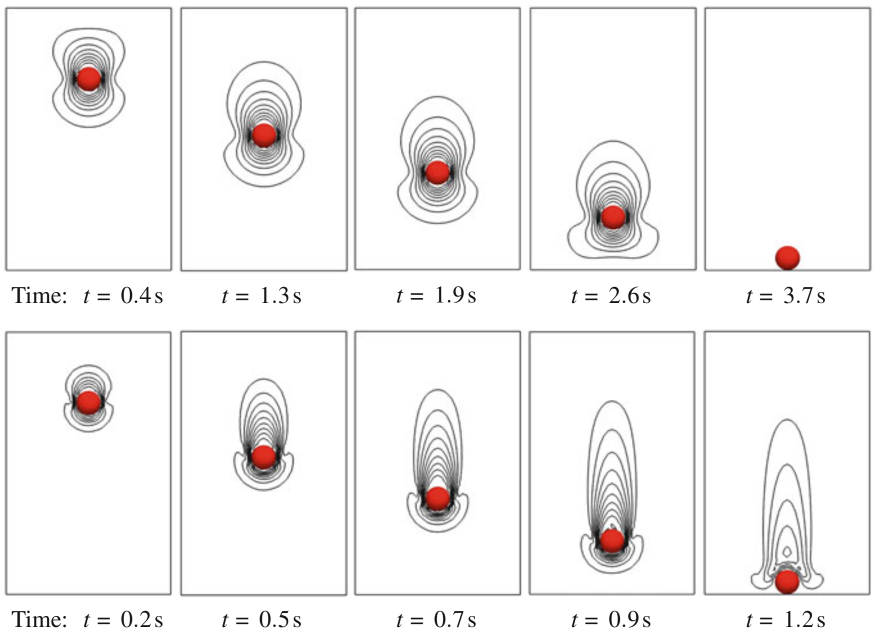
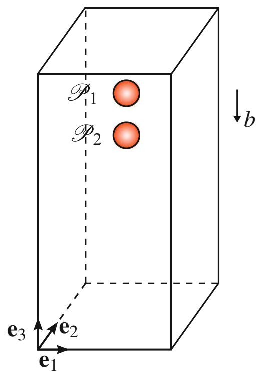


Fig. 4 Contour plots of the normalized velocity magnitude in the symmetry plane at different points in time. The plots of the *upper row* belong to case C1 and those of the *lower row* to case C2

respect to the experimental data is less than 8 %. In addition, the obtained results are in a very good agreement with those of Veeramani et al. [40], but there is a small mismatch compared with the results of Feng and Michaelides [14]. Figure 4 shows the computed contour plots of the normalized velocity magnitude $||\mathbf{u}_f||/U_\infty$ in the symmetry plane. Accordingly, the depicted contours range between 0 and 1. Here, an equal spacing of 0.1 is chosen. It shows that these plots agree well with those of ten Cate et al. [37], and that the presented model is well suited to mimic their sedimentation experiments. There is also a good agreement with the computed plots given in Apte et al. [2].

6.2 Sedimentation of Two Particles in a Box

This benchmark simulation is carried out to reproduce the *drafting-kissing-tumbling* effect of two particles sedimenting in a row, as it can be observed in laboratory experiments (see, e.g., the experiment reported in Fortes et al. [15]). Figure 5 shows the system with the particles being studied in this benchmark test. The depicted setup has been already numerically investigated by Glowinski et al. [16], Apte et al. [2] and Breugem [4]. For the discretization of the computational domain a uniform mesh of 1,048,576 $\tilde{Q}_1/Q_0$ elements ($64 \times 64 \times 256$ elements) is used. It is assumed that



	Box dimension:	$1 \times 1 \times 4\,\text{cm}$
	Particle position:	$\mathscr{P}_1\ (\approx 0.5/\approx 0.5/3.5)\,\text{cm}$
		$\mathscr{P}_2\ (0.5/0.5/3.167)\,\text{cm}$
	Gravity:	$b = 980\,\text{cm/s}^2$
Fluid:	Viscosity:	$\nu_f = 0.01\,\text{cm}^2/\text{s}$
	Density:	$\rho_f = 1.00\,\text{g/cm}^3$
Particle:	Radius:	$R_1 = R_2 = 1/12\,\text{cm}$
	Density:	$\rho_{P_1} = \rho_{P_2} = 1.14\,\text{g/cm}^3$
	Young modulus:	$E = 10^6\,\text{N/cm}^2$
	Poisson ratio:	$\nu = 0.25$
	Friction coeff.:	$\mu_s/\mu_d = 0.35/0.32$
	Damping coeff.:	$A = 5 \cdot 10^{-5}\,\text{s}$

Fig. 5 Sedimentation of two particles in a box. Geometry and material data

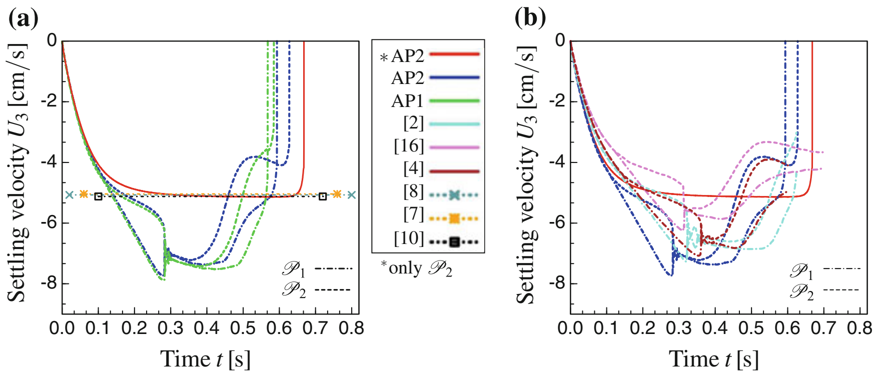


Fig. 6 Evolution of the settling velocities of the particles for the drafting-kissing-tumbling problem: **a** results of the present work and **b** comparison of the results with the literature

the box is fully filled with fluid. Concerning the boundary conditions of the fluid domain, no-slip velocity conditions are applied at the box walls. In order to provoke that the particles tumble when they kiss each other, a slight initial horizontal offset in the position of $\mathscr{P}_1$ is introduced such that $(0.5075/0.5075/3.5)$ cm. This offset is here necessary, because the numerical results of previous test computations based on different uniform symmetric meshes showed that the implemented algorithm maintains nearly the complete symmetric properties of the system. Thus, the flow field features too weak lateral disturbances in order to trigger the tumbling case (as if, for instance, a free or just an anisotropic mesh is used).

In order to verify that the numerical model is able to predict the terminal settling velocity of an almost undisturbed sedimenting single particle within the frame of this benchmark setup, the sedimentation of $\mathscr{P}_2$ was computed in advance without

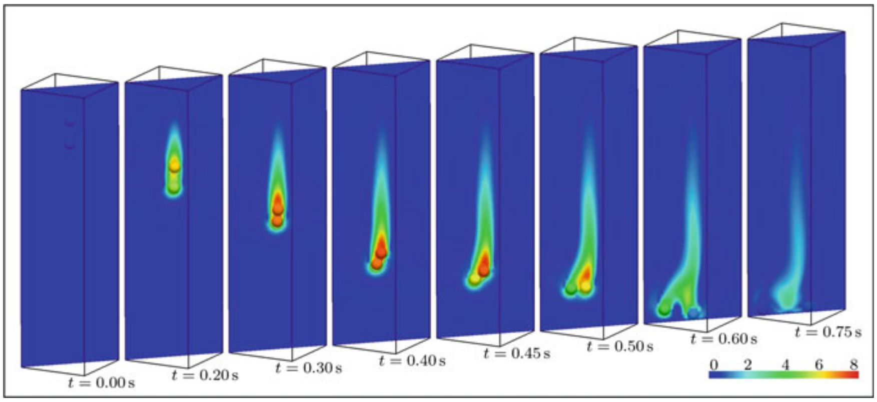


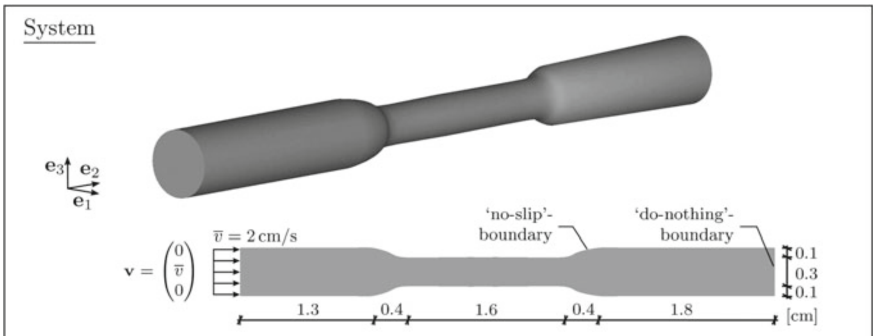
Fig. 7 Numerical results of the drafting-kissing-tumbling simulation at eight points in time

the presence of $\mathscr{P}_1$. Figure 6a shows the obtained time history of the falling velocity by using AP2, and one can see that the predicted terminal velocity matches those obtained with correlations. This diagram contains also the computed evolutions of the particle velocities by employing the approaches AP1 on the one hand and AP2 on the other hand for the *drafting-kissing-tumbling* case. It can be observed that the results obtained for both simulations agree quite well. With regard to a further verification of the presented algorithm and of its implementation, the predictions of AP2 are compared to other numerical predictions that can be found in the literature for the same setup, see Fig. 6b. The comparison is generally found to be good where the presented results are particularly consistent with the simulation results of Apte et al. [2] and Breugem [4]. At this point, one has to underline that all predicted evolutions in Fig. 6b rely on different FD method concepts.

In Fig. 7, the process of *drafting-kissing-tumbling* is displayed at eight different points in time. One observes that once the trailing particle is located within the gradually growing wake region of the leading particle, it experiences a lower drag force. Consequently, this results in a higher fall velocity for the trailing particle compared to the leading particle (drafting). With increasing time, the gap between both particles decreases due to their velocity difference, and thus the particles get—after a while—into contact (kissing). Clearly, this configuration is unstable. The particles tumble as a consequence and start to separate (tumbling). The flow phenomena observed in the experiment of Fortes et al. [15] are definitely reproduced by the presented computational approach.

6.3 *Particle-Laden Flow Through a Tube*

In this test case, a particle-laden flow through a tube with different cross sections is considered. It is assumed that the particles are allowed to adhere to each other and as


System
e3
e2
e1
$\overline{v} = 2\,\mathrm{cm/s}$
$\mathbf{v} = \begin{pmatrix} 0 \\ \overline{v} \\ 0 \end{pmatrix}$
'no-slip'-boundary
'do-nothing'-boundary
0.1
0.3
0.1
1.3
0.4
1.6
0.4
1.8
[cm]

Fluid: $\nu_F = 0.01\,\mathrm{cm^2/s}\,,\ \rho_F = 1.0\,\mathrm{g/cm^3}$

Particle: $R_{(1,2,3)} = (0.0150; 0.01625; 0.0175)\,\mathrm{cm}\,,\ \rho_P = 1.05\,\mathrm{g/cm^3}\,,\ \nu = 0.25\,,$
$E = 10^7\,\mathrm{N/cm^2}\,,\ \mu_{(G,H)} = 0.35\,,\ A = 3\cdot 10^{-5}\,\mathrm{s}\,,\ W_{(PP,PW)} = 10.0\,\mathrm{erg/cm^2}$

Fig. 8 Particle-laden flow through a tube. Geometry and material data

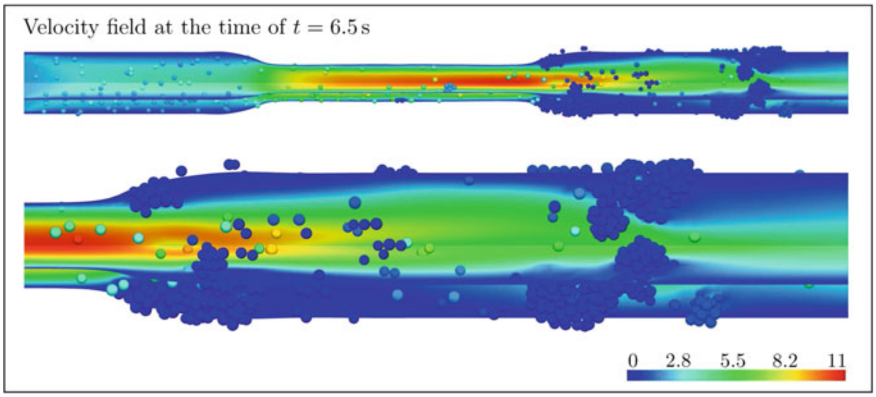


Fig. 9 Numerical results of the flow through the tube showing the velocity field at $t = 6.5$ [s]

well at the tube wall. Figure 8 shows the geometry of the tube including the model data, where the parameters W_{PP} and W_{PW} are the work of adhesion among particles and between particles and the tube wall, respectively. Gravitational effects are not considered. The suspended particles are randomly inserted at the inflow boundary with an initial velocity which is conform to the inflow velocity of the fluid. To compute the flow in the tube, a mesh consisting of 2,304,000 $\tilde{Q}_1/Q_0$ elements is used. This yields a discretized system with 2,304,000 pressure nodes and 6,953,280 velocity nodes. The time step sizes for the FEM and DEM solver are selected as $\Delta t = 10^{-4}$ s and $\Delta t_{\mathrm{D}} = 10^{-6}$ s, respectively.

The presence of van der Waals forces leads with the chosen parameters to a deposition of suspended particles onto the tube wall. The deposition grows with an increasing simulation time, in particular at the end of the smaller cross section

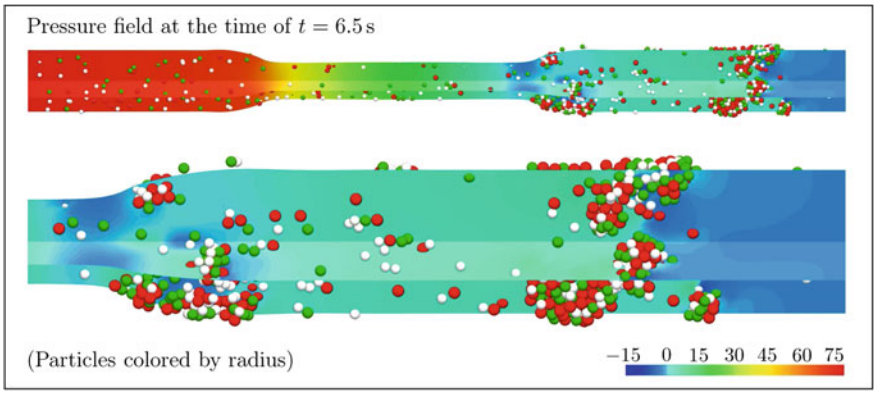

Pressure field at the time of $t = 6.5$ s
(Particles colored by radius)
−15 0 15 30 45 60 75

Fig. 10 Numerical results of the flow through the tube showing the pressure field at $t = 6.5$ [s]

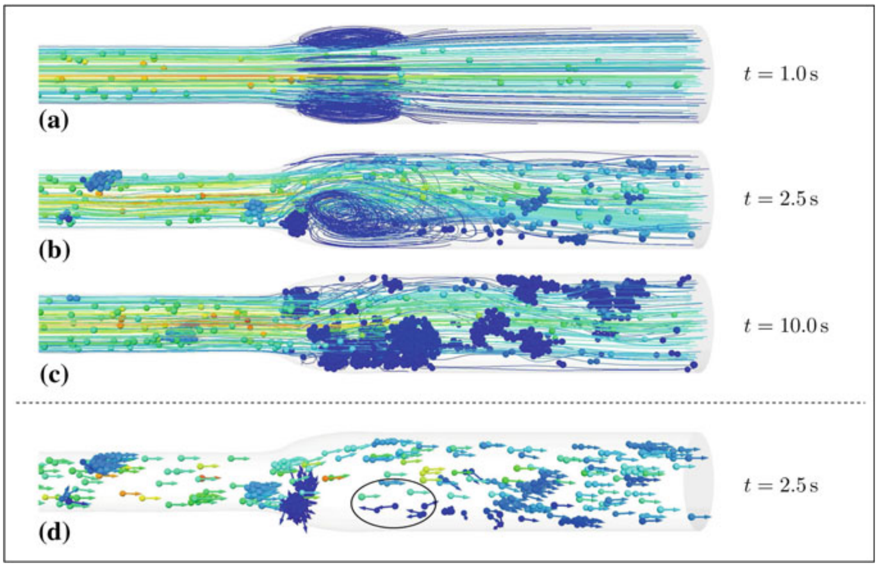


Fig. 11 Numerical results of the flow through the tube showing: **a**–**c** the streamlines and the particles at different points in time and **d** the particle velocity vectors at $t = 2.5$ s. The *colors* represent in all cases the velocity magnitudes

part. With more particles adhering to the wall, the velocity increases locally, and the particles experience higher drag forces. A situation like this is depicted in Fig. 9. Therein, one can see the influence of the developed agglomerates on the velocity field when they stick to the tube wall or slide along the wall. Due to the fully coupled description of the particulate flow, the strong mutual phase interactions, as in the situation shown in Fig. 9, can be captured by the developed DNS fluid-particle solver. The strong local impact of the dispersed phase on the fluid phase is here

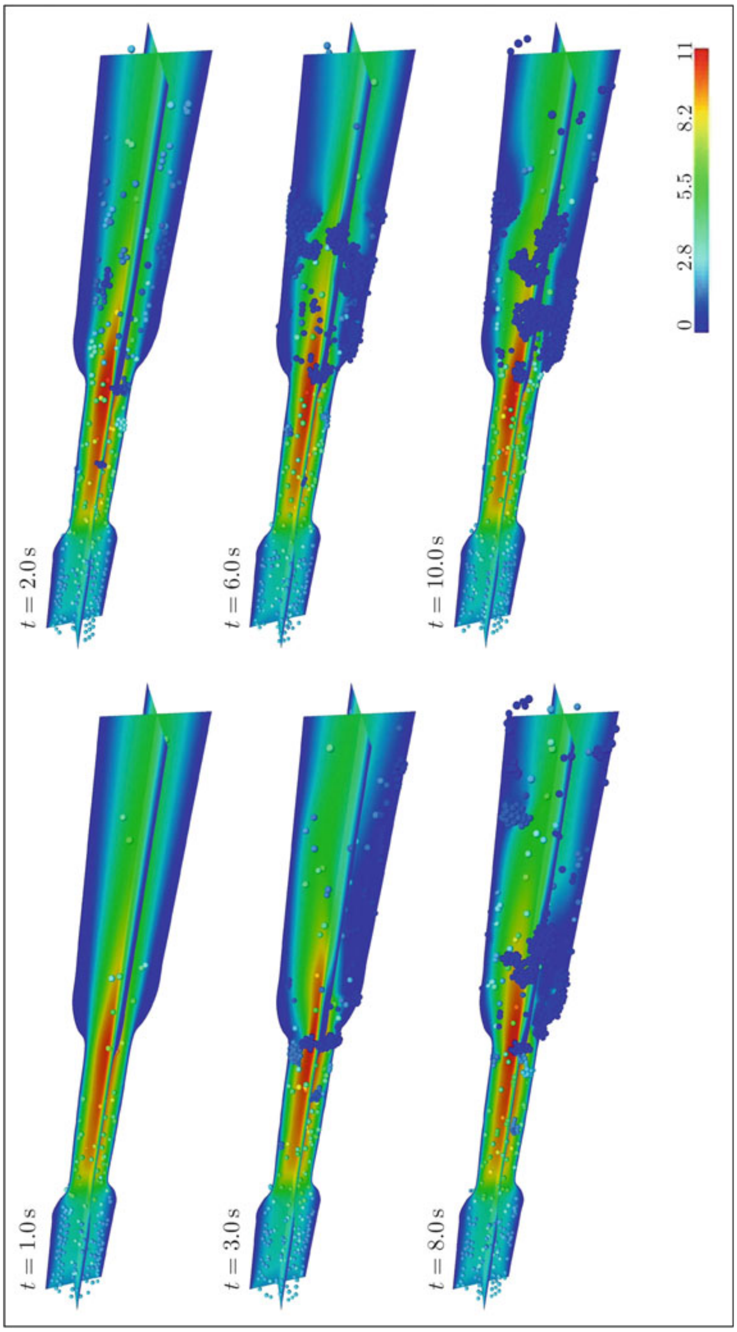


Fig. 12 Numerical results of the flow through the tube showing the fluid and particle velocities at six points in time

obvious. In Fig. 10, one can observe the corresponding effects regarding the pressure field (see the pressure difference when comparing the luv and the lee side of the large agglomerate). For the case when the traction forces exceed the adhesive forces, single particles and agglomerates break off and are subsequently transported away by the flow field. The complexity of the evolution of the multiphase flow situation at the outflow of the smaller cross section part is reflected in Fig. 11. There, the image (a), which shows the flow at time $t = 1.0$ s, illustrates a fully developed axisymmetric fluid flow with a certain eddy zone evolved in this region. Having a look at the images (b) and (c), $t = 2.5$ s and $t = 10.0$ s, one can see that this axial symmetry is totally lost and that a large number of particles has deposited on the wall. The image (d) shows the same flow situation as in (b), but in this case including particle velocity vectors and without streamlines. Here, the influence of the eddy on the trajectories of the passing particles can be clearly seen. In fact, the particles which are fully caught by the eddy change the flow direction so that they are subsequently transported against the main flow in the tube. The different phases of the flow process are shown for the whole system in Fig. 12.

7 Conclusion

In this work, a computational approach for the full 3D DNS of particulate flows is presented. The approach is based on the FD method. The developed solver treats the coupled fluid-particle problem in a staggered way by solving the phases explicitly in succession. The mutual phase interactions are computed on the other hand implicitly. As numerical solvers, the FEM and DEM are applied, respectively, to simulate the fluid and particle part. In the framework of the DEM, the particle collision is described using an adhesive viscoelastic model, and additionally, friction and rolling resistance are considered. To verify the reliability of the algorithm and its implementation, various test computations were performed. In this contribution, the simulation results of two sedimentation problems were presented and discussed. Furthermore, the solver was applied to simulate an aggregation dynamics problem of a particulate flow where the formation of agglomerates is considered. The chosen system for this task was a tube with different cross sections. The obtained numerical results show that the developed solver is appropriate to deal with fluid-particle interaction problems.

References

1. Allen MP, Tildesley DJ (1987) Computer simulation of liquids. Oxford University Press, New York
2. Apte SV, Martin M, Patankar NA (2009) A numerical method for fully resolved simulation (FRS) of rigid particle-flow interactions in complex flows. J Comput Phys 228(8):2712–2738
3. Avci B, Wriggers P (2012) A DEM-FEM coupling approach for the direct numerical simulation of 3D particulate flows. J Appl Mech 79:010901

4. Breugem WP (2012) A second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows. J Comput Phys 231(13):4469–4498
5. Brilliantov NV, Albers N, Spahn F, Pöschel T (2007) Collision dynamics of granular particles with adhesion. Phys Rev E 76(5, Part 1):051302
6. Brilliantov NV, Spahn F, Hertzsch JM, Pöschel T (1996) Model for collisions in granular gases. Phys Rev E 53(5, Part B):5382–5392
7. Brown PP, Lawler DF (2003) Sphere drag and settling velocity revisited. J Environ Eng 129(3):222–231
8. Cheng NS (2009) Comparison of formulas for drag coefficient and settling velocity of spherical particles. Powder Technol 189(3):395–398
9. Chokshi A, Tielens AGGM, Hollenbach D (1993) Dust coagulation. Astrophys J 407(2, Part 1):806–819
10. Clift R, Grace JR, Weber ME (1978) Bubbles, drops and particles. Academic Press, New York
11. Crowe CT (2006) Multiphase flow handbook. CRC Press, Boca Raton
12. Cundall PA, Strack ODL (1979) Discrete numerical model for granular assemblies. Geotechnique 29(1):47–65
13. Feng YT, Han K, Owen DRJ (2007) Coupled lattice Boltzmann method and discrete element modelling of particle transport in turbulent fluid flows: computational issues. Int J Numer Meth Eng 72(9):1111–1134
14. Feng ZG, Michaelides EE (2005) Proteus: a direct forcing method in the simulations of particulate flows. J Comput Phys 202(1):20–51
15. Fortes AF, Joseph DD, Lundgren TS (1987) Nonlinear mechanics of fluidization of beds of spherical-particles. J Fluid Mech 177:467–483
16. Glowinski R, Pan TW, Hesla TI, Joseph DD, Periaux J (2001) A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: application to particulate flow. J Comput Phys 169(2):363–426
17. Haeri S, Shrimpton JS (2012) On the application of immersed boundary, fictitious domain and body-conformal mesh methods to many particle multiphase flows. Int J Multiph Flow 40:38–55
18. Hertz H (1882) Über die berührung fester elastischer körper. Journal für die reine und angewandte Mathematik 92:156–171
19. Hu HH (1996) Direct simulation of flows of solid-liquid mixtures. Int J Multiph Flow 22(2):335–352
20. Hu HH, Joseph DD, Crochet MJ (1992) Direct simulation of fluid particle motions. Theor Comput Fluid Dyn 3:285–306
21. Hu HH, Patankar NA, Zhu MY (2001) Direct numerical simulations of fluid-solid systems using the Arbitrary Lagrangian-Eulerian technique. J Comput Phys 169(2):427–462
22. Iwashita K, Oda M (1998) Rolling resistance at contacts in simulation of shear band development by DEM. J Eng Mech 124(3):285–292
23. Johnson AA, Tezduyar TE (1996) Simulation of multiple spheres falling in a liquid-filled tube. Comput Methods Appl Mech Eng 134(3–4):351–373
24. Johnson AA, Tezduyar TE (1997) 3D simulation of fluid-particle interactions with the number of particles reaching 100. Comput Methods Appl Mech Eng 145(3–4):301–321
25. Johnson AA, Tezduyar TE (1999) Advanced mesh generation and update methods for 3D flow simulations. Comput Mech 23(2):130–143
26. Johnson AA, Tezduyar TE (2001) Methods for 3D computation of fluid-object interactions in spatially periodic flows. Comput Methods Appl Mech Eng 190(24–25):3201–3221
27. Johnson KL, Kendall K, Roberts AD (1971) Surface energy and contact of elastic solids. Proc R Soc Lond A 324(1558):301–313
28. Kruggel-Emden H, Wirtz S, Scherer V (2008) A study on tangential force laws applicable to the discrete element method (DEM) for materials with viscoelastic or plastic behavior. Chem Eng Sci 63(6):1523–1541
29. Kuhn MR, Bagi K (2004) Alternative definition of particle rolling in a granular assembly. J Eng Mech 130(7):826–835

30. Lebedev VI, Laikov DN (1999) Quadrature formula for the sphere of 131-th algebraic order of accuracy. Dokl Akad Nauk SSSR 366(6):741–745
31. Loskofsky C, Song F, Newby BZ (2006) Underwater adhesion measurements using the JKR technique. J Adhes 82(7):713–730
32. Luding S (2004) Micro-macro transition for anisotropic, frictional granular packings. Int J Solids Struct 41(21):5821–5836
33. Luding S (2008) Cohesive, frictional powders: contact models for tension. Granular Matter 10(4):235–246
34. Luo K, Wang Z, Fan J (2007) A modified immersed boundary method for simulations of fluid-particle interactions. Comput Methods Appl Mech Eng 197(1–4):36–46
35. Maugis D (1992) Adhesion of spheres—the JKR-DMT transition using a Dugdale model. J Colloid Interface Sci 150(1):243–269
36. Pöschel T, Schwager T (2005) Computational granular dynamics. Springer, New York
37. ten Cate A, Nieuwstad CH, Derksen JJ, van den Akker A (2002) Particle imaging velocimetry experiments and lattice-Boltzmann simulations on a single sphere settling under gravity. Phys Fluids 14(11):4012–4025
38. Turek S (1999) Efficient solvers for incompressible flow problems: an algorithmic and computational approach. Springer, Berlin
39. Turek S, Becker C (1998) FEATFLOW. Finite element software for the incompressible Navier-Stokes equations. Institute for Applied Mathematics, University of Heidelberg, Heidelberg
40. Veeramani C, Minev PD, Nandakumar K (2007) A fictitious domain formulation for flows with rigid particles: a non-lagrange multiplier version. J Comput Phys 224(2):867–879
41. Wachs A (2009) A DEM-DLM/FD method for direct numerical simulation of particulate flows: sedimentation of polygonal isometric particles in a Newtonian fluid with collisions. Comput Fluids 38(8):1608–1628
42. Wan D, Turek S (2006) Direct numerical simulation of particulate flow via multigrid FEM techniques and the fictitious boundary method. Int J Numer Meth Fluids 51(5):531–566
43. Wriggers P (2006) Computational contact mechanics, 2nd edn. Springer, Berlin

A Particle Finite Element Method (PFEM) for Coupled Thermal Analysis of Quasi and Fully Incompressible Flows and Fluid-Structure Interaction Problems

Eugenio Oñate, Alessandro Franci and Josep M. Carbonell

Abstract We present a Lagrangian formulation for coupled thermal analysis of quasi and fully incompressible flows and fluid-structure interaction (FSI) problems that has excellent mass preservation features. The success of the formulation lays on a residual-based stabilized expression of the mass balance equation obtained using the Finite Calculus (FIC) method. The governing equations are discretized with the FEM using simplicial elements with equal linear interpolation for the velocities, the pressure and the temperature. The merits of the formulation in terms of reduced mass loss and overall accuracy are verified in the solution of 2D and 3D adiabatic and thermally-coupled quasi-incompressible free-surface flows and FSI problems using the Particle Finite Element Method (PFEM). Examples include the sloshing of water in a tank and the falling of a water sphere and a cylinder into a tank containing water.

1 Introduction

The analysis of thermally coupled flows and their interaction with structures is relevant in many fields of engineering. In this work we present a Lagrangian numerical technique for solving this kind of problems for quasi and fully incompressible fluids using the Particle Finite Element Method (PFEM, www.cimne.com/pfem).

E. Oñate (✉) · A. Franci
Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE), Campus Norte UPC, 08034 Barcelona, Spain
e-mail: onate@cimne.upc.edu
http://www.cimne.com

A. Franci
e-mail: falessandro@cimne.upc.edu

J. M. Carbonell
Universitat Politècnica de Catalunya (UPC), 08034 Barcelona, Spain
e-mail: cpuigbo@cimne.upc.edu

The PFEM treats the mesh nodes in the analysis domain as particles which can freely move and even separate from the domain representing, for instance, the effect of water drops or cutting particles in drilling problems. A mesh connects the nodes discretizing the domain where the governing equations are solved using a stabilized FEM. Examples of application of PFEM to problems in fluid and solid mechanics including fluid-structure interaction (FSI) situations can be found in [4–6, 8–19, 28, 29, 33, 35, 36, 40–43]. Early attempts of the PFEM for solving thermally coupled flows were reported in [1, 2].

In Lagrangian analysis procedures (such as the PFEM) the motion of the fluid particles is tracked during the transient solution. Hence, the convective terms vanish in the momentum and heat transfer equations and no numerical stabilization is needed for treating those terms. Two other sources of mass loss, however, remain in the numerical solution of Lagrangian flows, i.e. that due to the treatment of the incompressibility constraint by a stabilized numerical method, and that induced by the inaccuracies in tracking the flow particles and, in particular, the free surface.

In this work the PFEM equations for analysis of thermally coupled flows and FSI problems are derived using the stabilized formulation based in the Finite Calculus (FIC) method proposed by Oñate et al. [20–27, 30–32, 37–39] that has excellent mass preservation features.

The lay-out of the paper is the following. In the next section we present the basic equations for conservation of linear momentum, mass and heat transfer for a quasi-incompressible fluid in a Lagrangian framework. A full incompressible fluid can be considered as a particular limit case of the former. Next we derive the stabilized FIC form of the mass balance equation. Then the finite element discretization using simplicial element with equal order approximation for the velocity, the pressure and the temperature is presented and the relevant matrices and vectors of the discretized problem are given. Details of the implicit solution of the Lagrangian FEM equations in time using a Newton iterative scheme are presented. The relevance of the bulk stiffness terms in the tangent matrix for enhancing the convergence and accuracy of the iterative solution scheme is discussed. The basic steps of the PFEM for solving FSI problems are described.

The efficiency and accuracy of the PFEM technique are verified by solving a set of adiabatic and thermally coupled quasi-incompressible free surface flow problems in two (2D) and three (3D) dimensions involving FSI situations. The adiabatic problems are the sloshing of water in a tank and the penetration of a water sphere into a cylindrical tank containing water. The thermally coupled problems considered are the extended 2D version of the adiabatic cases. The excellent performance of the numerical method proposed in terms of mass conservation and general accuracy is highlighted.

2 Governing Equations

We write the governing equations for a quasi-incompressible Newtonian flow problem in the Lagrangian description as follows [3, 46].

2.1 Momentum Equations

$$\rho \frac{Dv_i}{Dt} - \frac{\partial \sigma_{ij}}{\partial x_j} - b_i = 0, \quad i, j = 1, ..., n_s \quad \text{in } \Omega. \tag{1}$$

In Eq. (1), Ω is the analysis domain with boundary Γ, v_i and b_i are the velocity and body force components along the ith Cartesian axis, ρ is the density, n_s is the number of space dimensions (i.e. $n_s = 3$ for 3D problems) and σ_{ij} are the Cauchy stresses that are split in the deviatoric (s_{ij}) and pressure (p) components as

$$\sigma_{ij} = s_{ij} + p\delta_{ij} \tag{2}$$

where δ_{ij} is the Kronecker delta. Note that the pressure is assumed to be positive for a tension state. Summation of terms with repeated indices is assumed in Eq. (1) and in the following, unless otherwise specified.

The relationship between the deviatoric stresses s_{ij} and the strain rates ε_{ij} has the standard form for a Newtonian fluid,

$$s_{ij} = 2\mu \left(\varepsilon_{ij} - \frac{1}{3}\varepsilon_v \delta_{ij} \right) \quad \text{with} \quad \varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} \right) \tag{3}$$

where μ is the viscosity and ε_v is the volumetric strain rate defined as $\varepsilon_v = \varepsilon_{ii} = \frac{\partial v_i}{\partial x_i}$.

2.2 Mass Balance Equation

The standard mass balance equation for a quasi-incompressible fluid can be written as [3, 7, 46]

$$r_v = 0 \tag{4a}$$

with

$$r_v := -\frac{1}{c^2}\frac{Dp}{Dt} + \rho\varepsilon_v. \tag{4b}$$

In Eq. (4b) c is the speed of sound in the fluid. For a fully incompressible fluid $c = \infty$ and Eq. (4a) simplifies to the standard form, $\varepsilon_v = 0$. In our work we will retain the quasi-incompressible form of r_v of Eq. (4b) for convenience.

2.3 Thermal Balance

$$\rho c \frac{DT}{Dt} - \frac{\partial}{\partial x_i}\left(k \frac{\partial T}{\partial x_i}\right) - Q = 0 \quad i = 1, n_s \quad \text{in } \Omega \tag{5}$$

where T is the temperature, c is the thermal capacity, k is the heat conductivity and Q is the heat source.

2.4 Boundary Conditions

2.4.1 Mechanical Problem

The boundary conditions at the Dirichlet (Γ_v) and Neumann (Γ_t) boundaries with $\Gamma = \Gamma_v \cup \Gamma_t$ are

$$v_i - v_i^p = 0 \quad \text{on } \Gamma_v \tag{6}$$

$$\sigma_{ij} n_j - t_i^p = 0 \quad \text{on } \Gamma_t \quad i, j = 1, n_s \tag{7}$$

where v_i^p and t_i^p are the prescribed velocities and prescribed tractions on Γ_v and Γ_t, respectively and n_j are the components of the unit normal vector to the boundary [3, 7, 46].

2.5 Thermal Problem

$$\phi - \phi^p = 0 \quad \text{on } \Gamma_\phi \tag{8}$$

$$k \frac{\partial \phi}{\partial n} + q_n^p = 0 \quad \text{on } \Gamma_q \tag{9}$$

where ϕ^p and q_n^p are the prescribed temperature and the prescribed normal heat flux at the boundaries Γ_ϕ and Γ_q, respectively and n is the direction normal to the boundary.

Remark 1 The term $\frac{Dv_i}{Dt}$ in Eq. (1) is the *material derivative* of the ith velocity component v_i. This term is typically computed in a Lagrangian framework as

$$\frac{Dv_i}{Dt} = \frac{{}^{n+1}v_i - {}^{n}v_i}{\Delta t} \tag{10}$$

with

$${}^{n+1}v_i := v_i({}^{n+1}\mathbf{x}, {}^{n+1}t), \quad {}^{n}v_i := v_i({}^{n}\mathbf{x}, {}^{n}t) \tag{11}$$

where ${}^n v_i$ is the velocity of the material point that has the position ${}^n\mathbf{x}$ at time $t = {}^n t$, where $\mathbf{x}$ is the coordinates vector in a fixed Cartesian system [3, 7, 46].

3 Stabilized Mass Balance Equation

In this work we will use the *second order FIC form* of the mass balance equation in space for a quasi-incompressible fluid [37, 38], as well as the first order FIC form of the mass balance equation in time. These forms have the following expressions:

3.1 Second Order FIC Mass Balance Equation in Space

$$r_v + \frac{h_i^2}{12}\frac{\partial^2 r_v}{\partial x_i^2} = 0 \qquad \text{in } \Omega \qquad i = 1, ..., n_s. \tag{12a}$$

3.2 First Order FIC Mass Balance Equation in Time

$$r_v + \frac{\delta}{2}\frac{Dr_v}{Dt} = 0 \qquad \text{in } \Omega. \tag{12b}$$

Equation (12a) is obtained by expressing the balance of mass in a rectangular domain of finite size with dimensions $h_1 \times h_2$ (for 2D problems), where h_i are arbitrary distances, and retaining up to third order terms in the Taylor series expansions used for expressing the change of mass within the balance domain.

Equation (12b), on the other hand, is obtained by expressing the balance of mass in a space–time domain of infinitesimal length in space and finite dimension δ in time [20]. The derivation of Eqs. (12b) for a 1D problem are shown in [41].

The FIC terms in Eqs. (12b) play the role of space and time stabilization terms respectively. In the discretized problem, the space dimensions h_i and the time dimension δ are related to characteristic element dimensions and the time step increment, respectively as it will be explained later. Note that for $h_i = 0$ and $\delta = 0$ the standard infinitesimal form of the mass balance equation, $r_v = 0$, is obtained.

After some transformations the stabilized mass balance equation (12a) is written as [41]

$$-\frac{1}{\kappa}\frac{Dp}{Dt} + \varepsilon_v - \frac{\tau}{c^2}\frac{D^2 p}{Dt^2} + \tau\frac{\partial \hat{r}_{m_i}}{\partial x_i} = 0 \tag{13}$$

where $k = pc^2$ is the bulk modulus, τ is a *stabilization parameter* given by [41]

$$\tau = \left(\frac{8\mu}{h^2} + \frac{2\rho}{\delta}\right)^{-1} \tag{14}$$

and $\hat{r}_{m_i}$ is a *static momentum term* defined as

$$\hat{r}_{m_i} = \frac{\partial}{\partial x_j}(2\mu\varepsilon_{ij}) + \frac{\partial p}{\partial x_i} + b_i. \tag{15}$$

Equation (13) is used as the starting point for deriving the stabilized FEM formulation as explained in the following sections.

4 Variational Equations

4.1 Momentum Equations

Multiplying Eq. (1) by arbitrary test functions w_i with dimensions of velocity and integrating over the analysis domain Ω gives the weighted residual form of the momentum equations as [3, 7, 46]

$$\int_\Omega w_i \left(\rho \frac{Dv_i}{Dt} - \frac{\partial \sigma_{ij}}{\partial x_j} - b_i\right) d\Omega = 0. \tag{16}$$

Integrating by parts the term involving σ_{ij} and using the Neumann boundary conditions (7) yields the weak variational form of the momentum equations as

$$\int_\Omega w_i \rho \frac{Dv_i}{Dt} d\Omega + \int_\Omega \delta\varepsilon_{ij}\sigma_{ij} d\Omega - \int_\Omega w_i b_i d\Omega - \int_{\Gamma_t} w_i t_i^p d\Gamma = 0 \tag{17}$$

where $\delta\varepsilon_{ij} = \frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}$ is an arbitrary (virtual) strain rate field. Equation (17) is the standard form of the *principle of virtual power* [3, 7, 46].

Substituting the expression of the stresses from Eq. (2) into Eq. (17) gives

$$\int_\Omega w_i \rho \frac{Dv_i}{Dt} d\Omega + \int_\Omega \left[\delta\varepsilon_{ij} 2\mu \left(\varepsilon_{ij} - \frac{1}{3}\varepsilon_v \delta_{ij}\right) + \delta\varepsilon_v p\right] d\Omega - \int_\Omega w_i b_i d\Omega - \int_{\Gamma_t} w_i t_i^p d\Gamma = 0 \tag{18}$$

Equation (18) can be written in matrix form as

$$\boxed{\int_\Omega \mathbf{w}^T\rho\frac{D\mathbf{v}}{Dt}d\Omega+\int_\Omega \delta\boldsymbol{\varepsilon}^T\mathbf{D}\boldsymbol{\varepsilon}d\Omega+\int_\Omega \delta\boldsymbol{\varepsilon}^T\mathbf{m}p\,d\Omega-\int_\Omega \mathbf{w}^T\mathbf{b}d\Omega-\int_{\Gamma_t}\mathbf{w}^T\mathbf{t}^p d\Gamma=0.} \tag{19}$$

In Eq. (19) **w**, **v**, $\boldsymbol{\varepsilon}$ and $\delta\boldsymbol{\varepsilon}$ are vectors containing the test functions, the velocities, the strain rates and the virtual strain rates respectively; **b** and $\mathbf{t}^p$ are body force and surface traction vectors, respectively; **D** is the viscous constitutive matrix and **m** is an auxiliary vector. These vectors are defined as (for 3D problems)

$$\mathbf{w}=[w_1,w_2,w_3]^T,\quad \mathbf{v}=[v_1,v_2,v_3]^T,\quad \mathbf{b}=[b_1,b_2,b_3]^T,\quad \mathbf{t}^p=[t_1^p,t_2^p,t_3^p]^T$$

$$\boldsymbol{\varepsilon}=[\varepsilon_{11},\varepsilon_{22},\varepsilon_{33},\varepsilon_{12},\varepsilon_{13},\varepsilon_{23}]^T,\quad \delta\boldsymbol{\varepsilon}=[\delta\varepsilon_{11},\delta\varepsilon_{22},\delta\varepsilon_{33},\delta\varepsilon_{12},\delta\varepsilon_{13},\delta\varepsilon_{23}]^T$$

$$\mathbf{D}=\mu\begin{bmatrix}4/3 & -2/3 & -2/3 & 0 & 0 & 0\\ & 4/3 & -2/3 & 0 & 0 & 0\\ & & 4/3 & 0 & 0 & 0\\ & & & 1 & 0 & 0\\ \text{Sym.} & & & & 1 & 0\\ & & & & & 1\end{bmatrix},\quad \mathbf{m}=[1,1,1,0,0,0]^T. \tag{20}$$

4.2 Mass Balance Equations

We multiply Eq. (13) by arbitrary (continuous) test functions q (with dimensions of pressure) defined over the analysis domain Ω. Integrating over Ω gives

$$\int_\Omega -\frac{q}{\kappa}\frac{Dp}{Dt}d\Omega-\int_\Omega q\frac{\tau}{c^2}\frac{D^2p}{Dt^2}d\Omega+\int_\Omega q\varepsilon_v d\Omega+\int_\Omega q\tau\frac{\partial\hat{r}_{m_i}}{\partial x_i}d\Omega=0. \tag{21}$$

Integrating by parts the last integral in Eq. (21) and using Eq. (15) gives after some transformations [39, 40]

$$\begin{aligned}\int_\Omega \frac{q}{\kappa}\frac{Dp}{Dt}d\Omega+\int_\Omega q\frac{\tau}{c^2}\frac{D^2p}{Dt^2}d\Omega-\int_\Omega q\varepsilon_v d\Omega+\int_\Omega \tau\frac{\partial q}{\partial x_i}\left(\frac{\partial}{\partial x_i}(2\mu\varepsilon_{ij})+\frac{\partial p}{\partial x_i}+b_i\right)d\Omega\\ -\int_{\Gamma_t} q\tau\left[\rho\frac{Dv_n}{Dt}-\frac{2}{h_n}\left(2\mu\frac{\partial v_n}{\partial n}+p-t_n\right)\right]d\Gamma=0.\end{aligned} \tag{22}$$

Expression (22) holds for 2D and 3D problems. The terms involving the first and second material time derivative of the pressure and the boundary term in Eq. (22)

are important for preserving the conservation of mass in free-surface flow problems [10, 41].

4.3 Thermal Balance Equation

Application of the standard weighted residual method to the heat balance Eqs. (5) and (9) leads, after standard operations, to [7, 44]

$$\int_{\Omega} \hat{w}\rho c \frac{\partial T}{\partial t} d\Omega + \int_{\Omega} \frac{\partial \hat{w}}{\partial x_i} k \frac{\partial T}{\partial x_i} d\Omega - \int_{\Omega} \hat{w} Q d\Omega + \int_{\Gamma_q} \hat{w} q_n^p d\Gamma = 0 \tag{23}$$

where $\hat{w}$ are the space weighting functions for the temperature.

5 FEM Discretization

We discretize the analysis domain into finite elements with n nodes in the standard manner leading to a mesh with a total number of N_e elements and N nodes. In our work we will choose simple 3-noded linear triangles ($n = 3$) for 2D problems and 4-noded tetrahedra ($n = 4$) for 3D problems with local linear shape functions N_i^e defined for each node i ($i = 1, n$) of element e [34, 44]. The velocity components, the pressure and the temperature are interpolated over the mesh in terms of their nodal values in the same manner using the global linear shape functions N_j spanning over the elements sharing node j ($j = 1, N$) [34, 44–46]. In matrix form

$$\mathbf{v} = \mathbf{N}_v \bar{\mathbf{v}}\ ,\ \ p = \mathbf{N}_p \bar{\mathbf{p}}\quad ,\quad T = \mathbf{N}_T \bar{\mathbf{T}} \tag{24}$$

where

$$\bar{\mathbf{v}} = \begin{Bmatrix} \bar{\mathbf{v}}^1 \\ \bar{\mathbf{v}}^2 \\ \vdots \\ \bar{\mathbf{v}}^N \end{Bmatrix} \ \text{with}\ \bar{\mathbf{v}}^i = \begin{Bmatrix} \bar{v}_1^i \\ \bar{v}_2^i \\ \bar{v}_3^i \end{Bmatrix}\ ,\ \bar{\mathbf{p}} = \begin{Bmatrix} \bar{p}^1 \\ \bar{p}^2 \\ \vdots \\ \bar{p}^N \end{Bmatrix}\ ,\ \mathbf{T} = \begin{Bmatrix} \bar{T}_1 \\ \bar{T}_2 \\ \vdots \\ \bar{T}^N \end{Bmatrix} \\ \mathbf{N}_v = [\mathbf{N}_1, \mathbf{N}_2, \cdots, \mathbf{N}_N]^T\ ,\ \mathbf{N}_p = \mathbf{N}_T = [N_1, N_2, \cdots, N_N] \tag{25}$$

with $\mathbf{N}_j = N_j \mathbf{I}_n$ where $\mathbf{I}_n$ is the $n \times n$ unit matrix.

In Eq. (25) vectors $\bar{\mathbf{v}}$, $\bar{\mathbf{p}}$ and $\bar{\mathbf{T}}$ contain the nodal velocities, the nodal pressures and the nodal temperatures for the whole mesh, respectively and the upperindex denotes the nodal value for each vector or scalar magnitude.

Substituting Eq. (24) into Eqs. (17), (22) and (23) and choosing a Galerkin formulation with $w_i = q = \hat{w}_i = N_i$ leads to the following system of algebraic equations

$$\mathbf{M}_0\dot{\bar{\mathbf{v}}} + \mathbf{K}\bar{\mathbf{v}} + \mathbf{Q}\bar{\mathbf{p}} - \mathbf{f}_v = \mathbf{0} \tag{26a}$$

$$\mathbf{M}_1\dot{\bar{\mathbf{p}}} + \mathbf{M}_2\ddot{\bar{\mathbf{p}}} - \mathbf{Q}^T\bar{\mathbf{v}} + (\mathbf{L} + \mathbf{M}_b)\bar{\mathbf{p}} - \mathbf{f}_p = \mathbf{0} \tag{26b}$$

$$\mathbf{C}\dot{\bar{\mathbf{T}}} + \hat{\mathbf{L}}\bar{\mathbf{T}} - \mathbf{f}_T = \mathbf{0} \tag{26c}$$

where $\dot{\bar{\mathbf{a}}}$ and $\ddot{\bar{\mathbf{a}}}$ denote the first and second material time derivatives of the components of a vector $\mathbf{a}$. The different matrices and vectors in Eqs. (26a) are assembled from the element contributions given in Box 1.

$$\mathbf{M}^e_{0_{ij}} = \int_{\Omega^e} \rho N^e_i N_j \mathbf{I}_3 d\Omega \ , \ \mathbf{K}^e_{ij} = \int_{\Omega^e} \mathbf{B}^{eT}_i \mathbf{D}\mathbf{B}^e_j d\Omega \ , \ \mathbf{Q}^e_{ij} = \int_{\Omega^e} \mathbf{B}^{eT}_i \mathbf{m} N^e_j d\Omega$$

$$M^e_{1_{ij}} = \int_{\Omega^e} \frac{1}{\kappa} N^e_i N^e_j d\Omega \ , \ M^e_{2_{ij}} = \int_{\Omega^e} \frac{\tau}{c^2} N^e_i N^e_j d\Omega \ , \ M^e_{b_{ij}} = \int_{\Gamma_t} \frac{2\tau}{h_n} N^e_i N^e_j d\Gamma$$

$$L^e_{ij} = \int_{\Omega^e} \tau(\boldsymbol{\nabla}^T N^e_i)\boldsymbol{\nabla} N^e_j d\Omega \ , \ \mathbf{f}^e_{v_i} = \int_{\Omega^e} \mathbf{N}^e_i \mathbf{b} d\Omega + \int_{\Gamma_t} \mathbf{N}^e_i \mathbf{t}^p d\Gamma$$

$$f^e_{p_i} = \int_{\Gamma_t} \tau N^e_i \left[\rho \frac{Dv_n}{Dt} - \frac{2}{h_n}(2\mu\varepsilon_n - t_n)\right] d\Gamma - \int_{\Omega^e} \tau \boldsymbol{\nabla}^T N^e_i \mathbf{b} d\Omega$$

$$C^e_{ij} = \int_{\Omega^e} \rho c N^e_i N^e_j d\Omega \ , \ \hat{L}^e_{ij} = \int_{\Omega^e} k(\boldsymbol{\nabla}^T N^e_i)\boldsymbol{\nabla} N^e_j d\Omega \ , \ f^e_{T_i} = \int_{\Omega^e} N^e_i Q d\Omega - \int_{\Gamma_q} N^e_i q^p_n d\Gamma$$

with $i, j = 1, n$.
For 3D problems

$$\mathbf{B}^e_i = \begin{bmatrix} \frac{\partial N^e_i}{\partial x_1} & 0 & 0 \\ 0 & \frac{\partial N^e_i}{\partial x_2} & 0 \\ 0 & 0 & \frac{\partial N^e_i}{\partial x_3} \\ \frac{\partial N^e_i}{\partial x_2} & \frac{\partial N^e_i}{\partial x_1} & 0 \\ \frac{\partial N^e_i}{\partial x_3} & 0 & \frac{\partial N^e_i}{\partial x_1} \\ 0 & \frac{\partial N^e_i}{\partial x_3} & \frac{\partial N^e_i}{\partial x_2} \end{bmatrix} \ , \ \mathbf{N}^e_i = N^e_i \mathbf{I}_3 \quad \text{and} \quad \boldsymbol{\nabla} = \begin{Bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_3} \end{Bmatrix}$$

N^e_i: Local shape function of node i of element e [34, 44]

Box 1. Element form of the matrices and vectors in Eqs.(26)

Remark 2 The boundary terms of vector $\mathbf{f}_p$ can be incorporated in the matrices of Eq. (26b). This leads to a non symmetrical set of equations. These boundary terms are computed here iteratively within the incremental solution scheme.

Remark 3 The presence of matrix $\mathbf{M}_b$ in Eq. (26b) allows us to compute the pressure without the need of prescribing its value at the free surface. This eliminates the error introduced when the pressure is prescribed to zero in free boundaries, which leads to considerable mass losses in viscous flows [15].

Remark 4 The stabilization parameter τ of Eq. (14) is computed for each element e using $h = l^e$ and $\delta = \Delta t$ as

$$\tau = \left(\frac{8\mu}{(l^e)^2} + \frac{2\rho}{\Delta t} \right)^{-1} \tag{27}$$

where Δt is the time step used for the transient solution and l^e is a characteristic element length computed as $l^e = 2(\Omega^e)^{1/n_s}$ where Ω^e is the element area (for 3-noded triangles) or volume (for 4-noded tetrahedra). For fluids with heterogeneous material the values of μ and ρ are computed at the element center.

The characteristic boundary length h_n in the expression of $\mathbf{f}_p$ (Box 1) has been taken equal to l^e in our computations.

6 Transient Solution of the Discretized Equations

Equations (26a), (26b), (26c) are solved in time with an implicit Newton-Raphson type iterative scheme [3, 7, 44, 46]. The basic steps within a time interval $[n, n+1]$ are:

- Initialize variables: $({}^{n+1}\mathbf{x}^1, {}^{n+1}\bar{\mathbf{v}}^1, {}^{n+1}\bar{\mathbf{p}}^1, {}^{n+1}\bar{\mathbf{T}}^1, {}^{n+1}\bar{\mathbf{r}}_m^1) \equiv \{{}^n\mathbf{x}, {}^n\bar{\mathbf{v}}, {}^n\bar{\mathbf{p}}, {}^n\bar{\mathbf{T}}, {}^n\bar{\mathbf{r}}_m\}$.
- Iteration loop: $i = 1, NITER$.
  For each iteration.

Step 1. Compute the nodal velocity increments $\Delta\bar{\mathbf{v}}$
From Eq. (26a), we deduce

$${}^{n+1}\mathbf{H}_v^i \Delta\bar{\mathbf{v}} = -{}^{n+1}\bar{\mathbf{r}}_m^i \rightarrow \Delta\bar{\mathbf{v}} \tag{28a}$$

with the momentum residual $\bar{\mathbf{r}}_m$ and the iteration matrix $\mathbf{H}_v$ given by

$$\bar{\mathbf{r}}_m = \mathbf{M}_0\dot{\bar{\mathbf{v}}} + \mathbf{K}\bar{\mathbf{v}} + \mathbf{Q}\bar{\mathbf{p}} - \mathbf{f}_v, \quad \mathbf{H}_v = \frac{1}{\Delta t}\mathbf{M}_0 + \mathbf{K} + \mathbf{K}_v \tag{28b}$$

Step 2. Update the nodal velocities

$${}^{n+1}\bar{\mathbf{v}}^{i+1} = {}^{n+1}\bar{\mathbf{v}}^i + \Delta\bar{\mathbf{v}} \tag{29}$$

Step 3. Compute the nodal pressures ${}^{n+1}\bar{\mathbf{p}}^{i+1}$

From Eq. (26b) we obtain

$$
{}^{n+1}\mathbf{H}_p^i\,{}^{n+1}\bar{\mathbf{p}}^{i+1} = \frac{1}{\Delta t}\mathbf{M}_1\,{}^{n+1}\bar{\mathbf{p}}^i + \frac{1}{\Delta t^2}\mathbf{M}_2(2\,{}^n\bar{\mathbf{p}} - {}^{n-1}\bar{\mathbf{p}}) + \mathbf{Q}^T\,{}^{n+1}\bar{\mathbf{v}}^{i+1} + {}^{n+1}\bar{\mathbf{f}}_p^i \rightarrow {}^{n+1}\bar{\mathbf{p}}^{i+1} \tag{30a}
$$

with

$$
\mathbf{H}_p = \frac{1}{\Delta t}\mathbf{M}_1 + \frac{1}{\Delta t^2}\mathbf{M}_2 + \mathbf{L} + \mathbf{M}_b \tag{30b}
$$

Step 4. Update the nodal coordinates

$$
{}^{n+1}\mathbf{x}^{i+1} = {}^{n+1}\mathbf{x}^i + \frac{1}{2}({}^{n+1}\bar{\mathbf{v}}^{i+1} + {}^n\bar{\mathbf{v}})\Delta t \tag{31}
$$

A more accurate expression for computing ${}^{n+1}\mathbf{x}^{i+1}$ can be used involving the nodal accelerations [40].

Step 5.Compute the nodal temperatures

From Eq. (26c) we obtain

$$
\left[\frac{1}{\Delta t}\mathbf{C} + \hat{\mathbf{L}}\right]\Delta\bar{\mathbf{T}} = -{}^{n+1}\bar{\mathbf{r}}_T^i \quad , \quad {}^{n+1}\bar{\mathbf{T}}^{i+1} = {}^n\bar{\mathbf{T}}^i + \Delta\bar{\mathbf{T}} \tag{32}
$$

with

$$
\mathbf{r}_T = \mathbf{C}\dot{\bar{\mathbf{T}}} + \hat{\mathbf{L}}\bar{\mathbf{T}} - \mathbf{f}_T \tag{33}
$$

Step 6. Check convergence

Verify the following conditions:

$$
\begin{aligned}
\|{}^{n+1}\bar{\mathbf{v}}^{i+1} - {}^{n+1}\bar{\mathbf{v}}^i\| &\le e_v\|{}^n\bar{\mathbf{v}}\| \\
\|{}^{n+1}\bar{\mathbf{p}}^{i+1} - {}^{n+1}\bar{\mathbf{p}}^i\| &\le e_p\|{}^n\bar{\mathbf{p}}\| \\
\|{}^{n+1}\bar{\mathbf{T}}^{i+1} - {}^{n+1}\bar{\mathbf{T}}^i\| &\le e_T\|{}^n\bar{\mathbf{T}}\|
\end{aligned} \tag{34}
$$

where e_v, e_p and e_T are prescribed error norms. In our examples we have set $e_v = e_p = e_T = 10^{-3}$.

If conditions (34) are satisfied then make $n \leftarrow n+1$ and proceed to the next time step. Otherwise, make the iteration counter $i \leftarrow i+1$ and repeat Steps 1–5.

Remark 5 In Eqs. (28a, 28b)–(34) ${}^{n+1}(\cdot)$ denotes the values of a matrix or a vector computed using the nodal unknowns at time $n+1$. In this work the derivatives and integrals in all the matrices and the residual vectors $\bar{\mathbf{r}}_m$ and $\bar{\mathbf{r}}_T$ are computed on the *discretized geometry at time n* while the nodal force vectors $\mathbf{f}_v$, $\mathbf{f}_p$ and $\mathbf{f}_v$ are computed on the current configuration at time $n+1$. This is equivalent to using an *updated Lagrangian formulation* [3, 45, 46].

Remark 6 Including the bulk stiffness matrix $\mathbf{K}_v$ in $\mathbf{H}_v$ has proven to be essential for the fast convergence, mass preservation and overall accuracy of the iterative solution [10, 41]. The element expression of $\mathbf{K}_v$ can be obtained as [41]

$$\mathbf{K}_v^e = \int_{\Omega^e} \mathbf{B}^T \mathbf{m}\theta\Delta t\kappa \mathbf{m}^T \mathbf{B} d\Omega \tag{35}$$

where θ is a positive number such that $0 < \theta \leq 1$ that has the role of preventing the ill-conditioning of the iteration matrix $\mathbf{H}_v$ for highly incompressible fluids. An adequate selection of θ also improves the overall accuracy of the numerical solution and the preservation of mass for large time steps [10]. For fully incompressible fluids (c and $\kappa = \infty$), a finite value of κ is used in practice in $\mathbf{K}_v$ as this helps to obtaining an accurate solution for velocities and pressure with reduced mass loss in few iterations per time step [10]. These considerations, however, do not affect the value of κ within matrix $\mathbf{M}_1$ in Eq. (26b) that vanishes for a fully incompressible fluid. Clearly, the value of the terms of $\mathbf{K}_v^e$ can also be limited by reducing the time step size. This, however, leads to an increase in the cost of the computations. A similar approach for improving mass conservation in incompressible flows was proposed in [42].

Remark 7 The iteration matrix $\mathbf{H}_v$ in Eq. (28a) is an *approximation of the exact tangent matrix* in the updated Lagrangian formulation for a quasi-incompressible fluid [40]. The simplified form of $\mathbf{H}_v$ used in this work has yielded good results with convergence achieved for the nodal velocities, the pressure and the temperature in 3–4 iterations in all the problems analyzed.

Remark 8 The time step within a time interval $[n, n+1]$ is chosen as $\Delta t = \min\left(\frac{{}^n l^e_{\min}}{|{}^n\mathbf{v}|_{\max}}, \Delta t_b\right)$ where ${}^n l^e_{\min}$ is the minimum characteristic distance of all elements in the mesh, with l^e computed as explained in Remark 4, $|{}^n\mathbf{v}|_{\max}$ is the maximum value of the modulus of the velocity of all nodes in the mesh and Δt_b is the critical time step of all nodes approaching a solid boundary defined as $\Delta t_b = \min\left(\frac{{}^n l_b}{|{}^n\mathbf{v}_b|_{\max}}\right)$ where ${}^n l_b$ is the distance from the node to the boundary and ${}^n\mathbf{v}_b$ is the velocity of the node. This definition of Δt intends that no node crosses a solid boundary during a time step.

A method that allows using large time steps in the integration of the PFEM equations can be found in [16].

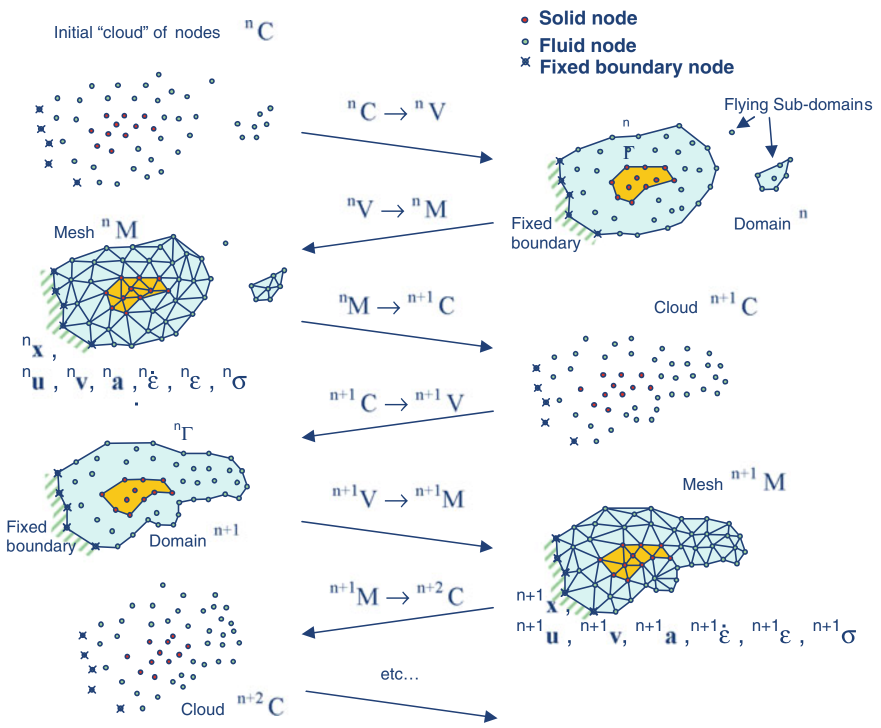

Initial "cloud" of nodes ${}^nC$
● Solid node
○ Fluid node
✱ Fixed boundary node
${}^nC \rightarrow {}^nV$
Flying Sub-domains
${}^n\Gamma$
Fixed boundary
Domain ^{n}
${}^nV \rightarrow {}^nM$
Mesh ${}^nM$
${}^nM \rightarrow {}^{n+1}C$
Cloud ${}^{n+1}C$
${}^n\mathbf{x}$, ${}^n\mathbf{u}$, ${}^n\mathbf{v}$, ${}^n\mathbf{a}$, ${}^n\dot{\varepsilon}$, ${}^n\varepsilon$, ${}^n\sigma$
${}^{n+1}C \rightarrow {}^{n+1}V$
${}^n\Gamma$
Fixed boundary
Domain ${}^{n+1}$
${}^{n+1}V \rightarrow {}^{n+1}M$
Mesh ${}^{n+1}M$
${}^{n+1}M \rightarrow {}^{n+2}C$
${}^{n+1}\mathbf{x}$, ${}^{n+1}\mathbf{u}$, ${}^{n+1}\mathbf{v}$, ${}^{n+1}\mathbf{a}$, ${}^{n+1}\dot{\varepsilon}$, ${}^{n+1}\varepsilon$, ${}^{n+1}\sigma$
etc…
Cloud ${}^{n+2}C$

Fig. 1 Sequence of steps to update a "cloud" of nodes representing a domain containing a fluid and a solid part from time n $(t =^n t)$ to time $n + 2$ $(t =^n t + 2\Delta t)$

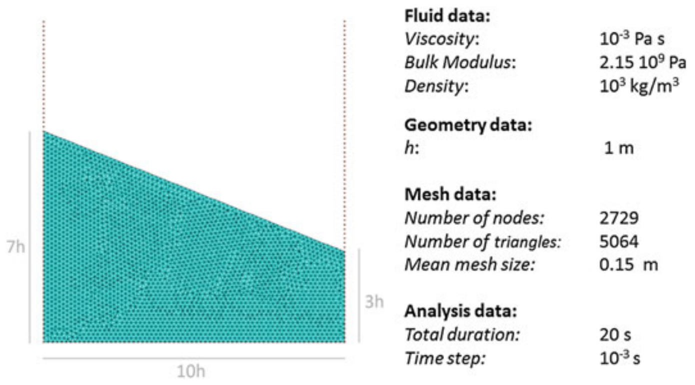
7 About the Particle Finite Element Method (PFEM)

7.1 The Basis of the PFEM

Let us consider a domain V containing fluid and solid subdomains. Each subdomain is characterized by a set of points, hereafter termed *particles*. The particles contain all the information for defining the geometry and the material and mechanical properties of the underlying subdomain. In the PFEM both subdomains are modelled using an *updated Lagrangian formulation* [3, 45].

The solution steps within a time step in the PFEM are as follows:

1. The starting point at each time step is the cloud of points C in the fluid and solid domains. For instance ${}^nC$ denotes the cloud at time $t = {}^nt$ (Fig. 1).
2. Identify the boundaries defining the analysis domain ${}^nV$, as well as the subdomains in the fluid and the solid. This is an essential step as some boundaries (such as the free surface in fluids) may be severely distorted during the solution, including separation and re-entering of nodes. The Alpha Shape method [8] is



Fluid data:	
Viscosity:	10^{-3} Pa s
Bulk Modulus:	2.15 10^{9} Pa
Density:	10^{3} kg/m^{3}
Geometry data:	
h:	1 m
Mesh data:	
Number of nodes:	2729
Number of triangles:	5064
Mean mesh size:	0.15 m
Analysis data:	
Total duration:	20 s
Time step:	10^{-3} s

Fig. 2 2D analysis of sloshing of water in rectangular tank. Initial geometry, analysis data and mesh of 5064 3-noded triangles discretizing the water in the tank

used for the boundary definition. Clearly, the accuracy in the reconstruction of the boundaries depends on the number of points in the vicinity of each boundary and on the Alpha Shape parameter. In the problems solved in this work the Alpha Shape method has been implementation as described in [12, 28].

3. Discretize the the analysis domain ${}^nV$ with a finite element mesh ${}^nM$. We use an efficient mesh generation scheme based on an enhanced Delaunay tesselation [11, 12].
4. Solve the Lagrangian equations of motion for the overall continuum using the standard FEM. Compute the state variables in at the next (updated) configuration for ${}^nt+\Delta t$: velocities, pressure and viscous stresses in the fluid and displacements, stresses and strains in the solid.
5. Move the mesh nodes to a new position ${}^{n+1}C$ where n+1 denotes the time ${}^nt+\Delta t$, in terms of the time increment size.
6. Go back to step 1 and repeat the solution for the next time step to obtain ${}^{n+2}C$.

Note that the key differences between the PFEM and the classical FEM are the remeshing technique and the identification of the domain boundary at each time step.

The CPU time required for meshing grows linearly with the number of nodes. As a general rule, meshing consumes for 3D problems around 15 % of the total CPU time per time step, while the solution of the equations (with typically 3 iterations per time step) and the system assembly consume approximately 70 % and 15 % of the CPU time per time step, respectively. These figures refer to analyses in a single processor Pentium IV PC [36]. Considerable speed can be gained using parallel computing techniques.

In this work we will apply the PFEM to problems involving a rigid domain containing fluid particles only. Application of the PFEM in fluid and solid mechanics and in fluid-structure interaction problems can be found in [4–6, 8–19, 28, 29, 33, 35, 36, 40–43], as well as in www.cimne.com/pfem.

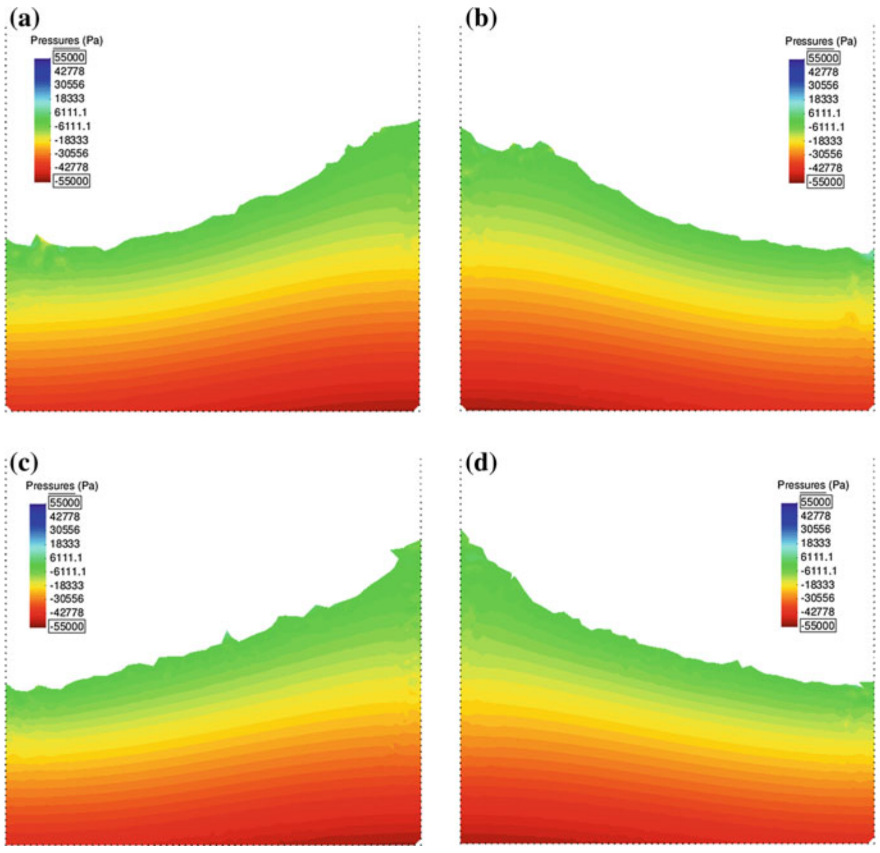


Fig. 3 2D sloshing of water in rectangular tank. Snapshots of water geometry at two different times ($\theta = 1$). Colours indicate pressure contours. **a** t = 5.7 s; **b** t = 7.4 s; **c** t = 13.3 s; **d** t = 18.6 s

8 Examples

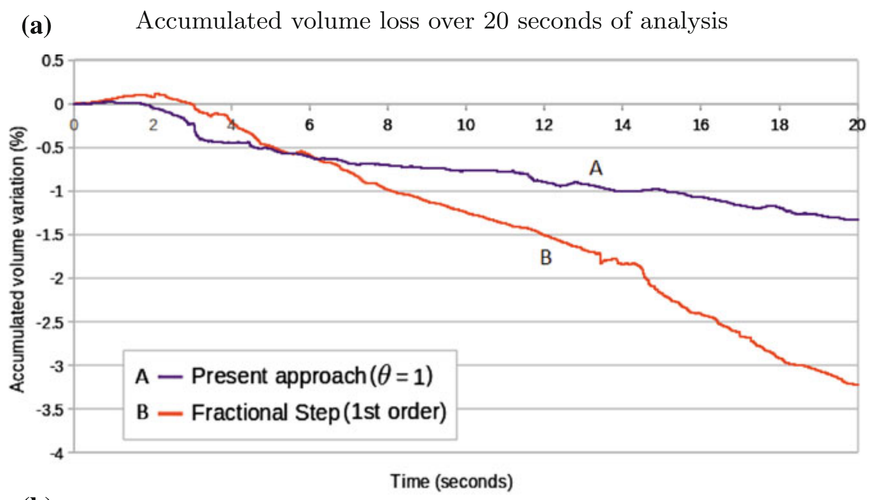
8.1 Sloshing of Water in Prismatic Tank

The problem has been solved first in 2D. Figure 2 shows the analysis data. The fluid oscillates due to the hydrostatic forces induced by its original position.

The problem has been run using different values of the parameter θ in the tangent bulk stiffness matrix $\mathbf{K}_v^e$ (Eq. 35). The first set of results (Figs. 3 and 4) were obtained with $\theta = 1$. The problem was then solved for $\theta = 0.08$, thereby, reducing in one order the magnitude the diagonal terms in $\mathbf{K}_v^e$.

Figure 3 shows snapshots of the water geometry at different times. Pressure contours are superposed to the deformed geometry of the fluid in the figures.

(a)



(b)

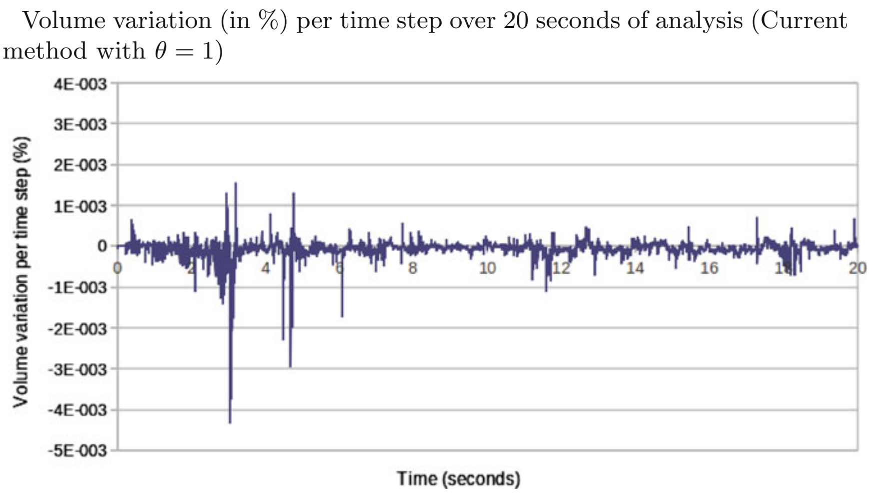


Fig. 4 2D sloshing of water in rectangular tank. **a** Time evolution of the percentage of water volume loss due to the numerical algorithm. **b** Average volume variation per time step. Current method. 1.09×10^{-4} %. Fractional step: 2.07×10^{-4} %

Figure 4 shows the evolution of the percentage of water volume (i.e. mass) loss introduced by the numerical solution scheme. The accumulated volume loss (in percentage versus the initial volume) for the method proposed with $\theta = 1$ is approximately 1.33 % over 20 s of simulation time (Fig. 4a). The average volume variation in absolute value per time step is 1.09×10^{-4} % (Fig. 4b). The total water volume loss is the sum of the losses induced by the numerical scheme and the losses due to the updating of the free surface using the PFEM. No correction of mass was introduced

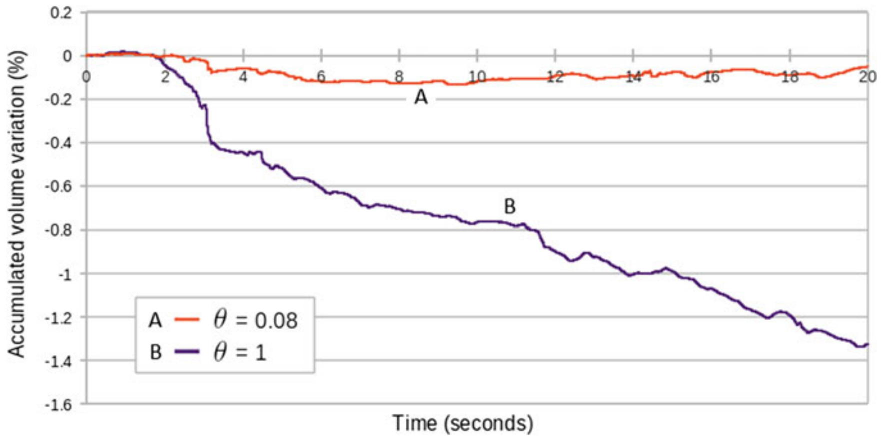


Fig. 5 2D sloshing of water in rectangular tank. Time evolution of percentage of water volume loss obtained using the current method with $\theta = 0.08$ (*curve A*) and $\theta = 1$ (*curve B*) $\Delta t = 10^{-3}$ s

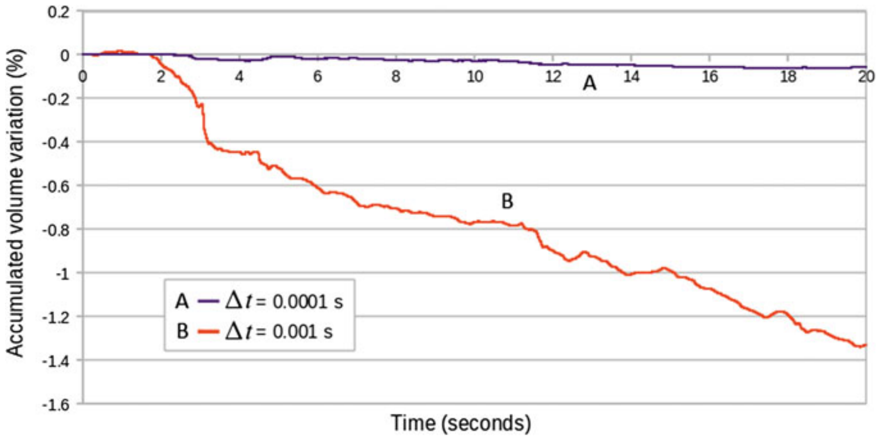


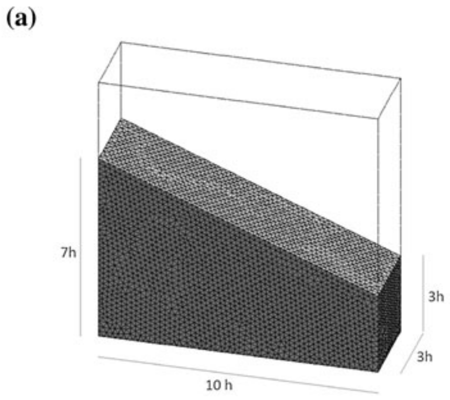
Fig. 6 2D sloshing of water in rectangular tank. Time evolution of percentage of water volume loss obtained with the current method. *Curve A* $\theta = 1$ and $\Delta t = 10^{-4}$ s. *Curve B* $\theta = 1$ and $\Delta t = 10^{-3}$ s

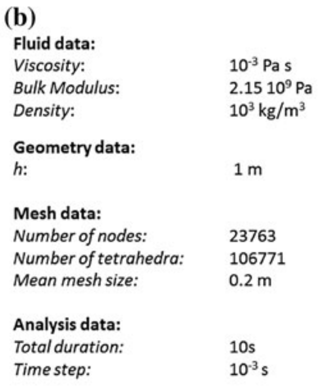
at the end of each time step. Taking all this into account, the fluid volume loss over the analysis period is remarkably low.

The volume losses induced by the free surface updating can be reduced using a finer mesh in that region in conjunction with an enhanced alpha shape technique.

The total fluid volume loss can be reduced to almost zero by introducing a small correction in the free surface at the end of each time step [41].

The fluid volume losses obtained using a standard first order fractional step method [41] and the PFEM are shown in Fig. 4a for comparison. Clearly the method proposed




(b)

Fluid data:	
Viscosity:	10^{-3} Pa s
Bulk Modulus:	$2.15\ 10^{9}$ Pa
Density:	10^{3} kg/m^{3}
Geometry data:	
h:	1 m
Mesh data:	
Number of nodes:	23763
Number of tetrahedra:	106771
Mean mesh size:	0.2 m
Analysis data:	
Total duration:	10s
Time step:	10^{-3} s

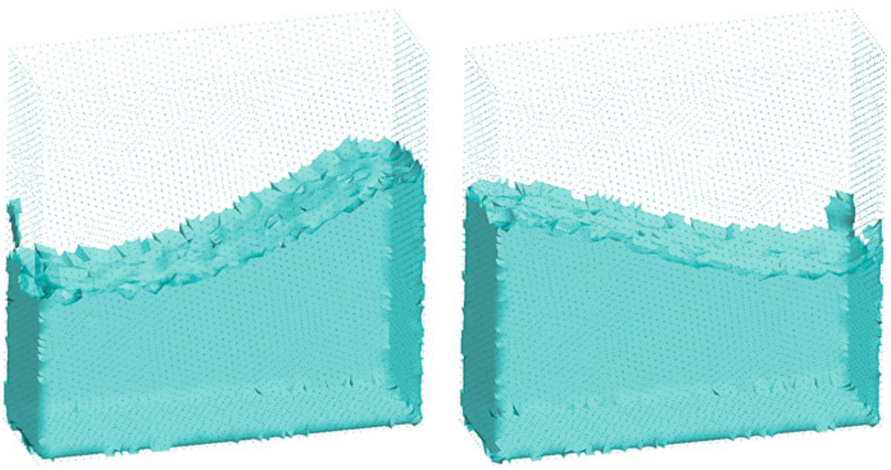


Fig. 7 3D analysis of sloshing of water in prismatic tank ($\theta = 1$). Analysis data and snapshots of water geometry at **a** $t = 5.7$ s (*left*) and **b** $t = 7.4$ s (*right*)

in this work leads to a reduction in the overall fluid volume loss, as well as in the volume loss per time step.

Figure 5 shows a comparison between the fluid volume loss for $\theta = 1$ and $\theta = 0.08$ using the same time step in both cases ($\Delta t = 10^{-3}$ s). Results show that the reduction of the tangent bulk stiffness matrix terms leads to an improvement in the preservation of the initial volume of the fluid. It is noted that the convergence of the iterative solution for $\theta = 0.08$ was the same as for $\theta = 1$.

Figure 6 shows that a similar improvement in the volume preservation can be obtained using $\theta = 1$ and reducing the time step to $\Delta t = 10^{-4}$ s. This, however, increases the cost of the computations.

These results indicate that accurate numerical results with reduced volume losses can be obtained by appropriately adjusting the parameter θ in the tangent bulk modulus matrix while keeping the time step size to competitive values in terms of CPU

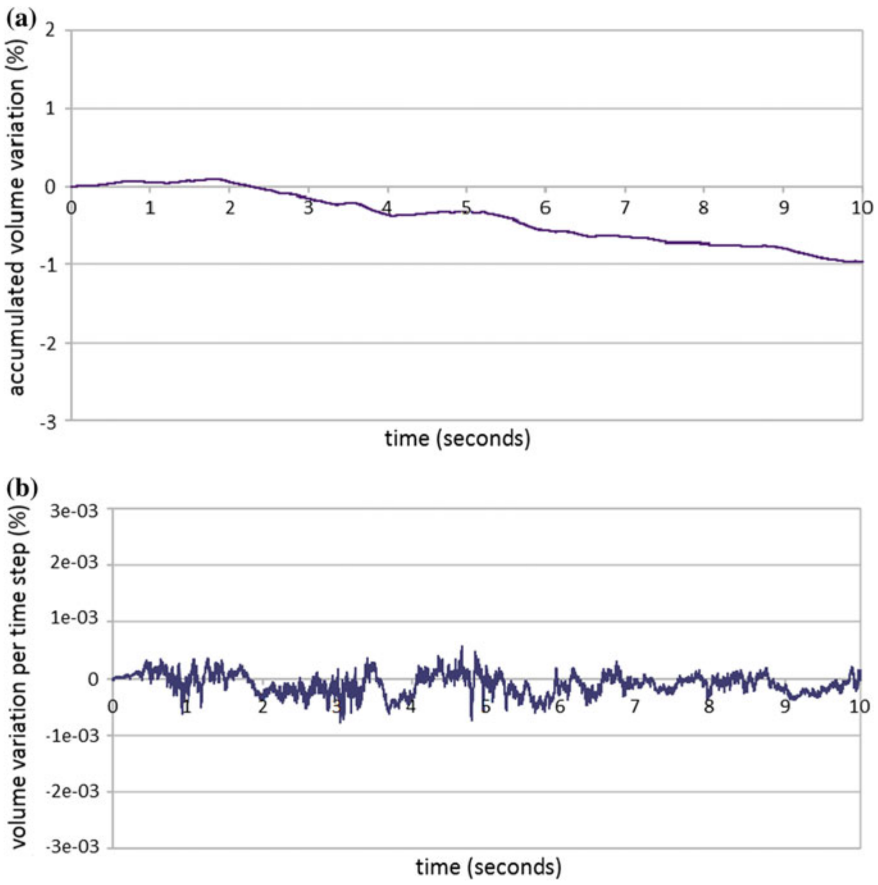
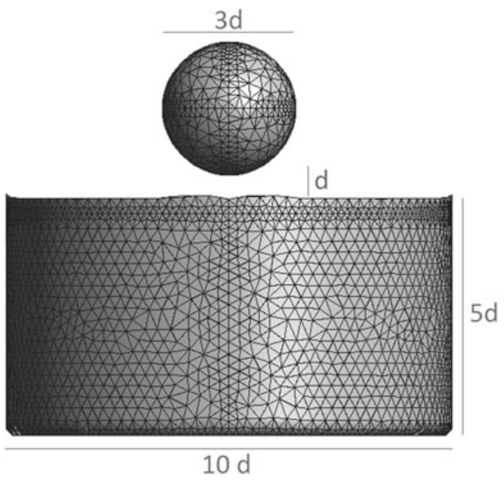


Fig. 8 3D analysis of sloshing of water in prismatic tank ($\theta = 1$). **a** Time evolution of accumulated water volume loss (in %) due to the numerical algorithm. **b** Volume loss (in %) per time step over 2 s of analysis. Average volume loss per time step: 1.64×10^{-4} %

cost. A study of the influence of θ in the numerical solution for quasi-incompressible free surface fluids in terms of volume preservation and overall accuracy using the formulation here presented can be found in [10].

More results for this example can be found in [41].

Figures 7 and 8 show a similar set of results for the 3D analysis of the same sloshing problem using a relative coarse initial mesh of 106771 4-noded tetrahedra and $\theta = 1$. It is remarkable that the percentage of total fluid volume loss due to the numerical scheme after 10 s of analysis is approximately 1 %.



Fluid data:	
Viscosity:	10^{-3} Pa s
Bulk Modulus:	2.15 10^9 Pa
Density:	10^3 kg/m^3
Geometry data:	
d:	0.1 m
Mesh data:	
Number of nodes:	16935
Number of tetrahedra:	88892
Mean mesh size:	0.03 m
Analysis data:	
Total duration:	3s
Time step:	2 10^{-4} s

Fig. 9 Water sphere falling in a tank filled with water. Analysis data and initial discretization of the sphere and the water in the tank with 88892 4-noded tetrahedra

8.2 *Falling of a Water Sphere in a Cylindrical Tank Containing Water*

This example is the 3D analysis of the impact of a sphere made of water as it falls in a cylindrical tank containing water. Both the water in the sphere and in the tank mix in a single fluid after the impact. Figure 9 shows the material and analysis data and the initial discretization of the sphere and the water in the tank in 88892 4-noded tetrahedra. The problem was solved with the new stabilized method presented in the paper with $\theta = 1$. Figure 10 shows snapshots of the mixing process at different times. An average of four iterations for convergence of the velocity and the pressure were needed during all the steps of the analysis. The total water mass lost in the sphere and the tank due to the numerical algorithm was $\simeq 2\,\%$ after 3 s of analysis (Fig. 11a).

8.3 *Sloshing of a Fluid in a Heated Tank*

A fluid at initial temperature $T = 20\,°\text{C}$ oscillates due to the hydrostatic forces induced by its initial position in a rectangular tank heated to a uniform and constant temperature of $T = 75\,°\text{C}$. The geometry and the problem data of the 2D simulation are shown in Fig. 12. The fluid, with a very high thermal conductivity, changes its temperature only due to the contact with the hotter tank walls. The heat flux along the free surface has been considered null. The fluid domain has been initially discretized with 2828 3-noded triangles. The coupled thermal-fluid dynamics simulation has been run for 100 s using a time step increment of $\Delta t = 0.005$ s.

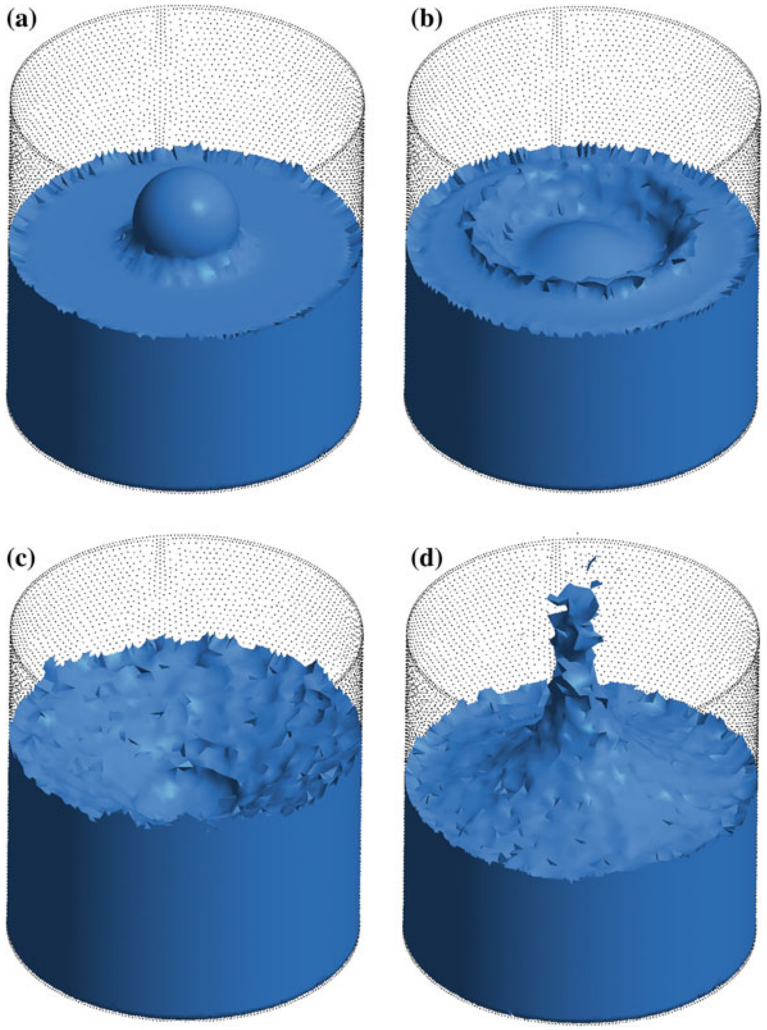


Fig. 10 Water sphere falling in tank containing water. Evolution of the impact and mixing of the two liquids at different times. Results for $\theta = 1$. **a** $t = 0.175$ s, **b** $t = 0.275$ s, **c** $t = 0.5$ s, **d** $t = 0.9$ s

Figure 13 shows some snapshots of the numerical simulation. The temperature contours have been superposed on the fluid domain at the different time instants.

In Fig. 14 the evolution of temperature with time at the points A, B and C of Figure 12 is plotted. The coordinates of these sample points are (0.1 m, 0.1 m), (0.5 m, 0.4 m) and (0.9 m, 0.1 m), respectively. Figures 13 and 14 show that the fluid does not heat uniformly because of the convection effect automatically captured by the Lagrangian technique here presented.

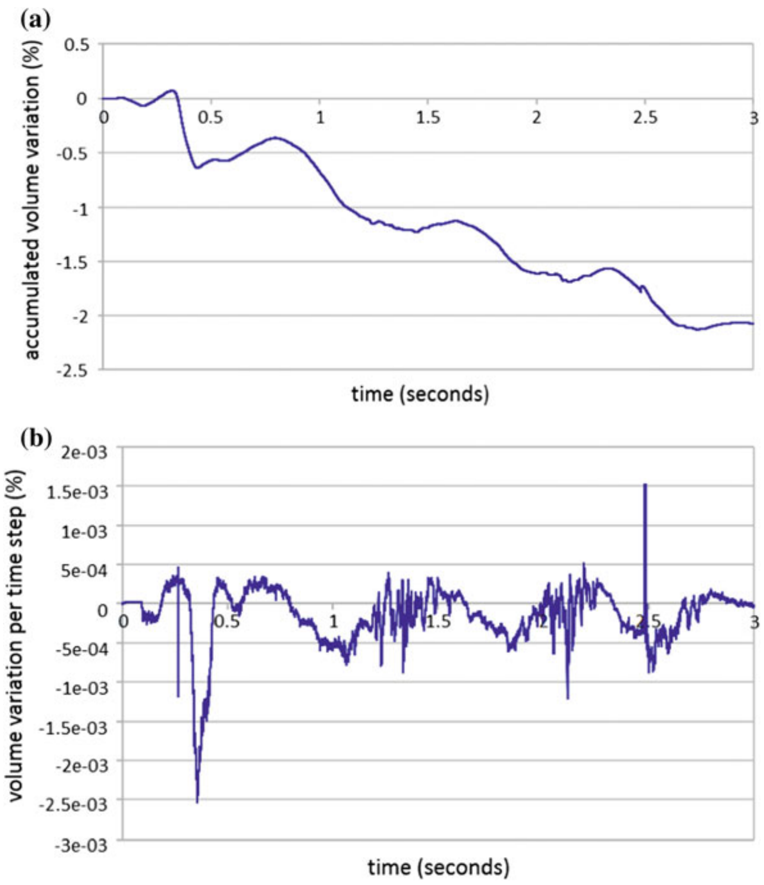
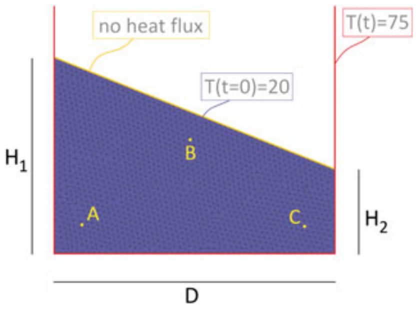


Fig. 11 Water sphere falling in a tank containing water. **a** Accumulated volume over three seconds of analysis due to the numerical algorithm ($\theta = 1$). **b** Volume loss (in %) per time step. Average volume variation per time step: 2.54×10^{-4} %

8.4 *Falling of a Cylindrical Object in a Heated Tank Filled with Fluid*

An elastic object falls in a tank containing a fluid at rest. The tank walls are maintained at temperature $T = 75\,^\circ$C during the whole analysis, while the fluid and the solid object have an initial temperature of $T = 20\,^\circ$C. The geometry and the problem data of the 2D simulation as well the thermal initial and boundary conditions, are shown in Fig. 15. Both the fluid and the solid have a high thermal conductivity. The heat flux along the fluid and solid surfaces in contact with the air has been considered null. The fluid and the solid domains have been discretized with 1986 and 108 3-noded



Material data
Viscosity: $10^{-2} Pa \cdot s$
Bulk modulus: $10^{9} Pa$
Density: $10^{3}\ kg/m^{3}$
Conductivity: $2 \cdot 10^{3}\ W/(m \cdot °C)$
Thermal capacity: $4 \cdot 10^{3}\ J/(kg \cdot °C)$
Geometry data
H_1: $0.7\ m$
H_2: $0.3\ m$
D: $1\ m$
Average mesh size: $0.\ 02\ m$
Analysis data
Total duration: $100\ s$
Time step increment: $0.005\ s$

Fig. 12 2D sloshing of a fluid in a heated tank. Initial geometry, problem data, thermal boundary and initial conditions

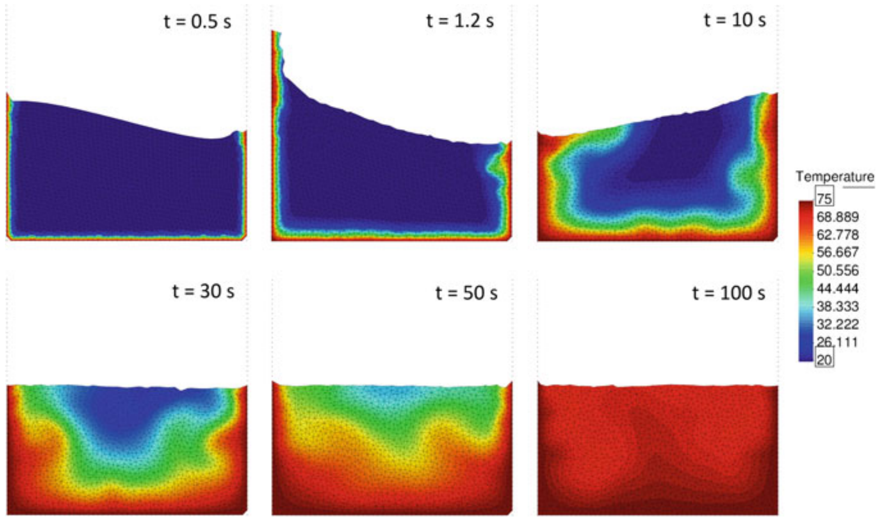


Fig. 13 2D sloshing of a fluid in a heated tank. Snapshots of fluid geometry at six different times. Colours indicate temperature contours

triangular finite elements, respectively. The duration of the simulation is 10 s and the time step increment chosen is $\Delta t = 0.0005$ s.

Figure 16 collects some representative snapshots of the numerical simulation with the temperature results plotted over the fluid and the solid domains.

The graph of Figure 17 is the evolution of temperature at the central point of the solid object. As expected, its temperature tends to $T = 75\,°$C.

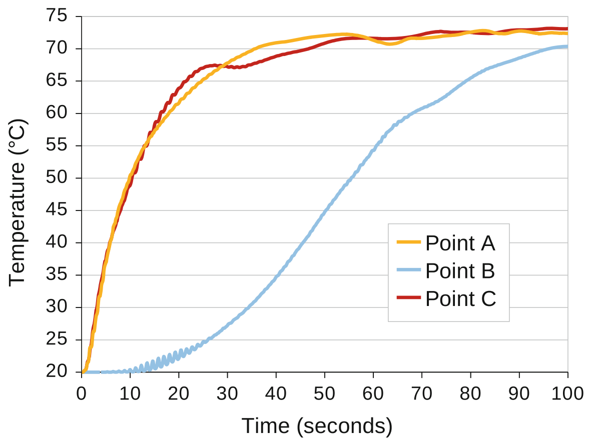
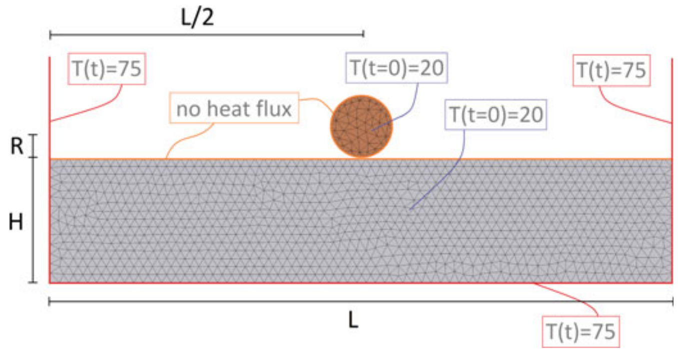


Fig. 14 2D sloshing of a fluid in a heated tank. Evolution of temperature with time at the points A, B and C of Fig. 12



Fluid data
Viscosity: $10^{-3} Pa \cdot s$
Density: $10^{3} \frac{kg}{m^3}$
Bulk modulus: $10^{9} Pa$
Conductivity: $5\ 10^{3} \frac{W}{m \cdot °C}$
Thermal capacity: $4\ 10^{3} \frac{J}{kg \cdot °C}$

Solid data
Young modulus: $10^{6} Pa$
Density: $5\ 10^{2} \frac{kg}{m^3}$
Poisson coefficient: 0.33
Conductivity: $10^{4} \frac{W}{m \cdot °C}$
Thermal capacity: $4.9\ 10^{2} \frac{J}{kg \cdot °C}$

Analysis data
Total duration: $10\ s$
Time step increment: $5\ 10^{-4} s$
Geometry data
H: $0.1\ m$
L: $0.5\ m$
R: $0.025\ m$
Mean mesh size: $7.5\ 10^{-3} m$

Fig. 15 Falling of a solid object in a heated tank filled with fluid. Initial geometry, problem data, thermal boundary and initial conditions

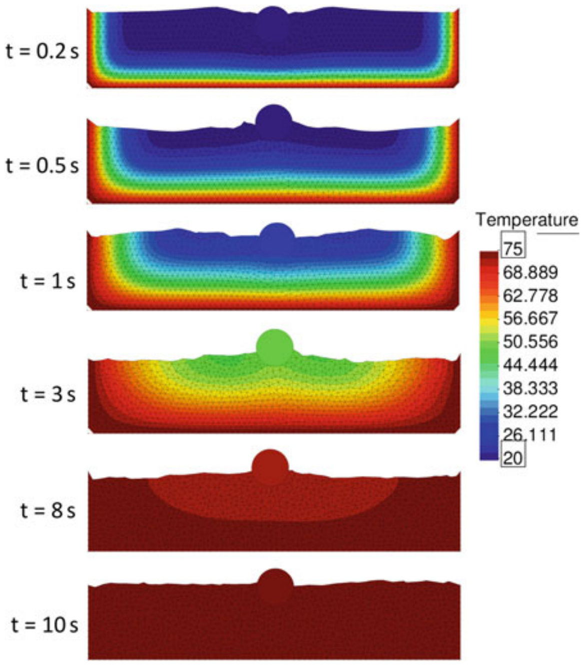


Fig. 16 Falling of a solid object in a heated tank filled with fluid. Snapshots at six different times. Colours indicate temperature contours

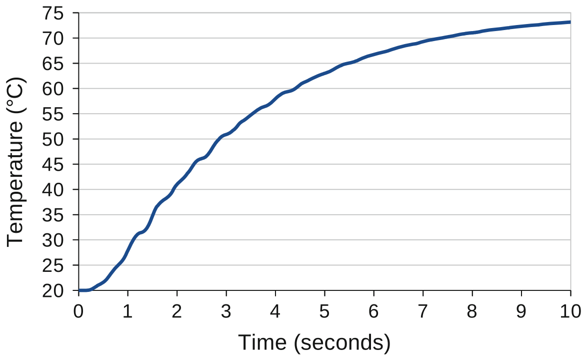


Fig. 17 Falling of a solid object in a heated tank filled with fluid. Time evolution of the temperature at the center of the solid object

9 Concluding Remarks

We have presented a FIC-based stabilized Lagrangian finite element method for thermal-mechanical analysis of quasi and fully incompressible flows and FSI problems that has excellent mass preservation properties. The method has been successfully applied to the adiabatic and thermal-mechanical coupled analysis of free-surface quasi-incompressible flows including FSI situations using the PFEM and an updated Lagrangian formulation. These problems are more demanding in terms of the mass preservation features of the numerical algorithm. The method proposed has yielded excellent results for 2D and 3D adiabatic and thermally-coupled free surface flow problems involving surface waves, water splashing, violent impact of flows with containment walls and FSI situations.

Acknowledgments This research was partially supported by the Advanced Grant project SAFECON of the European Research Council.

References

1. Aubry R, Idelsohn SR, Oñate E (2005) Particle finite element method in fluid-mechanics including thermal convection-diffusion. Comput Sruct 83(17–18):1459–1475
2. Aubry R, Idelsohn SR, Oñate E (2006) Fractional step like schemes for free surface problems with thermal coupling using the Lagrangian PFEM. Comput Mech 38(4–5):294–309
3. Belytschko T, Liu WK, Moran B (2013) Non linear finite element for continua and structures, 2nd edn. Wiley, New York
4. Carbonell JM, Oñate E, Suárez B (2010) Modeling of ground excavation with the particle finite element method. J Eng Mech (ASCE) 136(4):455–463
5. Carbonell JM, Oñate E (2013) Suárez B (2013) Modelling of tunnelling processes and cutting tool wear with the Particle Finite Element Method (PFEM). Comput Mech 52:607–629. doi:10.1007/s00466-013-0835-x (Accepted)
6. Cremonesi M, Frangi A, Perego U (2011) A Lagrangian finite element approach for the simulation of water-waves induced by landslides. Comput Struct 89:1086–1093
7. Donea J, Huerta A (2003) Finite element method for flow problems. Wiley, Chichester
8. Edelsbrunner H, Mucke EP (1999) Three dimensional alpha shapes. ACM Trans Graphics 13:43–72
9. Felippa F, Oñate E (2007) Nodally exact Ritz discretizations of 1D diffusion-absorption and Helmholtz equations by variational FIC and modified equation methods. Comput Mech 39:91–111
10. Franci A, Oñate E, Carbonell JM (2013) On the effect of the tangent bulk stiffness matrix in the analysis of free surface Lagrangian flows using PFEM. Research Report CIMNE PI402. Int J Numer Meth Biomed Eng 38(2):125–138 (Submitted)
11. Idelsohn SR, Calvo N, Oñate E (2003c) Polyhedrization of an arbitrary point set. Comput Meth Appl Mech Eng 192(22–24):2649–2668
12. Idelsohn SR, Oñate E, Del Pin F (2004) The particle finite element method: a powerful tool to solve incompressible flows with free-surfaces and breaking waves. Int J Numer Meth Biomed Eng 61(7):964–989
13. Idelsohn SR, Marti J, Limache A, Oñate E (2008) Unified Lagrangian formulation for elastic solids and incompressible fluids: Application to fluid-structure interaction problems via the PFEM. Comput Meth Appl Mech Eng 197:1762–1776

14. Idelsohn SR, Mier-Torrecilla M, Oñate E (2009) Multi-Fluid flows with the particle finite element method. Comput Meth Appl Mech Eng 198:2750–2767
15. Idelsohn SR, Oñate E (2010) The challenge of mass conservation in the solution of free-surface flows with the fractional-step method: problems and solutions. Int J Numer Meth Biomed Eng 26:1313–1330
16. Idelsohn SR, Nigro N, Limache A, Oñate E (2012) Large time-step explicit integration method for solving problem with dominant convection. Comput Meth Appl Mech Eng 217–220:168–185
17. Larese A, Rossi R, Oñate E, Idelsohn SR (2008) Validation of the particle finite element method (PFEM) for simulation of free surface flows. Eng Comput 25(4):385–425
18. Limache A, Idelsohn SR, Rossi R, Oñate E (2007) The violation of objectivity in Laplace formulation of the Navier-Stokes equations. Int J Numer Meth Fluids 54:639–664
19. Oliver X, Cante JC, Weyler R, González C, Hernández J (2007) Particle finite element methods in solid mechanics problems. In: Oñate E, Owen R (eds) Computational Plasticity. Springer, Berlin, pp 87–103
20. Oñate E (1998) Derivation of stabilized equations for advective-diffusive transport and fluid flow problems. Comput Meth Appl Mech Eng 151:233–267
21. Oñate E, Manzan M (1999) A general procedure for deriving stabilized space-time finite element methods for advective-diffusive problems. Int J Numer Meth Fluids 31:203–221
22. Oñate E (2000) A stabilized finite element method for incompressible viscous flows using a finite increment calculus formulation. Comput Meth Appl Mech Eng 182(1–2):355–370
23. Oñate E, García J (2001) A finite element method for fluid-structure interaction with surface waves using a finite calculus formulation. Comput Meth Appl Mech Eng 191:635–660
24. Oñate E (2003) Multiscale computational analysis in mechanics using finite calculus: an introduction. Comput Meth Appl Mech Eng 192(28–30):3043–3059
25. Oñate E, Taylor RL, Zienkiewicz OC, Rojek J (2003) A residual correction method based on finite calculus. Eng Comput 20:629–658
26. Oñate E (2004) Possibilities of finite calculus in computational mechanics. Int J Num Meth Eng 60(1):255–281
27. Oñate E, Rojek J, Taylor R, Zienkiewicz O (2004a) Finite calculus formulation for incompressible solids using linear triangles and tetrahedra. Int J Num Meth Eng 59(11):1473–1500
28. Oñate E, Idelsohn SR, Del Pin F, Aubry R (2004b) The particle finite element method. An overview. Int J Comput Meth 1(2):267–307
29. Oñate E, Celigueta MA (2006a) Modeling bed erosion in free surface flows by the particle finite element method. Acta Geotech 1(4):237–252
30. Oñate E, Valls A, García J (2006b) FIC/FEM formulation with matrix stabilizing terms for incompressible flows at low and high Reynold's numbers. Comput Mech 38(4–5):440–455
31. Oñate E, García J, Idelsohn SR, Del Pin F (2006c) FIC formulations for finite element analysis of incompressible flows. Eulerian, ALE and Lagrangian approaches. Comput Meth Appl Mech Eng 195(23–24):3001–3037
32. Oñate E, Valls A, García J (2007) Computation of turbulent flows using a finite calculus-finite element formulation. Int J Numer Meth Eng 54:609–637
33. Oñate E, Idelsohn SR, Celigueta MA, Rossi R (2008) Advances in the particle finite element method for the analysis of fluid-multibody interaction and bed erosion in free surface flows. Comput Meth Appl Mech Eng 197(19–20):1777–1800
34. Oñate E (2009) Structural analysis with the finite element method. Linear statics, vol 1. Basis and solids. Springer (CIMNE)
35. Oñate E, Rossi R, Idelsohn SR, Butler K (2010) Melting and spread of polymers in fire with the particle finite element method. Int J Numer Meth Eng 81(8):1046–1072
36. Oñate E, Celigueta MA, Idelsohn SR, Salazar F, Suárez B (2011) Possibilities of the particle finite element method for fluid-soil-structure interaction problems. Comput Mech 48(3):307–318
37. Oñate E, Nadukandi P, Idelsohn SR, García J, Felippa C (2011) A family of residual-based stabilized finite element methods for Stokes flows. Int J Num Meth Fluids 65(1–3):106–134

38. Oñate E, Idelsohn SR, Felippa C (2011) Consistent pressure Laplacian stabilization for incompressible continua via higher-order finite calculus. Int J Numer Meth Eng 87(1–5):171–195
39. Oñate E, Nadukandi P, Idelsohn SR (2014) P1/P0+ elements for incompressible flows with discontinuous material properties. Comput Meth Appl Mech Eng 271:185–209
40. Oñate E, Carbonell JM (2013) Updated Lagrangian finite element formulation for quasi-incompressible fluids. Research Report PI393 (CIMNE). Submitted to Comput Mech
41. Oñate E, Franci A, Carbonell JM (2013) Lagrangian formulation for finite element analysis of quasi-incompressible fluids with reduced mass losses. Int J Numer Meth Fluids doi:10.1002/fld.3870
42. Ryzhakov P, Oñate E, Rossi R, Idelsohn SR (2012) Improving mass conservation in simulation of incompressible flows. Int J Numer Meth Eng 90(12):1435–1451
43. Tang B, Li JF, Wang TS (2009) Some improvements on free surface simulation by the particle finite element method. Int J Num Meth Fluids 60(9):1032–1054
44. Zienkiewicz OC, Taylor RL, Zhu JZ (2005) The finite element method. The basis, 6th edn. Elsevier, Oxford
45. Zienkiewicz OC, Taylor RL (2005) The finite element method for solid and structural mechanics, 6th edn. Elsevier, Oxford
46. Zienkiewicz OC, Taylor RL, Nithiarasu P (2005) The finite element method for fluid dynamics, 6th edn. Elsevier, Oxford

Numerical Simulation and Visualization of Material Flow in Friction Stir Welding via Particle Tracing

N. Dialami, M. Chiumenti, M. Cervera, C. Agelet de Saracibar, J. P. Ponthot and P. Bussetta

Abstract This work deals with the numerical simulation and material flow visualization of Friction Stir Welding (FSW) processes. The fourth order Runge-Kutta (RK4) integration method is used for the computation of particle trajectories. The particle tracing method is used to study the effect of input process parameters and pin shapes on the weld quality. The results show that the proposed method is suitable for the optimization of the FSW process.

1 Introduction

Friction Stir Welding is a solid-state joining technique lately found by Thomas et al. [1]. The basic concept of FSW is the following. A shouldered pin rotating at constant rotational speed is inserted into the line between the two plates to be welded. Once the

N. Dialami (✉) · M. Chiumenti · M. Cervera · C. Agelet de Saracibar
International Center for Numerical Methods in Engineering (CIMNE), Technical University of Catalonia, UPC BarcelonaTech, Building C1, North Campus C/ Gran Capitán s/n, 08034 Barcelona, Spain
e-mail: narges@cimne.upc.edu

M. Chiumenti
e-mail: michele@cimne.upc.edu

M. Cervera
e-mail: mcervera@cimne.upc.edu

C. Agelet de Saracibar
e-mail: agelet@cimne.upc.edu

J. P. Ponthot · P. Bussetta
LTAS- MN2L, Aerospace & Mechanical Engineering, University of Liege, Building B52/3 Chemin des Chevreuils, 1, B4000 Liege, Belgium
e-mail: JP.Ponthot@ulg.ac.be

P. Bussetta
e-mail: P.Bussetta@ulg.ac.be

insertion is completed, the pin is moved along the welding line at constant rotating and advancing speeds to form the joint.

Ideally, the pin is designed to disrupt the contacting surfaces of the work-piece, shear the material in front of the tool and move the material behind the tool. The depth of deformation and the tool travel speed are mainly governed by the pin. This serves two primary functions: heating of the work-piece, and moving the material to produce the joint. In general, both the heat and the material transfer depend on the work-piece material properties, tool geometry, and FSW process parameters.

One of the main issues in the study of FSW is heat generation. During the process, the material undergoes intense plastic deformation at elevated temperatures. In the FSW process, welding is achieved by the generated heat due to friction and the material mixing/stirring process. The generated heat must be enough to allow for the material to flow and to obtain a deep heat affected zone. Insufficient heat forms the voids as the material is not softened enough to flow properly. It is of practical importance to understand the material flow characteristics for optimal tool design and obtain high structural efficiency welds. The visualization of the material flow is very useful to understand its behavior during the weld. This has led to numerous investigations on material flow behavior during FSW. A method assessing the quality of the created weld by visualization of the joint pattern is advantageous. It can be used to have a pre-knowledge of the appropriate process parameters. However, following the position of the material during the welding process is not an easy task, neither experimentally or numerically.

The experimental material visualization is difficult and needs metallographic tools. In an attempt to better understand FSW, many investigators have used experimental techniques to visualize the material flow and to estimate characteristics of FSW. Most of the studies done so far are based on the experimental study of material flow tracing. Two different tracer techniques were used by different researchers for visualization of the material flow. The first was a tracer technique by marker material where a dissimilar material is inserted into the weld line. The second technique was to weld two dissimilar materials with the FSW process and then see the material mixing. The marker materials were different Al-composites [2–4], steel balls [5], copper foil [6, 7], plasticine and brass rods [8]. The used dissimilar base materials were different magnesium alloys [9], aluminum to copper alloys [10, 11].

Alternatively, establishing a numerical method for the visualization of the material trajectory in order to gain insight to the heat affected zone has been attempted. Computational methods including the finite element method have been used to model the material flow.

In the literature, there are several works to compute the material flow. On one hand, within a Lagrangian framework, no special technique for the tracking of the material is necessary as the mesh nodes are material points. In this format, re-meshing is unavoidable [12, 13]. Meshless methods used within an updated Lagrangian formulation [14] are an interesting alternative, even if its computational cost is usually higher than the classical finite element method. In this case, re-meshing is avoided but the material flow is known only at the nodal points. On the other hand, when using an Eulerian/ALE approach, a specific technique to compute the material trajectories must be implemented. However, the mesh density used for the FSW simulation is not related to the definition of the set of particles used for the visualization of the material flow. Hence, a large number of particles can be used without increasing the computational effort devoted to the simulation of the process itself. Following this approach, the ALE formulation together with a splitting method is proposed in [15] to analyze different phases of FSW process. In [16] an Eulerian formulation together with a simple mesh moving technique is used to avoid mesh distortions and reducing the computing time due to the ALE technique.

In this work, a numerical particle tracing technology is proposed to study the extent of material stirring during the FSW process and to study the weld quality.

The outline of the chapter is as follows. Firstly the particle tracing technique is described using a RK4 integration technique for the computation of particle trajectory. Afterwards, the proposed method is applied to different examples in order to study the quality of the final joint for different process parameters and pin shapes. Finally some conclusions are drawn.

2 Particle Tracing

In this work, the FSW process is simulated using an apropos kinematic framework based on the ALE formulation [17–20] and particle tracing is performed to be able to follow the material movement in the stirring zone integrating the velocity field [21].

Due to the ALE character of the finite element analysis used, the motion of the finite element mesh is not necessarily tied to the motion of the material. During the analysis, a material particle moves through the mesh and at different time it is located inside different elements. To observe material movement around the pin, it is necessary to construct and analyze material particle trajectories. This is possible with the use of a particle tracing method (particles are treated as material points not as mesh nodal points).

Particle tracing is a method used to simulate the motion of material points, following their positions at each time-step of the analysis. This method can be naturally applied to the study of the material flow in the welding process. In the Lagrangian framework, as the mesh nodes represent the material points, the trajectories are the solution of the governing system of equations. When using an Eulerian or ALE framework the solution does not gives directly information about the material points. However, the obtained velocity field can be used to get an insight of the extent of material mixing during the weld.

In this method, firstly, a set of points representing the material points (tracers) are distributed in the domain and then, a Lagrangian Ordinary Differential Equation (ODE) for the computation of material displacement at a post-process level must be solved. Each particle's path is followed in time integrating the following ODE equation:

$$\frac{D\left(\mathbf{X}\left(t\right)\right)}{Dt} = \mathbf{V}\left(\mathbf{X}(t), t\right) \tag{1}$$

Integrating (1) yields:

$$\mathbf{X}(t) = \mathbf{X}_0 + \int_0^t \mathbf{V}(\mathbf{X}(t), t)\ \mathrm{d}t \tag{2}$$

where $\mathbf{X}(t = 0) = \mathbf{X}_0$ is the initial position of the particle, $\mathbf{X}(t)$ is the position of the material points at time t and $\mathbf{V}(\mathbf{X}(t), t)$ is the velocity of the tracer in the position $\mathbf{X}(t)$ and time t.

An appropriate time integration method for the solution of the ODE equation is needed in order to track the particles. To solve the ODE equation, there exist a large amount of integration techniques ranging from the simple first order Backward Euler (BE) scheme to higher order Runge-Kutta schemes. Among them three well-known methods are chosen and compared in [21]: Backward Euler with Sub-stepping (BES), Fourth order Runge-Kutta (RK4) and Back and Forth Error Compensation and Correction methods (BFECC). These methodologies, widely used in fluid dynamics, are suitable and robust tools to study the FSW problem allowing for a clear visualization of the material movement at the stir-zone leading to a better understanding of the welding process itself. Among them, RK4 is found to be more precise for the simulation of FSW problem.

Moreover, a search algorithm must be executed to find the position of the material points in the Eulerian or ALE meshes in order to identify the element containing the tracer. The tracer velocity is obtained interpolating the nodal velocity of the background mesh and the corresponding interpolation (shape) functions, $N^j(\mathbf{X}(t))$, as:

$$\mathbf{V}(\mathbf{X}(t), t) = \sum_{j=1}^{n} \mathbf{v}_j(t) \cdot N^j(\mathbf{X}(t)) \tag{3}$$

where the velocity field, $\mathbf{v}_j(t)$, is known at each node, j, of the finite element mesh representing the domain at any time, t, of the analysis.

2.1 RK4 Method

According to the fourth order accurate RK4 method, the particle position at time-step n+1 is computed from the advection of the initial position by four weighted incremental displacement at intermediate time-steps.

$$X_{n+1} = X_n + \frac{1}{6}\left(\Delta\mathbf{X}^{(1)} + 2\Delta\mathbf{X}^{(2)} + 2\Delta\mathbf{X}^{(3)} + \Delta\mathbf{X}^{(4)}\right) \tag{4}$$

where the incremental displacements are computed as

$$\begin{cases} \Delta\mathbf{X}^{(1)} = \mathbf{V}\left(\mathbf{X}_n, t_n\right)\Delta t \\ \Delta\mathbf{X}^{(2)} = \mathbf{V}\left(\mathbf{X}_n + \frac{1}{2}\Delta\mathbf{X}^{(1)},\ t_n + \frac{\Delta t}{2}\right)\Delta t \\ \Delta\mathbf{X}^{(3)} = \mathbf{V}\left(\mathbf{X}_n + \frac{1}{2}\Delta\mathbf{X}^{(2)},\ t_n + \frac{\Delta t}{2}\right)\Delta t \\ \Delta\mathbf{X}^{(4)} = \mathbf{V}\left(\mathbf{X}_n + \Delta\mathbf{X}^{(3)},\ t_n + \Delta t\right)\Delta t \end{cases} \tag{5}$$

The RK4 method is a fourth order method, meaning that the error per step is $O\left(\Delta t^5\right)$, while the total accumulated error is $O\left(\Delta t^4\right)$. The RK4 method is an appropriate choice as it can integrate exactly a circular trajectory, a standard particle path in FSW.

The nodal velocity field, $\mathbf{v}_{n+\frac{1}{2}}$, corresponding to the mid-time step $t_n + \frac{\Delta t}{2}$ is obtained as:

$$\mathbf{v}_{n+\frac{1}{2}} = \frac{\mathbf{v}_n + \mathbf{v}_{n+1}}{2} \tag{6}$$

where $\mathbf{v}_n$ and $\mathbf{v}_{n+1}$ are the nodal velocity fields at times t_n and t_{n+1}, respectively.

3 Examples

The material flow during FSW is complex and the understanding of deformation process is limited. It is important to point out that there are many factors that can influence the material flow during FSW. These factors include tool geometry, welding parameters, material types, work-piece temperature, etc.

The proposed method is used to investigate the effect of these factors on the qualification of the final weld. The first example studies the effect of the input process parameters on the joint creation and the second one considers the effect of the pin shapes on the weld quality.

3.1 Input Process Parameters Effect on the Weld

To study the effect of the input process parameters on the joint quality, a 2D example is considered. The model is a transversal cut of the pin, with 10 mm diameter, perpendicular to the rotation axis. The cut represents the mid-section of the real threaded pin. The contact condition between the pin and the work-piece (AA2195-T8) is considered to be perfect sticking. Process parameters are the same as in the experiment: welding speed $\mathrm{V_s} = 5.0833$ mm/s and rotational speed $\mathrm{V_r} = 500$ rpm. A Sheppard-Wright constitutive model is used [21].

A set of 100×150 particles in a shape of a $20 \times 30\,\text{mm}^2$ rectangle at their initial position is located right in front of the pin. The whole model is discretized with a mesh of 4986 triangular elements. The problem is solved using the *v*/*p* mixed formulation stabilized by the OSS stabilization method [17–19]. The RK4 time integration method is applied for the solution of the particle tracing problem. The model has been already validated in the authors' work [21].

To study the effect of the input parameters and to gain an insight of the ratio influence between advancing and rotational velocities, two "limit cases" and the original case problem with the same boundary condition and material properties and the velocity fields: (a) Vs $=$ 5.0833 mm/s; Vr $=$ 0 rpm, (b) Vs $=$ 0.50833 mm/s; Vr $=$ 500 rpm and (c) Vs $=$ 5.0833 mm/s; Vr $=$ 500 rpm are analyzed. Apart from the original problem, one case where the rotational velocity is zero and the other case where the advancing velocity is very small (advancing velocity cannot be zero) are considered.

Studying these three cases reveals some characteristics of the material flow pattern. Figure 1 shows the particles pattern after joint creation and the velocity streamlines for these three sets of velocity. Case (c) is the original problem [21]. It shows an initially straight transverse set of particles that has been welded through. Note that the material moves backward in a curve, and a thin zone is swept forward on the advancing side, as seen in the predicted particle tracks.

It can be observed that when the pin is not rotating (case (a)), the flow passes through the pin (an obstacle). In this case, the advancing velocity is the same as in the original problem. The discontinuous line is located in the center of the plate on the weld line and the joint is not created.

In case (b), the rotational velocity of the pin is the same as the original problem but the advancing velocity is 10 times smaller. Even though case (b) and case (c) show the same velocity contour field, streamlines in case (b) represent more material rotation around the pin. The reason is the much bigger ratio between the rotational and advancing velocities than in the case (c). Moreover, via particle tracing technique the effect of the input parameters on the weld quality can be observed. It is shown that when the advancing velocity is much smaller than the rotational one, the joint is defective and non-qualified and the weld line is not located at the center line.

By comparing the three cases, it can be concluded that the ratio between the rotational and advancing velocities is crucial to obtain a qualified joint. They cannot be arbitrary selected. Particularly, a very low advancing velocity comparing with the rotational one does not lead to a qualified joint.

3.2 Different Pin Shapes Effect on the Weld

The second example investigates the effect of different pin shapes on the joint creation using the proposed particle tracing method. Different types of pin shape are considered and shown in Fig. 2 including (a) triflute; (b) trivex; (c) circular; (d) triangular. The pins are generated from an originalyl circular section of 10 mm diameter

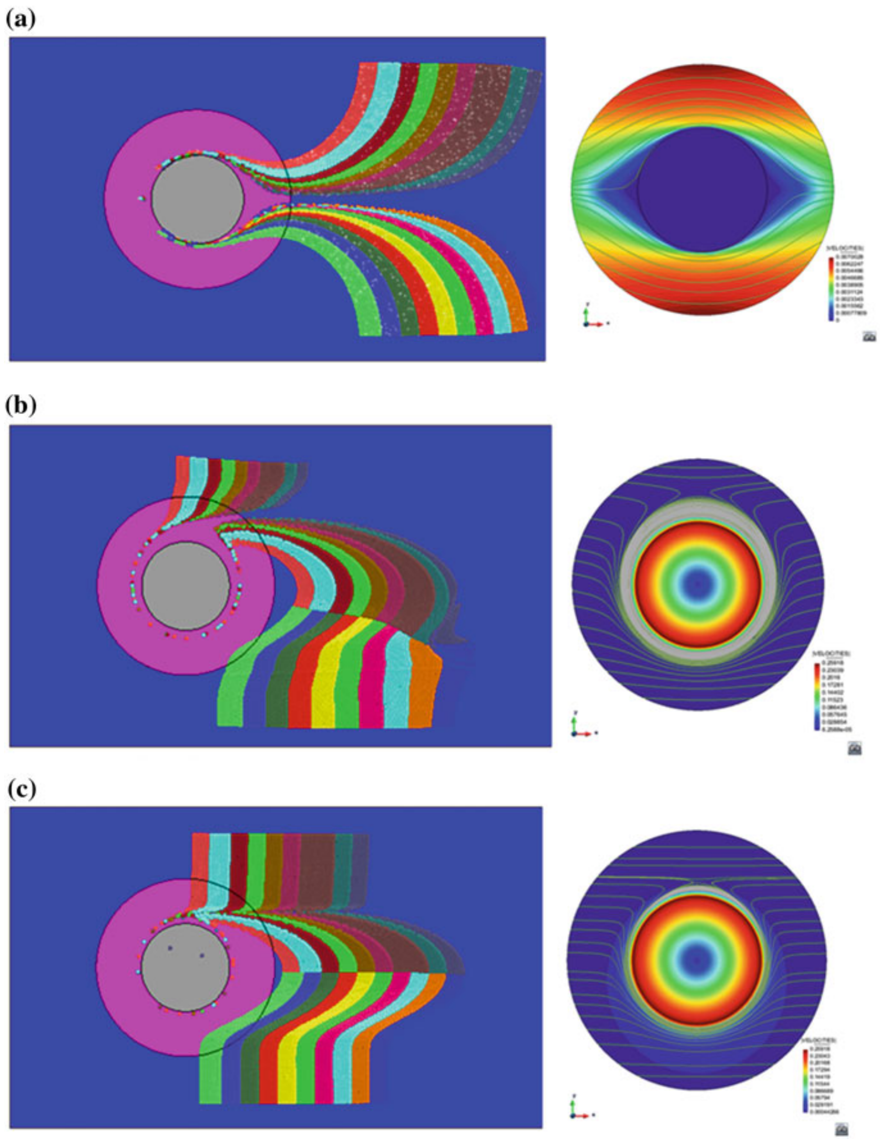


Fig. 1 Creation of the weld joint with different input parameters **a** Vs = 5.0833 mm/s; Vr = 0 rpm **b** Vs = 0.50833 mm/s; Vr = 500 rpm **c** Vs = 5.0833 mm/s; Vr = 500 rpm

(Fig. 2). The trivex pin design is approximately triangular; the three points of the pin form an equilateral triangle and are connected by convex sides. The triflute pin shape is obtained from an original circular section removing three circular segments.

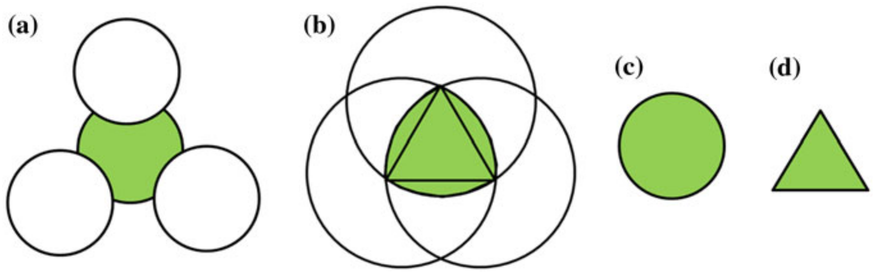


Fig. 2 Different types of pin shape. **a** Triflute, **b** trivex, **c** circular, **d** triangular

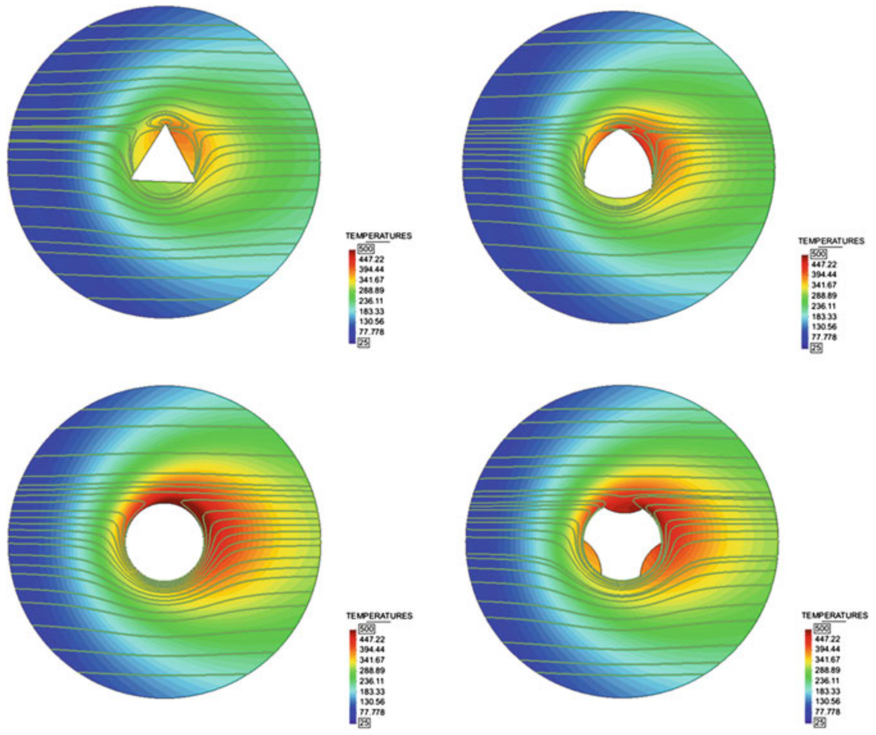


Fig. 3 Temperature field and streamlines obtained from different pin

A square domain of $80 \times 80\ \text{mm}^2$ is considered with a circular heat affected zone of 20 mm in diameter. An ALE framework is considered in the heat affected zone and the rest of the circular domain is defined in the Eulerian framework. The RK4 integration technique is used for the solution of the particle tracing problem.

In a first step, the problem is analyzed under the stick condition. Figure 3 illustrates the temperature contour fields obtained together with the streamlines. The streamlines show that: voids are created using the pins with sharp corners such as trivex and

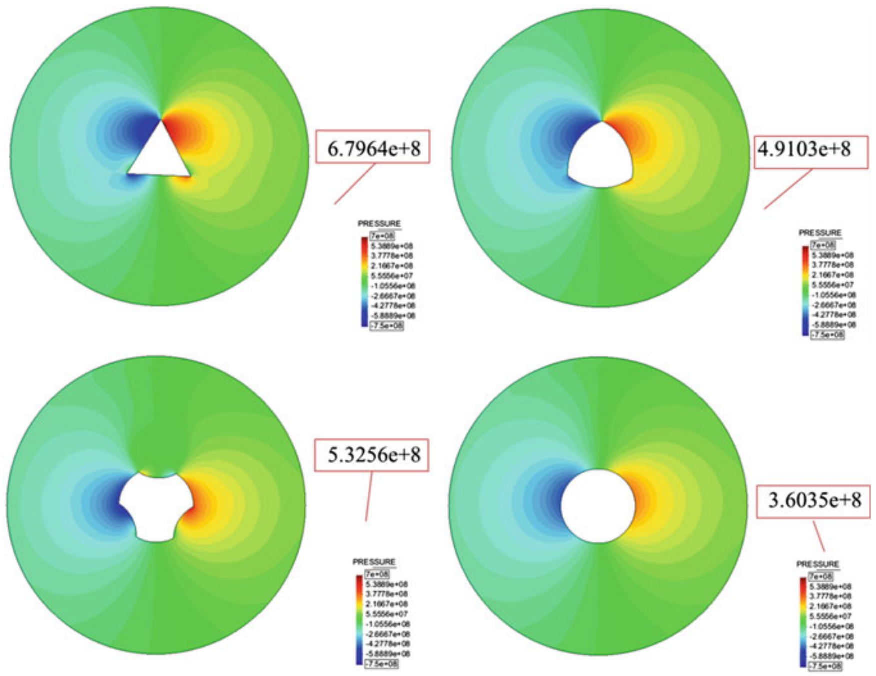


Fig. 4 Pressure contour field obtained from different pin shapes

especially triangular pins; the circular and triflute pin present similar streamlines taking into account that the stick condition is assumed. Note that the triflute shows significantly more material being captured and taken around the tool more than once, whereas the trivex struggles to fill the space behind the tool on the advancing side. This is therefore consistent with the generation of a void in the wake of a trivex tool. The triflute pin has a high swept rate due to the segments and a tool design with a higher swept rate reduces the voids.

It can be observed that the generated heat is greater for the triflute and the circular pin than for the trivex and the triangular pins as in the stick condition more material move together with the pins.

The pressure contour field is illustrated in Fig. 4. Pins with sharper corners have higher maximum pressure value as for triangular and trivex pins than the ones with convex sides as for circular and triflute pins. However the maximum pressure is of the same order.

In a next step, the effect of slip condition on the same problem is studied. In this case, the pin rotational velocity has less effect on the work-piece than in the stick case. The triflute and the circular pin lead to considerably different streamlines (Fig. 5). The streamlines for a circular pin show a passing flow through an obstacle while for a triflute pin, they show the trapped material in the segments of the pin moving with it. In the slip case, the joint is not qualified even though in the triflute case, the joint is created due to the effect of the segments. In the stick case, the joint

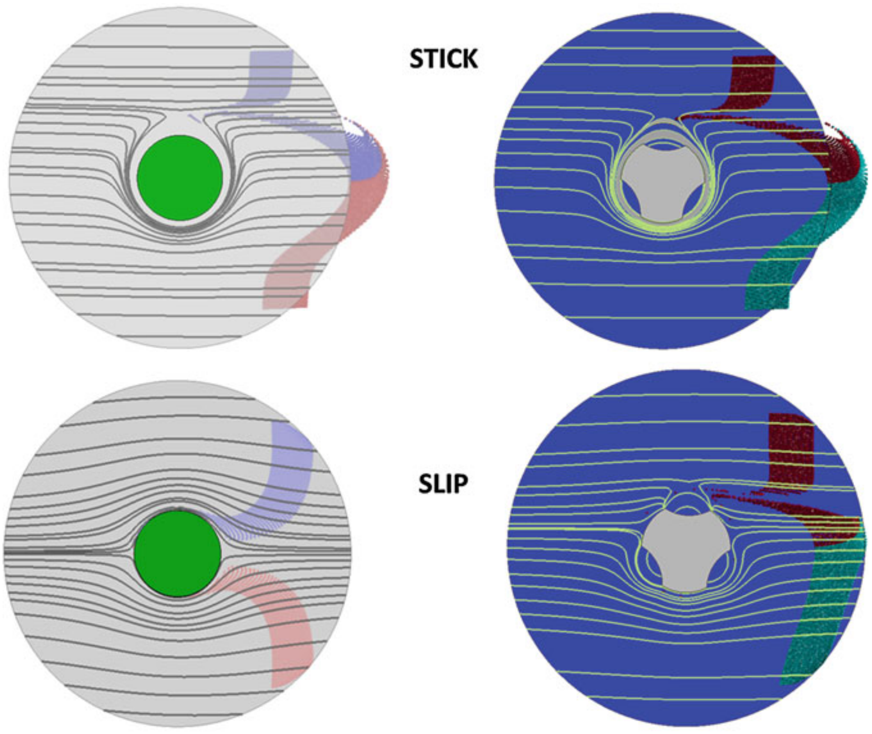


Fig. 5 Streamlines and created joints in both the stick and slip cases for circular and triflute pins

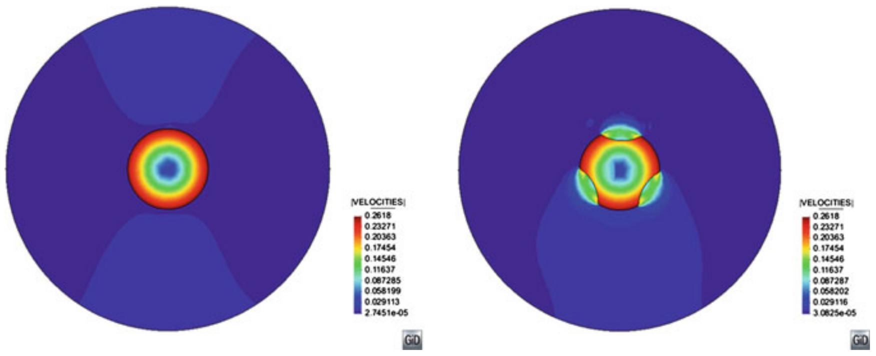


Fig. 6 Velocity contour field in the slip case for circular and triflute pins

is created following the ring patterns observed generally in the FSW process. The effect of the segments can be also seen in the Fig. 6 for the slip case as the material close to the pin is affected by the pin velocity.

Figure 7 shows that for both the slip and stick cases, joints are created using triangular and trivex pins. However, in the stick case, the joint is not qualified and

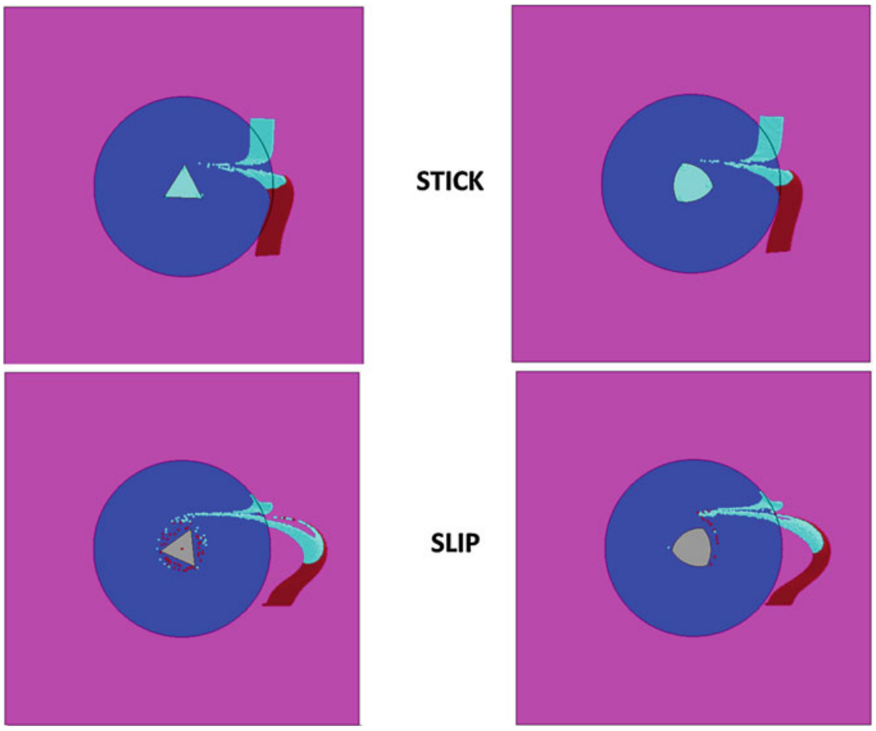


Fig. 7 Streamlines and created joints in both the stick and slip cases for triangular and trivex pins

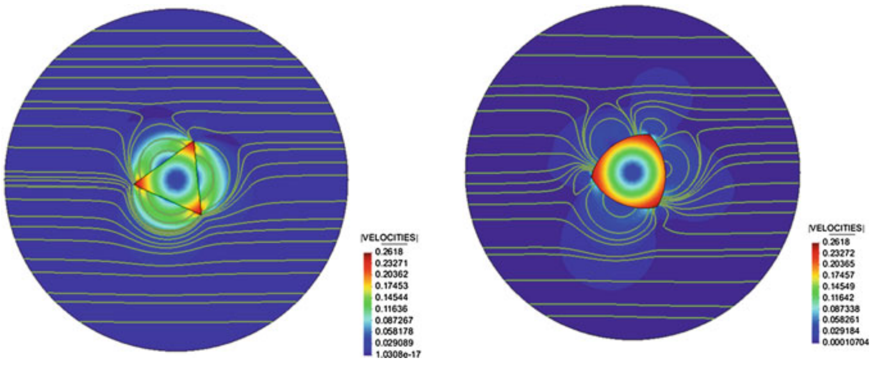


Fig. 8 Streamlines and velocity contour field in the slip case for triangular and trivex pins

does not follow the usual pattern of FSW due to the void creation. When the slip condition is assumed, the voids are not created. Material around the pin does not share the same velocity as the pin, but it moves due to the non-circular shape (Fig. 8) and a qualified joint is created.

It can be concluded that different types of pin shapes can be selected for different conditions of the weld. In stick case, pins without sharp corners create qualified joints while the pins with sharp corners can be used in slip cases.

Material deformed by the friction stir tool must be capable of filling the void produced by a traversing pin. If the tool design is incorrect, the deformed material will cool before the material can fully fill the region directly behind the tool.

The presented results are preliminary, but the proposed method could clearly be of great benefit in reducing experimental trials if near optimal welding conditions could be predicted directly from knowledge of the material joint behavior.

4 Conclusion

The work deals with the simulation and visualization of material flow. The simulation of the transient phase is important for understanding the material behavior. The model can provide this insight by computing the particles' thermo-mechanical history. If the process is defined in an ALE/Eulerian setting, an additional method must be introduced in order to find the particles' history. The particle tracing method for the material stirring during and after welding is applied to the material flow visualization of the FSW process. The RK4 integration method is used for the computation of particle trajectories.

From the ring shape flow pattern left after the welding, it is found that the ratio between the rotational and the advancing speed is one of the key points for the qualified joint creation. The effect of pin shapes on the weld quality is studied. It is found that in the stick case, pins with sharp corners (triangular and trivex) generate voids while this problem does not appears in the slip case. Moreover, the effect of the segments of a triflute pin on the weld quality is studied and shows that the material trapped in the segments moves with the pin in both stick and slip cases.

Acknowledgments This work was supported by the European Research Council under the Advanced Grant: ERC-2009-AdG "Real Time Computational Mechanics Techniques for Multi-Fluid Problems". The authors are also thankful for the financial support of the Spanish Ministerio de Educación y Ciencia (PROFIT programme) within the project CIT-020400–2007-82.

References

1. Thomas WM, Nicholas ED, Needham JC, Murch MG, Temple-Smith P, Dawes CJ (1991) Friction-stir butt welding. GB Patent No. 9125978.8, International Patent No. PCT/GB92 /02203
2. London B, Mahoney M, Bingel B, Calabrese R, Waldron D (2001) Experimental methods for determining material flow in friction stir welds. The third international symposium on friction stir welding, Kobe, Japan, 27–28 Sept 2001
3. Reynolds AP (2008) Flow visualization and simulation in fsw. Scripta Materialia 58:338–342

4. Seidel TU, Reynolds AP (2001) Visualization of the material flow in aa2195 friction stir welds using a marker insert technique. Metall Mater Trans A 32:2879–2884
5. Colligan K (1999) Material flow behaviour during friction stir welding of aluminium. Weld J 78:229–237
6. Guerra M, Schmids C, McClure JC, Murr LE, Nunes AC (2003) Flow patterns during friction stir welding. Mater Charact 49:95–101
7. Dickerson T, Shercliff HR, Schmidt H (2003) A weld marker technique for flow visualization in friction stir welding. 4th international symposium on friction stir welding, Park City, Utah, USA, 14–16 May 2003
8. Kallgren T, Jin L-Z, Sandstrom R (2008) Material flow during friction stir welding of copper. 7th international friction stir welding symposium, Awaji Island, Japan, 20–22 May
9. Johnson R, Threadgill P (2003) Friction stir welding of magnesium alloys. In: Kaplan HI (ed) Magnesium technology 2003 (TMS-The Minerals, Metals & Materials Society, 2003), pp 147–152
10. Ouyang J, Yarrapareddy E, Kovacevic R (2006) Microstructural evolution in the friction stir welded 6061 aluminum alloy (T6-temper condition) to copper. J Mater Proc Technol 172:110–122
11. Abdollah-Zadeh A, Saeid T, Sazgari B (2008) Microstructural and mechanical properties of friction stir welded aluminum/copper lap joints. J Alloy Compd 460:535–538
12. Buffa G, Fratini L, Micari F, Shivpuri R (2008) Material flow in fsw of t-joints: experimental and numerical analysis. Int J Mater Form 1(1):1283–1286
13. Buffa G, Ducato A, Fratini L (2011) Numerical procedure for residual stresses prediction in friction stir welding. Finite Elem Anal Des 47(4):470–476
14. Alfaro I, Racineux G, Poitou A, Cueto E, Chinesta F (2009) Numerical simulation of friction stir welding by natural elements method. Int J Mater Form 2(4):225–234
15. Guerdoux S, Fourment L (2009) A 3d numerical simulation of different phases of friction stir welding. Model Simul Mater Sci Eng 17:075001
16. Feulvarch E, Roux J-C, Bergheau J-M (2013) A simple and robust moving mesh technique for the finite element simulation of friction stir welding. J Comput Appl Math 246:269–277
17. Chiumenti M, Cervera M, Dialami N (2013) Numerical modeling of friction stir welding processes. Comput Methods Appl Mech Eng 254:353–369
18. Dialami N, Chiumenti M, Cervera M (2013) An apropos kinematic framework for the numerical modelling of friction stir welding. Comput Struct 117:48–57
19. Agelet de Saracibar C, Chiumenti M, Cervera M, Dialami N, Seret A (2014) Computational modeling and sub-grid scale stabilization of incompressibility and convection in the numerical simulation of friction stir welding processes. Archives of Computational Methods in Engineering 21(1):3–37. doi:10.1007/s11831-014-9094-z
20. Bussetta P, Dialami N, Boman R, Chiumenti M, Agelet de Saracibar C, Cervera M, Ponthot JP (2013) Comparison of a fluid and a solid approach for the numerical simulation of friction stir welding with a non-cylindrical pin. Steel Research International. doi:10.1002/srin.201300182
21. Dialami N, Chiumenti M, Cervera M, Agelet de Saracibar C, Ponthot JP (2013) Material flow visualization in friction stir welding via particle tracing. Int J Mater Form. doi:10.1007/s12289-013-1157-4

Some Considerations on Surface Condition of Solid in Computational Fluid-Structure Interaction

Masao Yokoyama, Kohei Murotani, Genki Yagawa and Osamu Mochizuki

Abstract The surface condition of solid or structure is a serious issue in the numerical simulation of the Fluid Structure Interaction. The present paper describes an engineering model for calculating the FSI with the surface condition for the hydro-gel solid, which is employed to modify the wall shear stress at the interface between the fluid and the solid. Using the proposed model, we show some numerical results including the large scale parallel computing of water splash generated by a hydro-gel sphere diving into water, which is compared well with the experimental observation.

Keywords Fluid structure interaction · Surface condition · Splash · Hydro-gel · Particle method · Numerical simulation

1 Introduction

The Fluid-Structure Interaction (FSI) is one of the most popular topics in the computational mechanics. It covers a wide range of phenomena of social, scientific and engineering fields such as vehicle, medicine, civil engineering and construction, agriculture, forestry, disaster prevention, music, sports, etc. (see Fig. 1).

Related research topics of fluid dynamics and the FSI are among others vortex, vibration of structure, sloshing, droplet, splash and bubble. For example, the vibration

M. Yokoyama
School of Information Science, Meisei University, Hino, Japan

K. Murotani
School of Engineering, University of Tokyo, Bunkyo, Japan

G. Yagawa (✉)
Center for Computational Mechanics Research, Toyo University, Bunkyo, Japan
e-mail: yagawag@gmail.com

O. Mochizuki
Faculty of Science and Engineering, Toyo University, Bunkyo, Japan

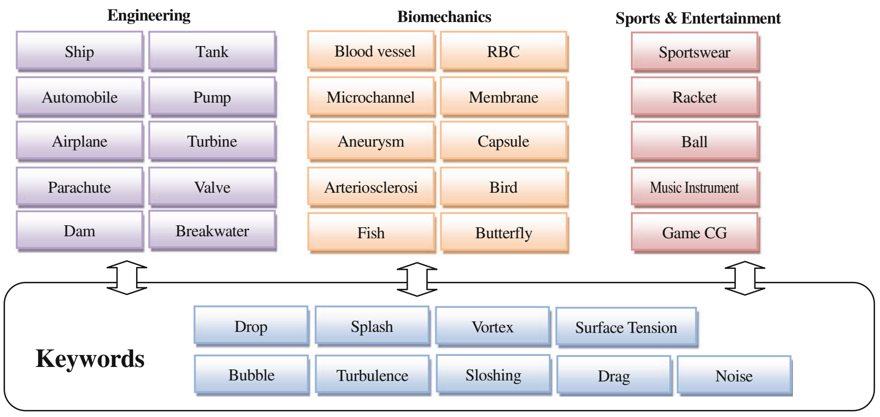


Fig. 1 Some keywords and applications of fluid structure interaction

of structure caused by the Kármán's vortex street has been studied for many years, which is a locking phenomenon caused by the vortex street behind a spherical cylinder [1]. The vortex and the exfoliation occur when a solid or a structure moves in fluid, which often result in the destruction, the noise, the stall of an airplane or the drag of a vehicle, and the various studies have been performed: the effect by the surface unevenness such as a turbulator, a vortex generator, a tripping-wire, a riblet of wing, dimples of golf ball [2], the drag reduction of ship by the micro bubble [3], the effect of deformation by an elastic body [4], the relation of vortex and vibration [5], etc. The FSI study has contributed to the sport engineering: the improvement of the movement form of swimmer [6, 7]. Regarding the sound of musical instruments, the vibration of a musical instrument and the circumference air and the pronunciation mechanism of an air lead of pipe organ or flute have been studied [8–10].

When we discuss the interaction between solid and fluid, the condition of the interface between them is controversial. It is well known that the surface condition, the roughness of surface or the uneven shape of the solid surface gives some influence on the flow fields and the movement of the solid as seen in the case of the dimple of a golf ball [11].

The free surface flow such as splash and drop induced by movement of a solid object is also interesting topic of the FSI. For example, even if the surface of wall of hydro-gel ball and that of acrylic resin ball look smooth each other, the splash's form created by the hydro-gel ball differs from that of the acrylic resin ball and the velocity distribution of the water around each ball is different as well.

In the biomechanics fields, there are some interesting papers besides well-known study such as flapping wings or swimming fish; Yabe et al. [12] developed an algorithm for the calculating surface tension and contact angle of the motion of water striders, finding two types of movement by the experiment and the simulation. On the other hand, the finite volume method simulation of the promotion function underwater by a flagellum as a micro propulsion was performed, where the effect of the flagellum with projecting mastigonemes as the surface condition was verified by Kobayashi et al. [13]. In these researches, the interfacial tension and shear friction due to the difference of physical shape of surface as for the interaction of living thing's movement and water flow were studied.

Although several studies have been published on the treatments of the surface of wall in numerical computational field, majority of them are those by the introduction of the contact angle expressing the wettability of the surface of wall. Vibration and separation of droplet by MPS method [14], deformation of the gas which goes up the inside of a liquid [15], contact angle of droplet on the wall with hybrid technique of particle and mesh [16] are among others. But, these are not so accurate because they calculate the curvature of interface with the ratio of discrete particle count using the Continuum Surface Force method.

Experimental observation of the splash of a ball which plunged into the water surface was studied by Worthington [17], where the influence of the splash by the difference in the state of the surface of a ball was discussed (Fig. 2). With the dry smooth ball, the size of the splash became small. With the ball made coarse with the sandpaper or the wet ball, it became a big crown-like splash. It shows that, in the dive game of swimming, since splash will become large if a swimmer jumps into water pool without wiping the body well or with swimming suit wet, the score becomes disadvantageous. Thus, it is important to jump into water pool after a swimmer wipes body well in order to get the high score.

Splash variation of milk-crown was observed by Krechetnikov and Homsy [18] (Fig. 3), where they found the crown type differs according to the Weber number and also discovered frustration phenomena in the wave number selection of the crown spike structure. Akers et al. [19] studied the influence of non-Newtonian fluid on splash formation, focusing on the property of water. Duez et al. [20] reported the relation between the splash formation and the sound generation, studying the effect of the hydrophobic or hydrophilic surface of the body surface on splash. Yoon et al. [21] studied the finger generated at the tip of film flow, which went along the surface of object when it plunging into water surface.

In the numerical simulation, a ship attacked by huge wave and solid cube falling into a recipient with water were calculated by Idelsohn et al. [22] (Fig. 4). They claimed in the research as follows; a method is presented for the solution of the incompressible fluid flow equations using a Lagrangian formulation. The interpolation functions are those used in the Meshless Finite Element Method (MFEM). Classical stabilization terms used in the momentum equations are unnecessary due to the lack of convective terms in the Lagrangian formulation. Furthermore, the Lagrangian formulation simplifies the connections with fixed or moving solid structures, thus providing a very easy way to solve fluid-structure interaction problems.

In the most of the FSI studies mentioned above, however, the wall of the solid or the structures is assumed to be as non-slip condition in numerical simulation. Thus, there are few studies which take the condition of slip at the surface of the solid object into consideration, although the analyses of the flow in consideration of structural unevenness or elasticity of the surface of a wall are conducted. The above non-slip condition is not a realistic assumption in some cases. For example, creatures

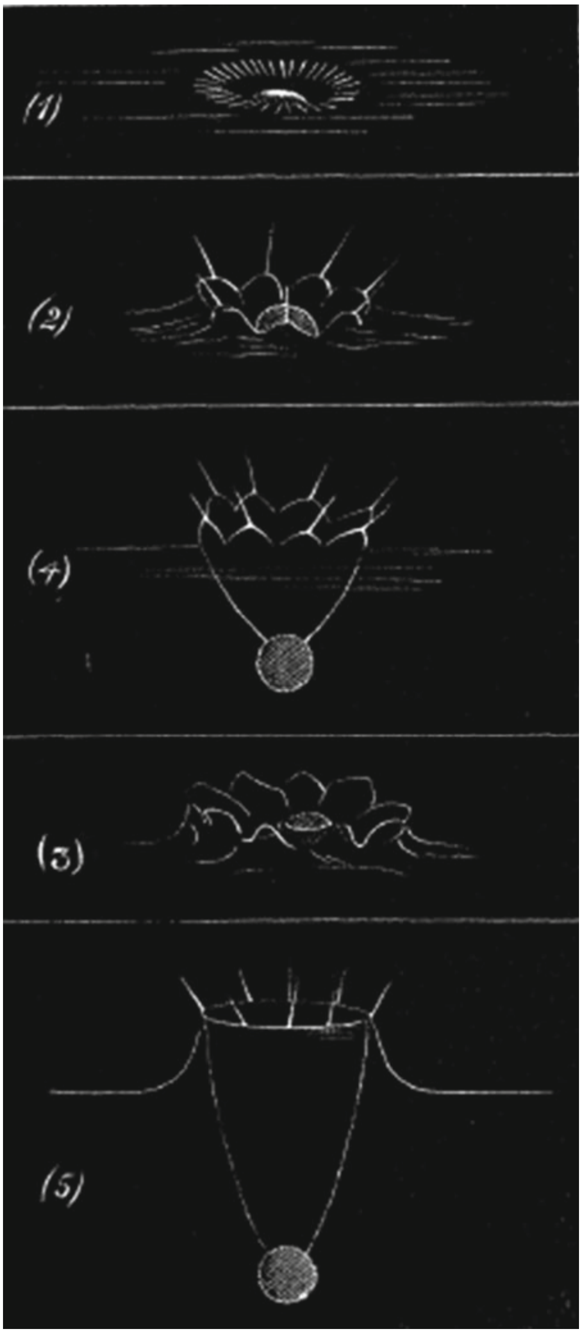


Fig. 2 Observation of the splash by a ball [17]

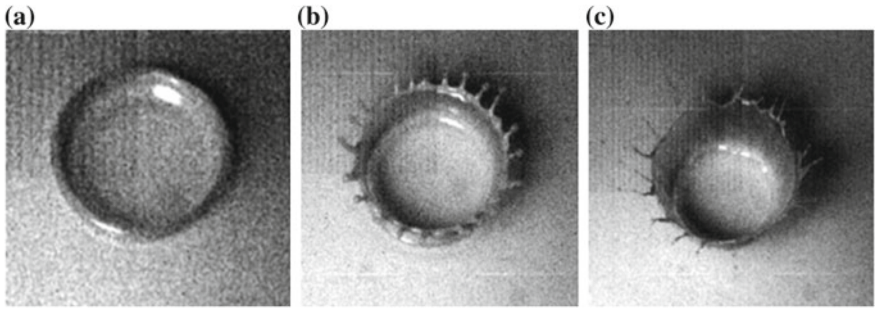


Fig. 3 Different crown type occurs according to the Weber number (We = $\rho D V^2 / \gamma$, ρ density, D diameter V velocity, γ: surface tension) [18]. **a** Regular axisymmetric crown. **b** Regular crown with spikes. **c** Irregular crown

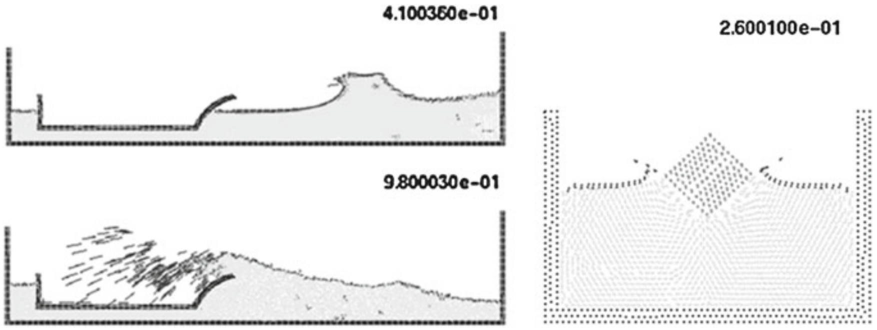
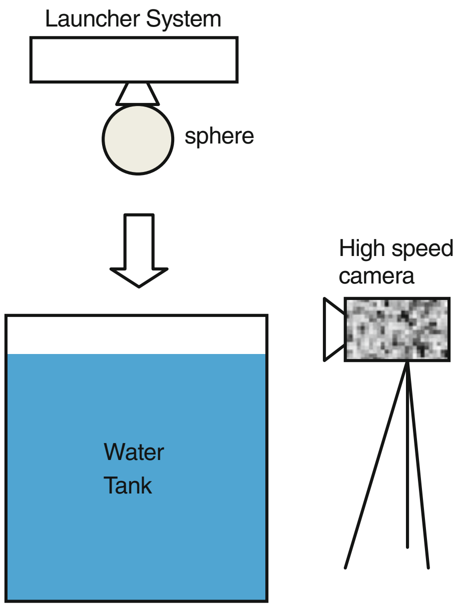


Fig. 4 Fluid-structure interaction by meshless FEM [22]

living in water such as fish and amphibians have a slimy mucus skin, whose principal ingredient is a hydrogel known as mucin [23]. Furthermore, since the inner wall of the digestive organs or the blood vessel has a slippery surface, it seems important to take the characteristics of such slippery surface into consideration in numerical simulation.

In this paper, we focus on the treatment of the surface condition of an object in the FSI problem. The experimental observation of splash is given as a suitable example in the Sect. 2. We explain an outline of the numerical method of the Navier Stokes equation by the particle method in the Sect. 3. The model is proposed introducing the slip of object's surface to the particle method, and the numerical simulation of the splash under different surface conditions is carried out, comparing with the experimental results in the Sect. 4. We show the results of splash by the large scare parallel calculation in the Sect. 5. The concluding remarks and future works are given in the Sect. 6.

Fig. 5 Experimental setup



2 Experimental Study of Splash

In this section, the experimental result of splash generated by a hydro-gel sphere and an acrylic resin sphere is shown to study the difference of splash patterns caused by the surface condition of the solid object.

The dynamic views of the splash are recorded by using a high speed CMOS camera (Vision Research Inc., Phantom v7.1), where the camera is set as 4,000 frames per second (Fig. 5). The test condition is as follows: the radius of a sphere R is 10 mm, the initial height h is 50R, and the impact velocity of the sphere at the water surface V_i is 2.4 m/s.

The experimental results of a splash formed by a sphere impinging on water surface are shown in Fig. 6, comparing the primary splash formed by a hydrogel sphere (Fig. 6a) with that by an acrylic sphere (Fig. 6b). The primary splash means the splash, which rises first after an object plunging. The primary splash formed in the case of the hydrogel is a kind of the crown-type. On the other hand, the acrylic sphere creates the column type primary splash. The splashes are considered to be formed by the dynamics of the film-flow [24], which is a thin water flow around a sphere surface and generated immediately after the sphere impacts the water surface. The difference between the formation processes of Fig. 6a and b is due to the difference of the film separation from the sphere surface.

When the film is separated from the sphere surface, a crown-type splash is formed. The above film separation is presumably caused by the increase in the film velocity according to the hydrophilic property of the solid wall and the attractive or repulsive force such as the electrostatic force between the solid wall and the water. This

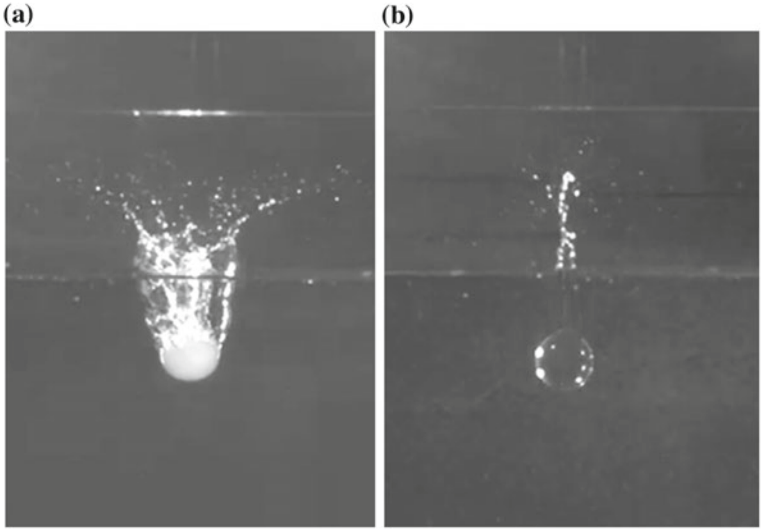


Fig. 6 Comparison of splash patterns between **a** hydrogel (Aqar) and **b** acrylic resin (radius of sphere = 10 mm and impact velocity = 2.21 m/s)

experimental observation suggests that the numerical simulation should take into consideration the various surface conditions as the interaction between the object and the water.

3 Governing Equations and Particle Method

Although several studies have paid attention to the coarseness of the surface or the liquid exfoliation in the FSI simulation, the difference of a splash by the material cannot be simulated with the conventional method. In other words, the simulated pattern of the splash by the hydrogel object and that by the acrylic resin object become the same result. The reason will be attributed to the fact that the above simulations have assumed the boundary between the fluid and the solid to be of the non-slip type. In this paper, we propose a calculation method, where the difference of a splash due to the difference of the solid material is realized.

Employed method in the present paper is the MPS method, which is a particle method recognized as an effective technique in performing the simulation of the FSI. The method is a semi-implicit method, where, after calculating the temporary position of particles with the equation of motion in the explicit stage, the Poisson equation of pressure is solved in order to satisfy the condition that the number of particles per a small volume is constant as the mass conservation. We discuss here how to introduce the effect of a slip into the MPS in order to solve the flow field around the hydro-gel surface.

The governing equation of the present flow is the incompressible Navier-Stokes equation as follows,

$$\frac{D\vec{u}}{Dt} = F - \frac{1}{\rho}\nabla P + \nu\nabla^2\vec{u} \tag{1}$$

where u is the velocity vector of fluid, ρ is the density of fluid P is the pressure, ν is the kinematic viscosity of fluid and F is the external force. Assuming two particles i and j, where there exist, respectively, pressure p_{i} and p_{j}. The gradient of pressure at the point i is written as [25]

$$\nabla P = \frac{d}{n^0}\sum_{j\neq i}\left[\frac{p_j - p_i}{|\vec{r}_j - \vec{r}_i|^2}(\vec{r}_j - \vec{r}_i)\kappa(|\vec{r}_j - \vec{r}_i|)\right] \tag{2}$$

where d is the constant value, which is equal to the dimension of space to be analyzed and n^0 is called the particle number density.

The Laplacian of velocity at the point i is written as

$$\nabla^2\vec{u} = \frac{2d}{\lambda n^0}\sum_{i\neq j}\left[(\vec{u}_j - \vec{u}_i)\kappa(|\vec{r}_j - \vec{r}_i|)\right] \tag{3}$$

where λ is a parameter, which is introduced in order to make statistical distribution coincided with an analytic solution, and κ is the weight function assumed as follows,

$$k(r) = \begin{cases} \frac{r_e}{r} - 1 & (0 \leq r \leq r_e) \\ 0 & (r_e \leq r) \end{cases} \tag{4}$$

Here, r is the distance between two particles and r_e is the cut-off radius.

The algorism of MPS method takes following procedure (i) to (iii); (i) Tentative velocity u^* is calculated at the explicit stage using F and viscous term of Navier-Stokes equation (1), (ii) the Poison's equation of the pressure is solved at the implicit calculation stage and the pressure p is obtained. Then, revised velocity u' obtained by this p in order that the particle number density in area is conserved, (iii) and u' is added to u^* then target velocity u on t is obtained, (iv) time step is increased, $t = t + dt$, where the time step dt is 0.001 s in the present paper.

4 Flow Around Hydro-Gel Wall with Slip

The hydro-gel, which is a kind of polymer gel is considered here, where the slip ratio is defined by the moisture content in the hydro-gel. We discuss here how we introduce heuristically the influence of the slip at the hydro-gel wall into the calculation. Let us use the diving sphere made of agar as the hydrophilic material, which is a kind of hydrogel like gelatin, and known to be easy to control its water content and to create arbitrary shape. Agar consists of crosslinked structure by polymer called agarose and lots of water molecule between the polymer structures, which is known to create a slippery surface. For example, Eddington et al. [26] reported the use of the hydrogel as the valve for flow control of a microchannel, and Beebe et al. [27] discussed the effectiveness of hydrogel structure for flow control on micro fluidic channels.

Figure 7 shows the velocity distributions of water flow near the surface of the acrylic resin versus that of the agar-gel, where δ is the height of water flow and u is the velocity of water. Here, τ being the wall shear stress under the no-slip condition and τ' that under the slip condition, the slip ratio α is defined as follows,

$$\alpha = \tau'/\tau \quad (0 < \alpha < 1) \tag{9}$$

Here, the shear stress is obtained by flow velocity near the wall experimentally.

$$\tau = \mu \frac{du}{dy}|_{y=0} \tag{10}$$

Figure 8 shows the experimental relations between the swelling ratio S and the slip ratio α for the agar-gel and the carrageenan-gel, respectively, where the swelling ratio S is defined as follows [28],

$$S = (m_{\text{water}} + m_{\text{gel}})/m_{\text{gel}} \tag{11}$$

where m_{water} is the mass of water and m_{gel} that of the solid-gel. S increases with the amount of water contained in the solid-gel. The agar employed in this study is a kind of hydrogel [29], which is easy in handling and controlling its shape and the degree of the swelling [30]. Figure 8 suggests that α decreases with the increase of S, or can be expressed as

$$\alpha = 1 - \beta S \tag{12}$$

where β is estimated to be 1.2×10^{-3} in the case of the agar. It is summarized that larger S gives more slip on the surface.

In this paper, the above relation between the increase of the swelling ratio and the reduction of the wall friction from our experiment is taken into consideration near the wall in the viscous term of the Navier-Stokes equation in a heuristic manner. Since the shear force acting between the wall and the fluid is directly related with the viscosity term of Eq. (3), we modify the term as

$$\nabla^2 \vec{u} = \frac{2d}{\lambda n^0} \sum_{i \neq j} \left[(\vec{u}_j - \vec{u}_i) \kappa_H (|\vec{r}_j - \vec{r}_i|) \right] \tag{13}$$

with

$$\kappa_H(r) = \alpha \kappa(r) \tag{14}$$

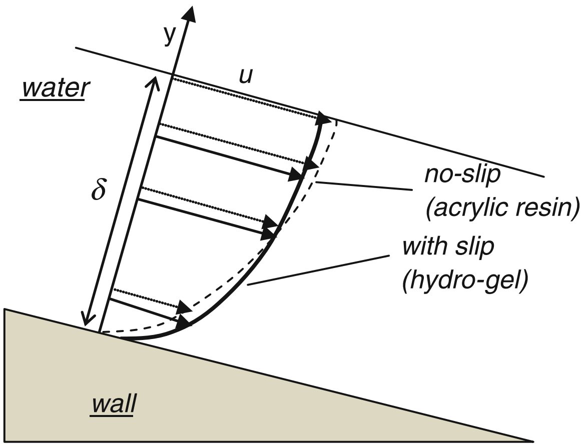


Fig. 7 Comparison of flow profiles near no-slip wall (*acrylic resin*) and slippery wall (*hydrogel*)

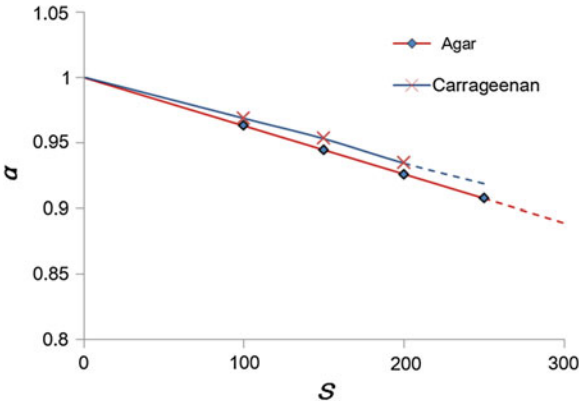


Fig. 8 Relationship of swelling degree S and slip ratio α

where index i denotes the water particle near hydro-gel wall and j the surface particle of hydro-gel wall. Namely, α is set effective only near the hydrogel wall, because the effect of slip is available near this area. The effective length of the above reduced weight function near the wall is assumed to be r_e in this study, and set to be $2.1l_{0,}$ where l_0 is the initial distance between the particles.

Summarizing the above procedure, (i) select S according to the hydrophilicity of the solid object, (ii) estimate α using Eq. (12), and (iii) apply α to the weight function of viscous term of Navier-Stokes equation for the calculation of shear force near the hydrogel wall using by Eqs. (13) and (14).

Next, we show the 2D splash simulation employing the above method. The effect of the slip ratio α on the flow around a hydrogel sphere can be taken care with Eq. (11). The comparison of the simulation result with $S = 100$ and the experimental

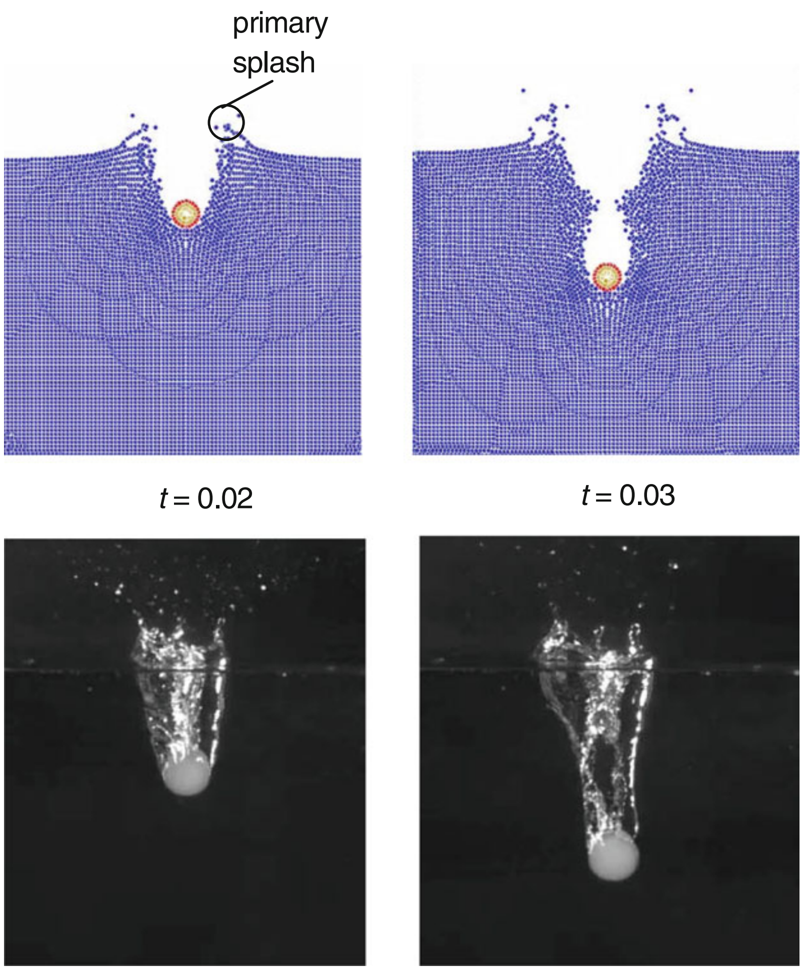


Fig. 9 Crown-type-splash of hydrogel ($S = 100$) by experiment and simulation, where primary splash ($t = 0.02$) and air cavity ($t = 0.03$) are shown

one is shown in Fig. 9, where the radius of sphere R is 10 mm and the initial height h is $50R$ in the both simulation and experiment. The water tank has the width of $20R$ and the depth of $20R$, where we confirmed the effect of the wall was negligible.

Assuming that the sphere touches the water surface at $t = 0$, the left figures in Fig. 9 are snapshots of the splashes at $t = 0.02$.

The first splash, which is created just after the sphere touches the surface of water is the so-called primary splash. It is seen that the sphere creates a cavity also. The pattern of the above crown-type splash and the air cavity obtained by the present simulation is similar to the experimental result. The above crown-type splash and the presence of the air cavity do not occur in the case of the acrylic resin sphere.

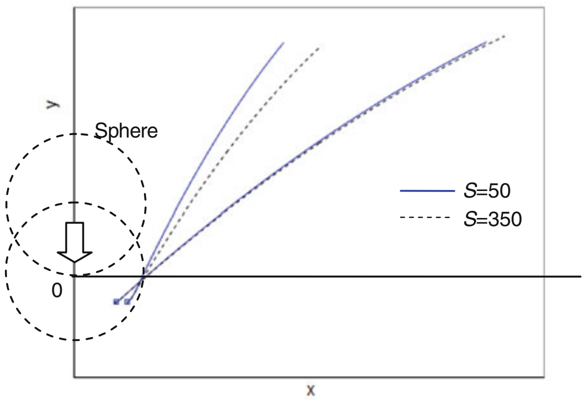


Fig. 10 Comparison between simulation results of representative path lines of water particles of primary splash for hydrogel spheres of different values of swelling parameter *S*

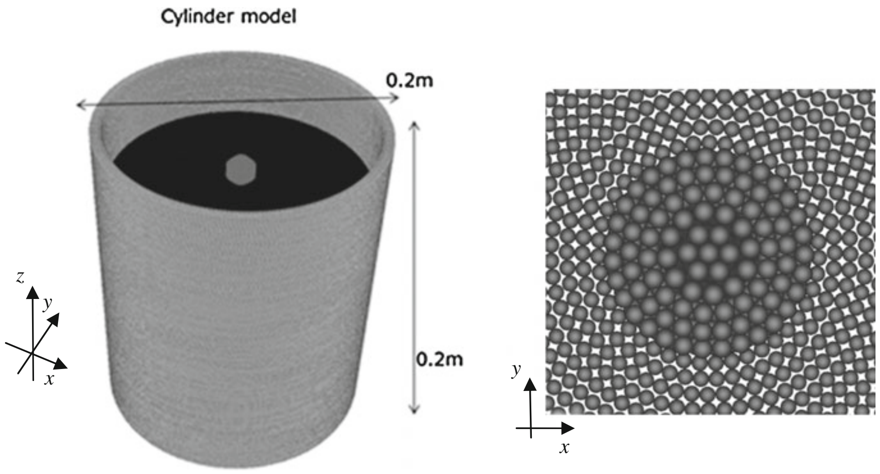


Fig. 11 Analysis domain for 3D splash simulation (*left figure*) and arrangement of particles of hydro-gel sphere and water viewed from the top (*right figure*)

Figure 10 shows the crown-type splashes and the representative path trajectories of particles for the different swelling ratios. The dotted solid lines are the path trajectories of particles when S is 50 or $\alpha = 0.94$, whereas the dotted lines are those when S is 350 or $\alpha = 0.7$. It is seen from the figure that the splashes spread widely with larger value of S or α, or the velocity of the water near the wall is larger with the swelling ratio, which causes the earlier exfoliation, creating the wider primary splash.

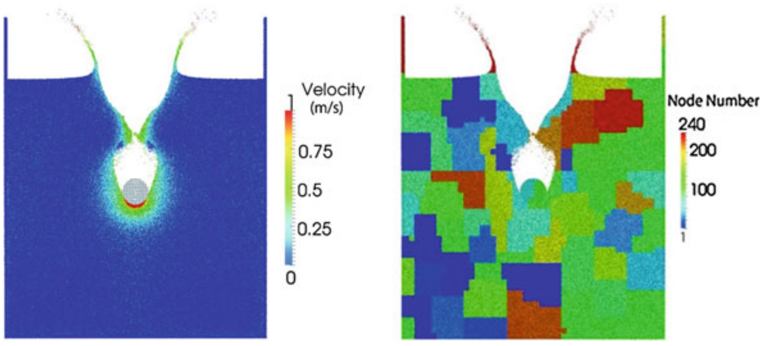


Fig. 12 3D splash (*left figure*) at t = 0.05 and domain decomposition (*right figure*)

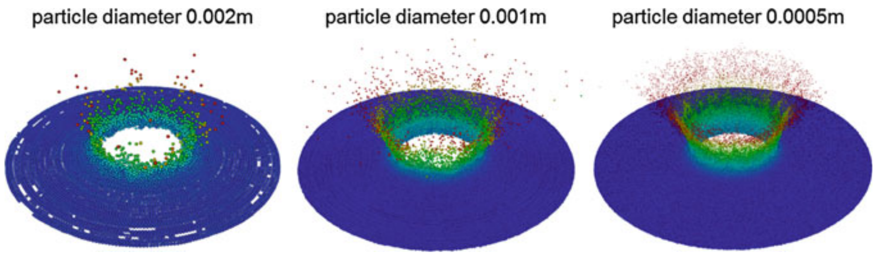


Fig. 13 Comparison of splash patterns with the different initial distances of water particles l_0 ($S = 100$, $r_e = 4.1$ and $t = 0.03\,\mathrm{s}$)

5 Extension to 3D Simulation with Large Scale Parallel Computing

It is considered that the 3D simulation makes it possible to observe more detailed behaviors of the splash. For example, the 3D simulation could allow us to calculate the crown-type splash with the finger [24] or the spike [18], which are impossible to realize with the 2D simulation. It is also expected that we can clarify the generation mechanism, including the number and the size of them, which depend on the Weber number, the rotational movement of the splash around an acrylic resin sphere, which is generated in the column-type splash and the texture at the wall of the air cavity created by the sinking sphere.

We perform here a large scale particle simulation using the distributed memory parallel computers, employing the two-level domain decomposition [31]. The first level domain decomposition is performed in order to keep the balance of the number of particles among nodes and the second level domain decomposition is performed in order to keep the balance of the number of particles among threads in each node.

The calculation flow is shown as follows;

1. A bounding box of a whole region is defined, and is filled with buckets. Since an influence radius of a particle is defined in the MPS method, the size of a bucket is set to be wider than the influence radius.
2. All the particles are embedded in the buckets.
3. The bucket-based domain decomposition is performed with an equal number of particles at each subdomain by ParMETIS [32].
4. If an imbalance in the number of particles among regions appears, the domain decomposition is performed again in order to recover the balance of the number of particles.

The parallel computer used here is the FX10 in the Information Technology Center of the University of Tokyo. The processor of the above FX10 is the SPARC64 IXfx, where a processor node has 16 cores of 1.848 GHz and 32 GB memory. In this research, the OpenMP is used for parallelization in each node and the MPI is used for parallelization among nodes.

Figure 11 shows the simulation setup of the crown-type splash in the case of the hydro-gel sphere.

The initial locations of particles are arranged concentrically, and the water tank is of a circular cylinder. Let the diameter of a particle and α be 0.0005 m and 0.4, respectively. Figure 12 is the result of the splash analysis at $t = 0.05$ s using 53 million particles. It took about 12 h for this analysis using 240 nodes of the FX10. The velocity of each particle is shown with color graduation in the left hand side of Fig. 12. The right hand side of Fig. 12 shows the time sequence of domain decomposition, which nodes are distinguished by different colors.

Figure 13 shows the simulation result by our 3D calculation, which are in good agreement with the experimental result as shown in Fig. 9, where the crown-type splash, the air cavity and the droplets scattering are expressed well. It is observed that the particle diameter, which is described as the initial distance of particle $l_{0,}$ influences the splash's pattern, namely, l_0 becomes smaller and the total number of particles is larger, the splash pattern is expressed clearer. The effect of the slip ratio on the splash pattern in 3D simulation is now under analyzing, though we are able to see the difference in the width and height of the splash pattern. In order to simulate the finger or the spike in the crown-type splash and its relation with the Weber number, we need larger scale computing yet.

6 Conclusion

1. Focusing on the treatments of the interface between the solid and the fluid, we propose a calculation method with the slip effect on the surface of a slimy material.
2. Experimental results of water splashes taken by a high-speed camera show that the splash pattern caused by an acrylic resin sphere is different from that caused by a hydro-gel sphere.

3. An engineering model to express the slimy surface, which the creatures living in water such as fish or frogs have, is proposed, where the slip ratio α, which is the reduction ratio of the shear stress near a solid wall obtained through the experiment, is introduced in the shear term of the Navier-Stokes equation.
4. The splash pattern calculated by the proposed method is in good agreement with the experimental result.
5. The above method for calculating the splash is applied to the large scale parallel computing in 3D, which depicts the more detailed splash patterns. As the future work, a larger scale computing and a modelling of the surface tension are needed to observe the finger or the spike as seen in the case of the milk-crown.

Acknowledgments This research was supported by the MEXT Program for the Strategic Research Foundation at Private Universities, 2012–2017 and the WCU (World Class University) Program through the Korea Science and Engineering Foundation funded by the Korean Ministry of Education, Science and Technology (R33-2008-000-10027-0).

References

1. Billah KY, Scanlan RH (1991) Resonance, tacoma narrows bridge failure, and undergraduate physics textbooks. Am J Phys 59(2):118–124
2. Davies JM (1949) The aerodynamics of golf balls. J Appl Phys 20(9):821–828
3. Kodama Y, Kakugawa A, Takahashi T, Kawashima H (2000) Experimental study on microbubbles and their applicability to ships for skin friction reduction. Int J Heat Fluid Flow 21(5):582–588
4. Étienne S, Pelletier D (2005) A general approach to sensitivity analysis of fluid-structure interactions. J fluids struct 21(2):169–186
5. He T, Zhou D, Bao Y (2012) Combined interface boundary condition method for fluid-rigid body interaction. Comput Methods Appl Mech Eng 223:81–102
6. Pendergast DR, Mollendorf JC, Cuviello R, Termin AC (2006) Application of theoretical principles to swimsuit drag reduction. Sports Eng 9(2):65–76
7. Moria H, Chowdhury H, Alam F, Subic A, Smits AJ, Jassim R, Bajaba NS (2010) Contribution of swimsuits to swimmer's performance. Procedia Eng 2(2):2505–2510
8. Fletcher NH (1976) Sound production by organ flue pipes. J Acoust Soc Am 60:926
9. Coltman JW (1968) Sounding mechanism of the flute and organ pipe. J Acoust Soc Am 44:983
10. Tsuchida J, Fujisawa T, Yagawa G (2006) Direct numerical simulation of aerodynamic sounds by a compressible cfd scheme with node-by-node finite elements. Comput Methods Appl Mech Eng 195(13):1896–1910
11. Maruyama T (1999) Surface and inlet boundary conditions for the simulation of turbulent boundary layer over complex rough surfaces. J Wind Eng Ind Aerodyn 81(1):311–322
12. Yabe T, Chinda K, Hiraishi T (2007) Computation of surface tension and contact angle and its application to water strider. Comput Fluids 36(1):184–190
13. Kobayashi S, Watanabe R, Oiwa T, Morikawa H (2009) Computational study of micropropulsion mechanism in water modeled on flagellum with projecting mastigonemes. J Biomech Sci Eng 4(1):11–22
14. Nomura K, Koshizuka S, Oka Y, Obata H (2001) Numerical analysis of droplet breakup behavior using particle method. J Nucl Sci Technol 38(12):1057–1064
15. Caboussat A (2006) A numerical method for the simulation of free surface flows with surface tension. Comput Fluids 35(10):1205–1216

16. Liu J, Koshizuka S, Oka Y (2005) A hybrid particle-mesh method for viscous, incompressible, multiphase flows. J Comput Phys 202(1):65–93
17. Worthington AM (1882) On impact with a liquid surface. Proc R Soc Lond 34(220–223):217–230
18. Krechetnikov R, Homsy GM (2009) Crown-forming instability phenomena in the drop splash problem. J Colloid Interface Sci 331(2):555–559
19. Akers B, Belmonte A (2006) Impact dynamics of a solid sphere falling into a viscoelastic micellar fluid. J Nonnewton Fluid Mech 135(2):97–108
20. Duez C, Ybert C, Clanet C, Bocquet L (2007) Making a splash with water repellency. Nat phys 3(3):180–183
21. Yoon SS, Jepsen RA, Nissen MR, O'Hern TJ (2007) Experimental investigation on splashing and nonlinear fingerlike instability of large water drops. J Fluids Struct 23(1):101–115
22. Idelsohn SR, Onate E, Del Pin F (2003) A lagrangian meshless finite element method applied to fluid-structure interaction problems. Comput struct 81(8):655–671
23. Ling SC, Ling TYJ (1974) Anomalous drag-reducing phenomenon at a water/fish-mucus or polymer interface. J Fluid Mech 65(03):499–512
24. Kubota Y, Mochizuki O (2009) Splash formation by a spherical object plunging into water. J Vis 12:339–345
25. Koshizuka S (1995) A particle method for incompressible viscous flow with fluid fragmentation. Comput Fluid Dynamics J 4:29–46
26. Eddington DT, Beebe DJ (2004) Flow control with hydrogels. Adv Drug Deliv Rev 56(2):199–210
27. Beebe DJ, Moore JS, Bauer JM, Yu Q, Liu RH, Devadoss C, Jo BH (2000) Functional hydrogel structures for autonomous flow control inside microfluidic channels. Nature 404(6778):588–590
28. Alupei IC, Popa M, Hamcerencu M, Abadie MJM (2002) Superabsorbant hydrogels based on xanthan and poly (vinyl alcohol): 1. the study of the swelling properties. Eur Polymer J 38(11):2313–2320
29. Narayanan J, Xiong JY, Liu XY (2006) Determination of agarose gel pore size: absorbance measurements vis a vis other techniques. J Phys: Conf Ser 28(1):83 (IOP Publishing)
30. Kikuchi K, Mochizuki O (2010) A flow on a hydrogel surface mimicked a living cell. In: Proceedings of the 21st international symposium on transport phenomena in Kaohsiung city, Taiwan
31. Yagawa G, Shioya R (1994) Parallel finite elements on a massively parallel computer with domain decomposition, 4. Comput Syst Eng 4:495–503
32. Karypis G, Kumar V (1998) A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput 20(1):359–392

Part IV
Reduced-Order Models

Reduced-Order Modelling Strategies for the Finite Element Approximation of the Incompressible Navier-Stokes Equations

Joan Baiges, Ramon Codina and Sergio R. Idelsohn

Abstract In this chapter we present some Reduced-Order Modelling methods we have developed for the stabilized incompressible Navier-Stokes equations. In the first part of the chapter, we depart from the stabilized finite element approximation of incompressible flow equations and we build an explicit proper-orthogonal decomposition based reduced-order model. To do this, we treat the pressure and all the non-linear terms in an explicit way in the time integration scheme. This is possible due to the fact that the reduced model snapshots and basis functions do already fulfill an incompressibility constraint weakly. This allows a hyper-reduction approach in which only the right-hand-side vector needs to be reconstructed. In the second part of the chapter we present a domain decomposition approach for reduced-order models. The method consists in restricting the reduced-order basis functions to the nodes belonging to each of the subdomains. The method is extended to the particular case in which one of the subdomains is solved by using the high-fidelity, full-order model, while the other ones are solved by using the low-cost, reduced-order equations.

1 Introduction

Reduced-order models (ROM) are nowadays receiving a lot of interest from the computational mechanics community. Their most attractive feature is the capability of reproducing the response of complex physical phenomena through the solution of systems of equations which involve only very few degrees of freedom.

J. Baiges (✉) · R. Codina · S. R. Idelsohn
Centre Internacional de Mètodes Numèrics a l'Enginyeria (CIMNE), Edifici C1, Campus Nord UPC C/ Gran Capità S/N, 08034 Barcelona, Spain
e-mail: jbaiges@cimne.upc.edu

J. Baiges, R. Codina
Universitat Politècnica de Catalunya, Jordi Girona 1-3, Edifici C1, 08034 Barcelona, Spain

S. R. Idelsohn
Instituciò Catalana de Recerca i Estudis Avançats (ICREA), Barcelona, Spain

Amongst the various families of reduced-order models, Proper Orthogonal Decomposition [14, 23, 26] based ROMs consist in the training of the model by taking snapshots from a high-fidelity simulation and using them to build an orthogonal basis which is capable of accurately representing the solution through the combination of few of this basis functions. Particularly, we are interested in the application of POD models to the incompressible Navier-Stokes equations, which we originally approximate by using finite elements and a stabilized formulation. The problem of applying POD models to the incompressible flow equations has been approached by several authors [12, 19, 20, 25, 29, 39, 40] in a range of applications like shape optimization [1, 9, 27, 34] or flow control [3, 21, 32].

The major concerns when making use of a reduced-order model are, on the one hand, computational cost, and on the other, accuracy. Obviously we are looking for a cheap reduced-order model which is as accurate as possible. Unfortunately, this is not always possible. In this chapter we present some approaches we have developed for POD models for the incompressible Navier-Stokes equations which help enhance the computational cost and accuracy of the reduced-order models.

One of the a priori drawbacks of traditional POD is that a straightforward application of a POD strategy to a non-linear problem does not turn out in a drastic reduction of the required computational cost. This is so because in order to solve the reduced-order equations, the full-order, non-linear, system of equations needs to be built first and then projected onto the reduced-order space. Recently, the so-called hyper-reduction [4, 8, 15, 22, 31, 35–38] has appeared as a means to circumvent this problem. The main idea is to compute the non-linear system entries only at some few nodes of the computational mesh, and then approximate the whole system by extrapolating it from the values of the system at these entries.

In the first part of this chapter, we describe a reduced-order model which is particularly suitable for hyper-reduction [7]. The basic idea is to build a reduced-order model based on a proper orthogonal decomposition and a Galerkin projection and treat all the terms in an explicit way in the time integration scheme. This results in a reduced-order model where only the right-hand side of the system needs to be rebuilt at each time step. This is possible because the reduced model snapshots do already fulfill the stabilized continuity equation and the pressure field can be automatically recovered at the end of each time step from the reduced order basis and solution coefficients. We also present a method for choosing the sampling entries from which the complete system is going to be extrapolated. The method consists of choosing the sampling points such that the distance between the right-hand-side snapshots and the recovered snapshots is minimized, with the restriction that the coordinates of sampling points must coincide with the coordinates of the finite element mesh nodes.

Another issue which needs to be dealt with when using reduced-order models is the lack of robustness with respect to changes in the parameters which characterize the numerical simulation. This lack of robustness causes the reduced model to be valid only in a small parameter region close to the parameter values for which the reduced model was built [3], requiring the snapshot collection and the reduced model to be updated when an optimization process leads to a parameter configuration which becomes too separated from the starting parameter set.

In the second part of this chapter we present a domain decomposition method for reduced-order models [6] which we apply to the finite element approximation of the incompressible Navier-Stokes equations. Domain decomposition methods for reduced-order models have been used for different simulation problems [2, 10, 28, 30, 33, 41]. In these partitioned approaches reduced-order models are formulated independently and then glued together in either a monolithic or an iterative way. Contrary to this approach, the domain decomposition method we propose is obtained simply by restricting the reduced model basis functions to be non-null only in the nodes of the computational mesh belonging to the considered subdomain. This definition of the partitioned problem directly ensures the continuity of the recovered reduced-order solution at the interfaces. Also, there is no need to use the classical domain-decomposition iteration by subdomain schemes, because the Domain Decomposition Reduced-Order Model (DD-ROM) is written in terms of the partitioned reduced bases in a monolithic way. One of the advantages of the proposed method is the ease for generating a hybrid full-order/reduced-order model, as a particular case of the general DD-ROM method. The proposed hybrid DD-ROM model can be easily used together with hyper-reduced models, as we demonstrate in the numerical examples section.

The chapter is organized as follows. In Sect. 2 we present an explicit ROM for the finite element approximation of the incompressible Navier-Stokes equations, and a numerical example illustrates the behavior of the model for low Reynolds flow cases. In Sect. 3 we describe the hyper-reduction strategy we apply to the explicit ROM, and we also present the Discrete Best Points Interpolation Method for the selection of sampling indices for the gappy-POD reconstruction process. Finally, the domain-decomposition reduced-order model is presented in Sect. 4, where we also explain the hybrid full-order/reduced-order domain decomposition approach and present a numerical example. Some conclusions close the chapter in Sect. 5.

2 An Explicit Reduced-Order Model for the Incompressible Navier-Stokes Equations

When solving a non-linear problem by means of a POD based ROM, it is necessary to project the full-order system of equations to the reduced-order space at each iteration of the non-linear problem. For non-linear problems, this is troublesome because the expected orders of magnitude reduction in the computational cost of solving the reduced-order system is not observed in practice: the computational time of the reduced-order model is governed by the need of rebuilding the full-order system at each iteration, and then projecting it to the reduced-order subspace.

This issue has motivated a lot of research recently, leading to several strategies which reduce the cost of computing projected reduced-order system of equations

[4, 8, 15, 22, 31, 35–38]. These approaches are known as hyper-reduced models. In these methods, the non-linear and parameter-dependent terms are recovered by means of a least-squares procedure from a series of sampling points where the function to be approximated is computed. This allows to effectively reduce the amount of computations required to build the reduced order system, and results in a reduced-order model whose computational cost is directly proportional to its number of degrees of freedom.

We have been working in a hyper-reduced approach for the incompressible Navier-Stokes equations. The particularity of the strategy we propose is that the equations for the reduced-order model are treated in an explicit way. This allows to send all the non-linear terms to the right hand-side of the reduced-order system, leaving in the left-hand side only the mass matrix due to the temporal derivatives. The main advantage is, of course, that the mass matrix is linear, and the hyper-reduced approaches need only to be applied to right-hand side vector. This effectively reduces the overall cost of the reduced-order model.

Let us start by introducing some notation for the POD approximation of a general problem. Let $\boldsymbol{U} \in \mathbb{R}^M$ be the global unknown vector associated to a non-linear variational problem. Suppose that after linearizing and fully discretizing in time and space the given problem, the following matrix form is obtained which allows to obtain the vector of nodal unknowns $\boldsymbol{U}$ at a given iteration of the non-linear procedure, for a certain time step:

$$\boldsymbol{A}\boldsymbol{U} = \boldsymbol{F}, \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{M\times M}$ is the matrix of the system whose solution is $\boldsymbol{U}$, and $\boldsymbol{F} \in \mathbb{R}^M$ the RHS vector. The POD approximation of the previous system is obtained by projecting it onto a low dimensional subspace $\mathcal{U} \subset \mathbb{R}^N$. Vectors $\boldsymbol{U}$ are now approximated by:

$$\boldsymbol{U} \approx \boldsymbol{\Phi}\boldsymbol{\alpha}, \tag{2}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{M\times N}$ is the basis for $\mathcal{U}$ and N is the dimension of the reduced order model, with $N < M$. $\boldsymbol{\alpha} \in \mathbb{R}^N$ are the components in $\mathcal{U}$ expressed in the reference system defined by $\boldsymbol{\Phi}$. The reduced-order basis $\boldsymbol{\Phi}$ is obtained by means of the POD method [14, 23, 26], that is by doing the singular value decomposition of a set of solution snapshots, which in our case are taken from the results of a full-order simulation. After projecting the full-order system to this reduced-order subspace and applying a least squares approach, the final reduced-order system is:

$$\boldsymbol{\Phi}^T\boldsymbol{A}\boldsymbol{\Phi}\boldsymbol{\alpha} = \boldsymbol{\Phi}^T\boldsymbol{F}. \tag{3}$$

2.1 Stabilized Finite Element Approximation of the Incompressible Navier-Stokes Equations

In this section we summarize the finite element stabilized formulation for the incompressible Navier-Stokes equations used in the rest of the chapter. Let us consider the transient incompressible Navier-Stokes equations, which consist of finding $\boldsymbol{u} : \Omega \times (0, T) \longrightarrow \mathbb{R}^d$ and $p : \Omega \times (0, T) \longrightarrow \mathbb{R}$ such that:

$$\begin{aligned} \partial_t \boldsymbol{u} - \nu \Delta \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \nabla p = \boldsymbol{f} \quad & \text{in } \Omega, \\ \nabla \cdot \boldsymbol{u} = 0 \quad & \text{in } \Omega, \\ \boldsymbol{u} = \bar{\boldsymbol{u}} \quad & \text{on } \Gamma_D, \\ -p\boldsymbol{n} + \nu \boldsymbol{n} \cdot \nabla \boldsymbol{u} = \boldsymbol{0} \quad & \text{on } \Gamma_N. \end{aligned}$$

for $t > 0$, where $\partial_t \boldsymbol{u}$ is the local time derivative of the velocity field. $\Omega \subset \mathbb{R}^d$ is a bounded domain, with $d = 2, 3$, ν is the viscosity, and $\boldsymbol{f}$ the given source term. Appropriate initial conditions have to be appended to this problem.

Let now $V = H^1(\Omega)^d$, and $V_0 = \{\boldsymbol{v} \in V | \boldsymbol{v} = \boldsymbol{0} \text{ on } \Gamma_D\}$. Let also $Q = L^2(\Omega)$ and $\mathcal{D}'(0, T; Q)$ be the distributions in time with values in Q. The variational problem consists of finding $[\boldsymbol{u}, p] \in L^2(0, T; V) \times \mathcal{D}'(0, T; Q)$ such that:

$$(\boldsymbol{v}, \partial_t \boldsymbol{u}) + B([\boldsymbol{v}, q], [\boldsymbol{u}, p]) = \langle \boldsymbol{v}, \boldsymbol{f} \rangle \quad \forall [\boldsymbol{v}, q] \in V \times Q, \tag{4}$$

with

$$\boldsymbol{u} = \bar{\boldsymbol{u}} \quad \text{on } \Gamma_D,$$

where

$$B([\boldsymbol{v}, q], [\boldsymbol{u}, p]) := \langle \boldsymbol{v}, \boldsymbol{u} \cdot \nabla \boldsymbol{u} \rangle + \nu(\nabla \boldsymbol{v}, \nabla \boldsymbol{u}) - (p, \nabla \cdot \boldsymbol{v}) + (q, \nabla \cdot \boldsymbol{u}).$$

Here, $(\cdot, \cdot)$ stands for the $L^2(\Omega)$ inner product and $\langle \cdot, \cdot \rangle$ for the integral of the product of two functions, not necessarily in $L^2(\Omega)$. Let $\{K\}$ be a finite element partition of Ω, from which we construct the finite element spaces $V_h \subset V$, $V_{h0} \subset V_0$, $Q_h \subset Q$. The semilinear form B suffers from the well-known stability issues due to the convective nature of the flow, but also requires a compatibility between the velocity and pressure approximation spaces due to the classical LBB inf-sup condition. In order to deal with these stability issues, we use a stabilized finite element formulation [16], which is as follows: for each t, find $\boldsymbol{u}_h(t) \in V_h$, $p_h(t) \in Q_h$ such that:

$$\begin{aligned} (\boldsymbol{v}_h, \partial_t \boldsymbol{u}_h) + B([\boldsymbol{v}_h, q_h], [\boldsymbol{u}_h, p_h]) + \sum_K \tau_K (\boldsymbol{u}_h \cdot \nabla \boldsymbol{v}_h + \nu \Delta \boldsymbol{v}_h \\ + \nabla q_h, \boldsymbol{r}([\boldsymbol{u}_h, p_h]))_K = \langle \boldsymbol{v}_h, \boldsymbol{f} \rangle, \end{aligned} \tag{5}$$

for all $\boldsymbol{v}_h \in V_{h,0}$, $q_h \in Q_h$. Initial conditions need to be appended to this problem. In (5):

$$\boldsymbol{r}([\boldsymbol{u}_h, p_h]) = \partial_t \boldsymbol{u}_h - \nu\Delta\boldsymbol{u}_h + \boldsymbol{u}_h \cdot \nabla\boldsymbol{u}_h + \nabla p_h - \boldsymbol{f}, \tag{6}$$

is the residual of the momentum equation, $(\cdot, \cdot)_K$ is used to denote the L^2 product in element K and τ_K is the stabilization parameter:

$$\tau_K = \left(c_1 \frac{\nu}{h^2} + c_2 \frac{|\boldsymbol{u}_h|_K}{h} \right)^{-1},$$

where $|\boldsymbol{u}_h|_K$ is the mean velocity modulus in element K, h is the element size and c_1 and c_2 are stabilization constants.

Regarding the discretization in time, we consider implicit integration schemes. For the full-order system, only implicit time integration schemes can be used, because no time derivatives of the pressure appear in the equations. Taking this into account, we can do the following: supposing that the velocity and pressure at time step n $[\boldsymbol{u}_h^n, p_h^n]$ are known, we may solve (5) for example with $\partial_t \boldsymbol{u}_h$ being discretized using a backward differences in time scheme:

$$\begin{aligned} &\partial_t \boldsymbol{u}_h \approx \delta_t \boldsymbol{u}_h^{n+1}, \\ &\delta_t \boldsymbol{u}_h^{n+1} := \begin{cases} \frac{1}{\delta t}(\boldsymbol{u}_h^{n+1} - \boldsymbol{u}_h^n) & \text{1st order scheme} \\ \frac{1}{\delta t}(\frac{3}{2}\boldsymbol{u}_h^{n+1} - 2\boldsymbol{u}_h^n + \frac{1}{2}\boldsymbol{u}_h^{n-1}) & \text{2nd order scheme} \end{cases} \end{aligned} \tag{7}$$

where δt is the time step size.

2.2 Explicit Reduced-Order Model

As explained in the previous sections, it is convenient to treat the reduced-order model by using an explicit time integration scheme, because this leads to important computational gains when using hyper-reduced order reconstruction methods. However, we have also explained the need of using an implicit time integration scheme for the full-order model of the incompressible Navier-Stokes equations, due to the presence of the pressure field. Here we summarize the strategy we use for building an explicit reduced-order model which is suitable for the incompressible Navier-Stokes equations [7].

Let us start by introducing the velocity and pressure reduced-order subspaces. $\hat{Q} \subset Q_h$ is the pressure subspace defined by the pressure part of the POD basis functions $\boldsymbol{\Phi}$, $\hat{p} \in \hat{Q}$ is the reduced-order pressure field. $\hat{V} \subset V_h$ is the velocity subspace defined by the velocity part of the POD basis functions $\boldsymbol{\Phi}$. For each time t, $\hat{\boldsymbol{u}}(t) \in \hat{V}$ is the reduced-order velocity. In order to develop the explicit reduced-order model where the pressure is treated in an explicit way, we take into account that:

- All reduced basis functions do already fulfill the stabilized continuity equation. Since the reduced-order basis is built from weakly incompressible solution snapshots and the incompressibility constraint is linear, the reduced basis functions (and their linear combinations) do also fulfill it.
- If basis functions are taken to be joint velocity-pressure basis functions (that is $\boldsymbol{\Phi}$ contains the coefficients of functions in $\hat{V} \times \hat{Q}$), then the pressure at time step $n+1$ is automatically recovered from coefficients $\boldsymbol{\alpha}_{n+1}$ and the reduced order basis $\boldsymbol{\Phi}$ even if all the terms involving the pressure are treated in an explicit way in the reduced order formulation.

The variational formulation for the first order in time reduced-order model that we propose is:

$$\begin{aligned}&(\hat{\boldsymbol{v}}, \delta_t \hat{\boldsymbol{u}}^{n+1}) + (\hat{\boldsymbol{v}}, \hat{\boldsymbol{u}}^{n*} \cdot \nabla \hat{\boldsymbol{u}}^{n*}) + \nu(\nabla \hat{\boldsymbol{v}}, \nabla \hat{\boldsymbol{u}}^{n*}) - (\hat{p}^{n*}, \nabla \cdot \hat{\boldsymbol{v}}) \\ &+ \sum_K \tau_K (\hat{\boldsymbol{u}}^n \cdot \nabla \hat{\boldsymbol{v}} + \nu \Delta \hat{\boldsymbol{v}}, \delta_t \hat{\boldsymbol{u}}^n - \nu \Delta \hat{\boldsymbol{u}}^n + \hat{\boldsymbol{u}}^n \cdot \nabla \hat{\boldsymbol{u}}^n + \nabla \hat{p}^n - \boldsymbol{f}^n)_K = \langle \hat{\boldsymbol{v}}, \boldsymbol{f}^n \rangle.\end{aligned} \tag{8}$$

where the terms $\hat{\boldsymbol{u}}^{n*}$ and $\hat{p}^{n*}$ are a second order approximation of the state at $n+1$ (the velocity and the pressure at $n+1$) given by:

$$\begin{aligned}\hat{\boldsymbol{u}}^{n*} &= 2\hat{\boldsymbol{u}}^n - \hat{\boldsymbol{u}}^{n-1}, \\ \hat{p}^{n*} &= 2\hat{p}^n - \hat{p}^{n-1}.\end{aligned} \tag{9}$$

In the case of the second order in time reduced-order model, we use the same variational formulation (8), but the terms $\hat{\boldsymbol{u}}^{n*}$ and $\hat{p}^{n*}$ are now a third order approximation of the state at $n+1$ given by:

$$\begin{aligned}\hat{\boldsymbol{u}}^{n*} &= \frac{12}{5}\hat{\boldsymbol{u}}^n - \frac{9}{5}\hat{\boldsymbol{u}}^{n-1} + \frac{2}{5}\hat{\boldsymbol{u}}^{n-2}, \\ \hat{p}^{n*} &= \frac{12}{5}\hat{p}^n - \frac{9}{5}\hat{p}^{n-1} + \frac{2}{5}\hat{p}^{n-2}.\end{aligned} \tag{10}$$

Note that for the first order explicit scheme we propose to use the second order extrapolation (9), and for the second order scheme the third order extrapolation (10).

The key point of this formulation is that only the temporal derivative terms involve values of the reduced-order velocity or pressures at the new time step. This ensures that the resulting reduced-order matrix is linear. However, the reduced-order right-hand-side still needs to be approximated. After solving the reduced-order system, the velocity and pressure fields at $n+1$ can be recovered by multiplying the reduced-order basis $\boldsymbol{\Phi}$ by the obtained reduced-order components $\boldsymbol{\alpha}_{n+1}$.

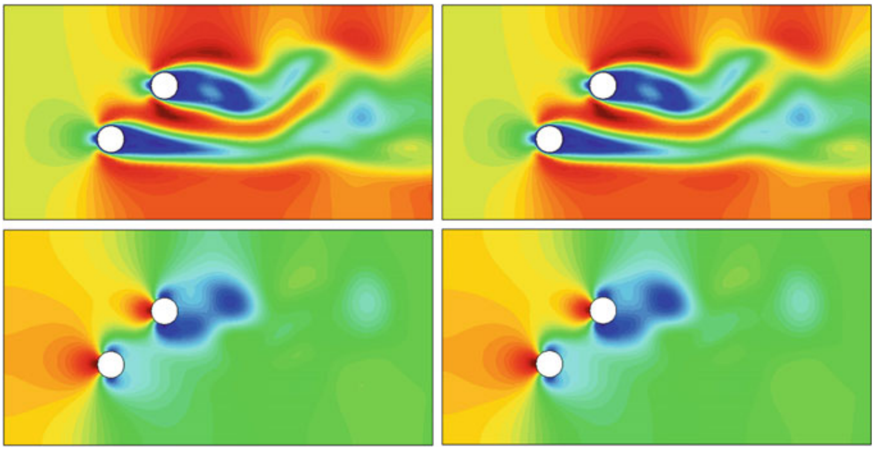


Fig. 1 Comparison of the FOM (*left*) and ROM (*right*) velocities (*top*) and pressures (*bottom*) after 400 time steps of simulation

2.3 Numerical Example. Bidimensional Flow Past Two Cylinders

The first numerical example consists in the bidimensional flow past two cylinders. The computational domain is a 16×8 rectangle. The cylinders are centered at coordinates $(3, 3)$ and $(6, 5)$, and both of them are of diameter 1. The inflow velocity is 1, which together with the density $\rho = 1$ and the viscosity $\mu = 0.01$ results in a Reynolds number $Re = 100$. The time step is set to $\delta t = 0.1$. The mesh is composed of 7310 linear triangular elements. After running the full-order simulation and taking the corresponding snapshots, the explicit reduced-order model is run. The number of degrees of freedom for the ROM is only 10.

Figure 1 shows a comparison of the velocity and pressure fields for the full-order and the explicit reduced-order model after 400 time steps of simulation. The high-fidelity and the reduced-order fields are very similar. In Fig. 2 we compare the time history and Fourier transform of the vertical velocity and the pressure at coordinates $(8.5, 4)$. We observe that the time history and Fourier transform of both vertical velocity and pressure are accurate for the reduced-order model. The cpu-time for running the full-order model is 53.24 s, the time for running the explicit reduced-order model is 19.78 s, a 63 % reduction in computational time.

3 Hyper-Reduction Approach

At this point, we already have an explicit reduced-order model in which all the non-linear terms are in the right-hand-side vector and the reduced system matrix is linear and does not change between time steps. However, computing the right-hand-side

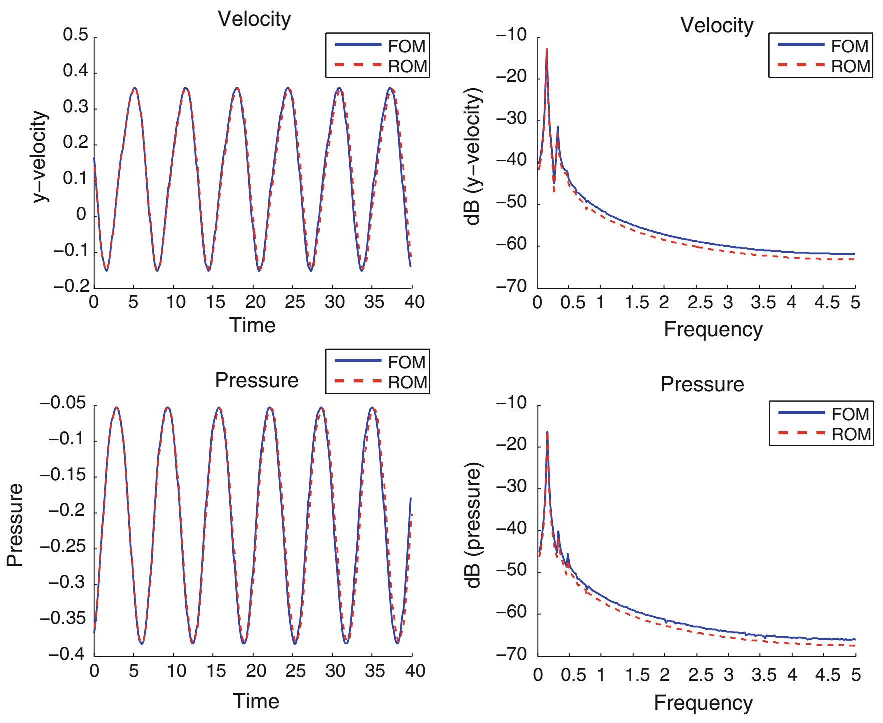


Fig. 2 Comparison of the FOM and ROM velocity and pressure time history at the point (8.5, 4) (*left*) and their Fourier transform (*right*)

vector at each time step is still expensive (number of operations of $\mathcal{O}(M)$), because we need to recompute $\boldsymbol{F}^{n+1}$ and then project it to the reduced-order subspace by calculating $\boldsymbol{\Phi}^T \boldsymbol{F}^{n+1}$. The approach we follow for reducing this computational cost is to reconstruct the non-linear vector $\boldsymbol{F}^{n+1}$by sampling only some of the entries of this vector and applying a lest-squares minimization strategy . The method we follow was first presented in [18], and a similar approach has been recently used in [13] applied to an implicit reduced-order method for the incompressible Navier-Stokes equations.

Let us consider a reduced order basis for the right-hand-side vectors $\boldsymbol{F}$, $\boldsymbol{\Phi}_F$, obtained by means of a proper orthogonal decomposition of a set of snapshots for $\boldsymbol{F}$. $\boldsymbol{\Phi}_F$ defines a low-dimensional subspace $\mathcal{F} \subset \mathbb{R}^M$, so that any right-hand-side vector $\boldsymbol{F}$ can be approximated as:

$$\boldsymbol{F} \simeq \boldsymbol{\Phi}_F \boldsymbol{F}_\Phi,$$

where now $\boldsymbol{F}_\Phi \in \mathbb{R}^N$ are the reduced-order coefficients for the reconstruction. Let us also consider that we only know the nodal values for $\boldsymbol{F}$ at some sampling components $F_{i(k)}$, $1 \le k \le n_s$, where n_s is the number of sampling components of the vector,

$i(k)$ denotes the kth sampling component. We now want to recover the reduced order basis coefficients $\boldsymbol{F}_\Phi$ of the reduced order basis $\boldsymbol{\Phi}_F$ for vector $\boldsymbol{F}$.

In order to recover $\boldsymbol{F}_\Phi$ we can solve the least-squares minimization problem:

$$\boldsymbol{F}_\Phi = \arg \min_{\boldsymbol{a}\in\mathbb{R}^N} \sum_{k=1}^{n_s} \sum_{j=1}^{N} (\Phi_{F,i(k)j} a_j - F_{i(k)})^2, \tag{11}$$

where $\Phi_{F,i(k)j}$ denotes the basis vector j evaluated at the kth sampling component, $i(k)$.

The previous procedure provides the tools required to extrapolate the right-hand-side vector arising from the finite element problem. The main advantage is that in order to do so, only the nodal values at certain few sampling components are needed. If n_s is $\mathcal{O}(N)$, then the computational cost of rebuilding $\boldsymbol{F}^{n+1}$ for solving each time step is reduced to $\mathcal{O}(N)$, and the overall cost of the reduced-order model is $\mathcal{O}(N)$.

3.1 A Discrete Version of the Best Points Interpolation Method (DBPM)

When using hyper-reduced order models the quality of the recovered right-hand-side vector highly depends on the selected sampling components. Several strategies have been developed for choosing these sampling components [5, 8, 17]. Amongst the most extensively used are the *Discrete Empirical Interpolation Method (DEIM)* [15], where the sampling components are selected iteratively by imposing that the error growth at each iteration is limited and the *Best Points Interpolation Method (BPIM)* approach presented in [31], where the sampling points are chosen so that the distance between the projection of the *right-hand-side snapshots* onto the reduced basis subspace and the recovered right-hand-side is minimized.

The strategy we use, presented in [7], is a hybrid between the BPIM and the DEIM. We call it a *Discrete version of the Best Point Interpolation Method (DBPIM).* Similarly to the BPIM, the method consists of minimizing the error between the recovered right-hand-side vector snapshots and the actual snapshots. However, in the strategy we use we force the sampling coordinates to coincide with nodal points of the finite element mesh. Plus, once a component associated to a node of the finite element mesh is selected, all the degrees of freedom associated to that node are included in the sampling selection. Moreover, due to the lack of smoothness of the vectors which are being approximated we do not use a Marquardt related strategy in order to advance to the optimal set of sampling nodes. Instead, we use an algorithm which advances from one set of sampling nodes to the next one by evaluating the error of the recovered snapshots at the neighbour points in the finite element mesh and replaces a sampling node with its neighbour if the error diminishes. The DBPIM algorithm is detailed in Algorithm 1 for a scalar unknown (where each sampling node is associated to a single sampling component).

The first step of the DBPIM algorithm consists of finding the projection $\Pi_{\mathcal{F}}$ of the snapshots onto the reduced order subspace defined by the reduced order basis, $\mathcal{F}$. For each snapshot, this yields the coefficients $\boldsymbol{F}_{\Phi}^{\alpha}$. In the second step we choose an initial set of sampling nodes, which can be done by using the DEIM method. If the DEIM method is used, it will give us a set of sampling components. For a scalar problem, each component corresponds to a node of the finite element mesh. If the unknown is a vector field, then the nodes associated to the DEIM sampling components are selected as initial sampling nodes, and the number of sampling nodes is equal to the number of reduced basis functions. Otherwise, we always choose the number of sampling nodes to be equal to the number of basis functions times an (usually low) integer. After defining the initial set of sampling nodes, the degree(s) of freedom associated to these sampling nodes become sampling components. For this initial set of sampling nodes, we recover the approximated coefficients $\boldsymbol{F}_{\Phi}^{\alpha,\text{aprox}}$ by means of the previously described least-squares strategy. The error associated to a set of sampling components $\boldsymbol{i} \in \mathbb{N}^{n_s}$, whose k-th component is indicated as $i(k) \in 1, ..., M$, is obtained by computing the difference between the exact and the approximated $\boldsymbol{F}_{\Phi}^{\alpha}$ coefficients:

$$e(\boldsymbol{i}) = \sum_{\alpha=1}^{N\text{snapshots}} ||\boldsymbol{F}_{\Phi}^{\alpha,\text{aprox}}(\boldsymbol{i}) - \boldsymbol{F}_{\Phi}^{\alpha}|| \tag{12}$$

Algorithm 1 Discrete best points interpolation method

```
Compute the optimal basis coefficients for the snapshot set:
$\boldsymbol{\Phi}_F \boldsymbol{F}_{\Phi}^{\alpha} = \Pi_{\mathcal{F}}(\boldsymbol{F}^{\alpha}),\ \alpha = 1, N_{\text{snapshots}}$
Choose an initial set of sampling components $\boldsymbol{i} \in \mathbb{N}^{n_s} \mid 1 \le i(k) \le M,\ k = 1, ...n_s$.
Solve: $\boldsymbol{F}_{\Phi}^{\alpha,\text{aprox}}(\boldsymbol{i}) = \arg\min_{\boldsymbol{a}\in\mathbb{R}^N} \sum_{k=1}^{n_s} \sum_{j=1}^{N} (\Phi_{F,i(k)j} a_j - F_{i(k)}^{\alpha})^2,\ \alpha = 1, N_{\text{snapshots}}$
$e(\boldsymbol{i}) = \sum_{j=1}^{N\text{snapshots}} ||\boldsymbol{F}_{\Phi}^{\alpha,\text{aprox}}(\boldsymbol{i}) - \boldsymbol{F}_{\Phi}^{\alpha}||^2$
while set of sampling points has changed do
  for $m = 1 : n_s$ do
    for $l = 1 : N_{\text{neigh}}(i(m))$ do
      if the $l$th neighbour of $i(m)$ has not been previously tested then
        Temporarily replace sampling node $i(m)$ by its $l$th neighbour
        Solve: $\boldsymbol{F}_{\Phi}^{\alpha,\text{aprox}}(\boldsymbol{i}) = \arg\min_{\boldsymbol{a}\in\mathbb{R}^N} \sum_{k=1}^{n_s} \sum_{j=1}^{N} (\Phi_{F,i(k)j} a_j - F_{i(k)}^{\alpha})^2,\ \alpha = 1, N_{\text{snapshots}}$
        $e_{\text{temp}}(\boldsymbol{i}) = \sum_{\alpha=1}^{N\text{snapshots}} ||\boldsymbol{F}_{\Phi}^{\alpha,\text{aprox}}(\boldsymbol{i}) - \boldsymbol{F}_{\Phi}^{\alpha}||$
        if $e_{\text{temp}} < e$ then
          $e = e_{\text{temp}}$
          Permanently replace sampling node $i(m)$ by its $l$th neighbour
          Restart $l$ loop
        end if
      end if
    end for
  end for
end while
```

where

$$\boldsymbol{F}_{\Phi}^{\alpha,\text{aprox}}(\boldsymbol{i}) = \arg\min_{\boldsymbol{a}\in\mathbb{R}^N} \sum_{k=1}^{n_s} \sum_{j=1}^{N} (\Phi_{F,i(k)j} a_j - F_{i(k)}^{\alpha})^2, \alpha = 1, N_{\text{snapshots}} \tag{13}$$

For this definition of the error, we can define the optimal set of sampling components as:

$$\boldsymbol{b} = \arg \min_{\boldsymbol{i}\in\mathbb{N}^{n_s} \mid 1\leq i(k)\leq M,\ k=1,\ldots n_s} e(\boldsymbol{i}) \tag{14}$$

where $e(\boldsymbol{i})$ is given in (12).

In order to obtain the set of sampling points to be used in the reduced order simulation we proceed as follows: for each sampling node of the finite element mesh, we loop over its neighbours in the computational mesh and we temporarily replace the sampling node by each of them. If the error of the new set is lower than the original error, the sampling node is permanently replaced by its neighbour. This procedure is repeated while the set of sampling points changes due to the algorithm (**while** loop in Algorithm 1).

3.2 Numerical Example. Two-dimensional Low Reynolds Flow Past a NACA Airfoil

In this section we simulate the incompressible flow around a NACA 0012 airfoil profile [24]. The computational domain is a 32×16 rectangle, with the trailing edge of the 8 unit long airfoil placed at $(16, 8)$. The horizontal inflow velocity is set to 1 at $x = 0$, and slip boundary conditions are applied at the upper and lower walls of the computational domain. Velocity is prescribed to 0 at the airfoil surface.

The viscosity has been set to $\nu = 0.001$, which yields a Reynolds number $Re = 1000$ based on the height of the airfoil. The time step has been set to $\delta t = 0.2$. In this numerical example, the CFL number associated to the finite element mesh was $CFL \sim 62$. A 29945 linear element mesh has been used. The mesh is refined around the airfoil surface in order to be able to better capture the solution in the region surrounding the boundary layer. The angle of attack has been set to $\alpha = 0.2$, and a second order backward differences scheme has been used for the time integration.

100 velocity-pressure snapshots have been taken and the 10 first reduced basis functions have been kept for the reduced-order model. For the hyper-reduced order model, 100 additional snapshots for the right-hand-side have been taken and the corresponding 12 first reduced basis functions have been kept. The number of sampling nodes is 36.

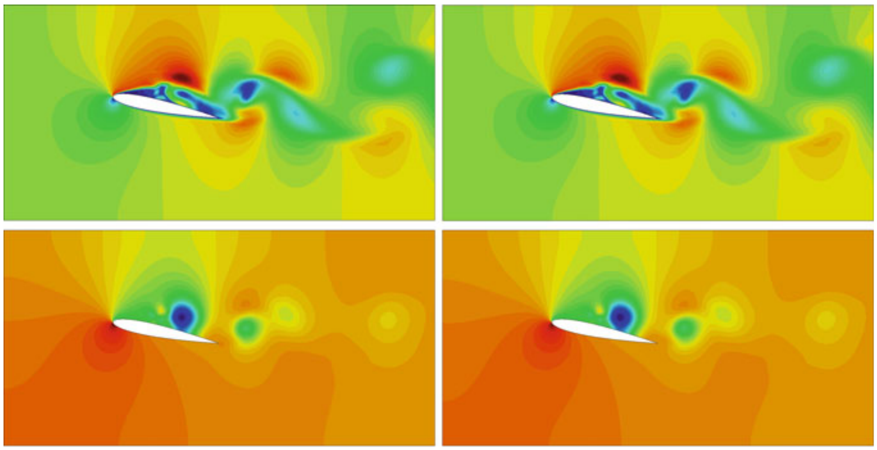


Fig. 3 Velocity (*top*) and pressure (*bottom*) contours at $Re = 1000$, $\alpha = 0.2$ after 200 time steps. Full-order (*left*) and Hyper-Reduced Order Model (*right*)

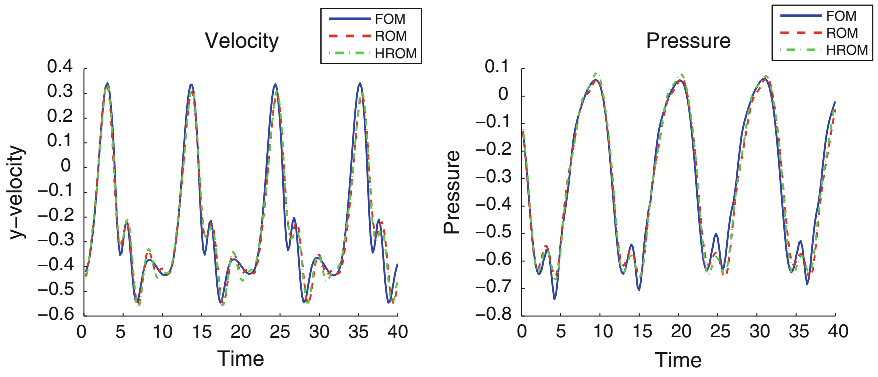


Fig. 4 Velocity (*left*) and pressure (*right*) time history at a control point at the wake of the airfoil, $Re = 1000$, $\alpha = 0.2$, second order time integration

Figure 3 compares the velocity and pressure fields after 200 time steps for the full-order and the hyper-reduced model. The reduced-order model almost exactly matches the results from the full order model.

Regarding the computational cost, the full order model takes 148.9 s to run, the reduced-order model takes 49.6 s (33 %). Finally, reduced-order model 2, in which the computational cost depends only on the size of the reduced-order model, takes only 0.71 s (0.45 %) to run.

Figures 4 and 5 show the time history and spectra for the velocity and pressure at (8, 0.5). Despite the complex flow and the high number of oscillation modes present in the solution, the reduced-order models manage to correctly capture the main modes amplitudes and frequencies.

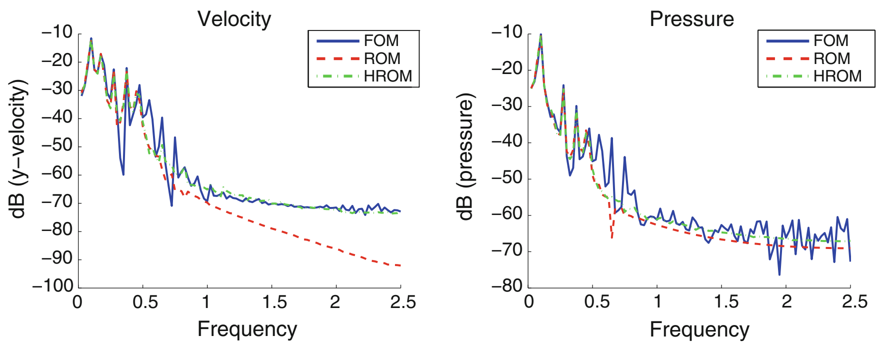


Fig. 5 Velocity (*left*) and pressure (*right*) spectra at a control point at the wake of airfoil, $Re = 1000$, $\alpha = 0.2$, second order time integration

4 A Domain Decomposition Approach for POD Reduced-Order Models

Despite the important reduction in computational cost provided by reduced-order models, one of their major drawbacks is the lack of robustness with respect to changes in the parameters which characterize the numerical simulation. This lack of robustness causes the reduced model to be valid only in a small parameter region close to the parameter values for which the reduced model was built [3], requiring the snapshot collection and the reduced model to be updated when, for instance, an optimization process leads to a parameter configuration which becomes too separated from the starting parameter set.

In this section we present a strategy which allows to improve the behavior of non-linear reduced models (where hyper-reduction is used for the reconstruction of the reduced-order equations) in parameter configurations which are not present in the snapshot set from which the reduced model is built [6]. It is based on introducing a domain decomposition approach to the model reduction, partitioning the computational domain into several regions, each of which is dealt with localized POD bases. This gives us the possibility of treating each of the subdomains with a different degree of approximation, or even, as we will see, solving the full-order equations in some of the subdomains.

4.1 Domain Decomposition POD Model

Let us consider the splitting of the computational domain Ω into two subdomains Ω_k, $k = 1, 2$, and the associated local unknowns $\boldsymbol{U}^k \in \mathbb{R}^{M_k}$, $M = M_1 + M_2$. If the domain decomposition is applied to the equations arising from a finite element problem, the partition into subdomains is done by assigning each of the nodes (and nodal unknowns) of the finite element mesh to a subdomain. This means that there are no interface nodes, instead we define interface elements as those elements who own nodes from different subdomains. Let us define a local reduced order basis ϕ^k consisting of the reduced basis functions $\phi_i^k \in \mathbb{R}^{M_k}$, $i = 1, ..., N_k$, to approximate $\boldsymbol{U}^k$ in each subdomain. Note that the number of basis functions in each subdomain is not necessarily the same, although we have considered it to be equal from now on for simplicity. The possible ways to construct this basis are discussed later. This local basis can be extended to the global domain by defining $\boldsymbol{\Phi}_i^k \in \mathbb{R}^M$:

$$\boldsymbol{\Phi}_i^1 := \begin{bmatrix} \phi_i^1 \\ \mathbf{0} \end{bmatrix}, \quad \boldsymbol{\Phi}_i^2 := \begin{bmatrix} \mathbf{0} \\ \phi_i^2 \end{bmatrix}, \tag{15}$$

where the null terms correspond to components of the global system which lie outside Ω_k. Taking this into account, the unknown $\boldsymbol{U}$ is approximated as:

$$\boldsymbol{U} \approx \sum_{i=1}^{N} (\boldsymbol{\Phi}_i^1 \alpha_i^1 + \boldsymbol{\Phi}_i^2 \alpha_i^2) = (\boldsymbol{\Phi}^1 \boldsymbol{\alpha}^1 + \boldsymbol{\Phi}^2 \boldsymbol{\alpha}^2), \quad \boldsymbol{\Phi}^k \in \mathbb{R}^{M\times N}, \ \boldsymbol{\alpha}^k \in \mathbb{R}^N \ k = 1, 2, \tag{16}$$

where $\boldsymbol{\alpha}^k$ are the solution coefficients for subdomain k.

Let $\boldsymbol{A} \in \mathbb{R}^{M\times M}$ be the matrix of the system whose solution is $\boldsymbol{U} \in \mathbb{R}^M$, and $\boldsymbol{F} \in \mathbb{R}^M$ the RHS vector. They can be partitioned into the components associated to each subdomain Ω_k, $k = 1, 2$, so that

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}|_{11} & \boldsymbol{A}|_{12} \\ \boldsymbol{A}|_{21} & \boldsymbol{A}|_{22} \end{bmatrix}, \quad \boldsymbol{A}|_{kl} \in \mathbb{R}^{M_k \times M_l}, \qquad \boldsymbol{F} = \begin{bmatrix} \boldsymbol{F}|_1 \\ \boldsymbol{F}|_2 \end{bmatrix}, \quad \boldsymbol{F}|_k \in \mathbb{R}^{M_k}.$$

The monolithic approach for the domain decomposition ROM is obtained by introducing the union of the extensions of the local bases to the global domain as the global reduced order basis:

$$\begin{aligned} (\boldsymbol{\Phi}^1)^T \boldsymbol{A}(\boldsymbol{\Phi}^1 \boldsymbol{\alpha}^1 + \boldsymbol{\Phi}^2 \boldsymbol{\alpha}^2) &= (\boldsymbol{\Phi}^1)^T \boldsymbol{F} \\ (\boldsymbol{\Phi}^2)^T \boldsymbol{A}(\boldsymbol{\Phi}^1 \boldsymbol{\alpha}^1 + \boldsymbol{\Phi}^2 \boldsymbol{\alpha}^2) &= (\boldsymbol{\Phi}^2)^T \boldsymbol{F}. \end{aligned} \tag{17}$$

Defining $\boldsymbol{a}_{kl} = (\boldsymbol{\Phi}^k)^T \boldsymbol{A} \boldsymbol{\Phi}^l \in \mathbb{R}^{N\times N}$ and $\boldsymbol{f}_k = (\boldsymbol{\Phi}^k)^T \boldsymbol{F} \in \mathbb{R}^N$, we may write this system as

$$\boldsymbol{a}_{11}\boldsymbol{\alpha}^1 + \boldsymbol{a}_{12}\boldsymbol{\alpha}^2 = \boldsymbol{f}_1, \tag{18}$$

$$\boldsymbol{a}_{21}\boldsymbol{\alpha}^1 + \boldsymbol{a}_{22}\boldsymbol{\alpha}^2 = \boldsymbol{f}_2. \tag{19}$$

If we also consider the decomposition of $\boldsymbol{A}$ and $\boldsymbol{F}$ the final reduced order system can be written in terms of the local bases ϕ^k:

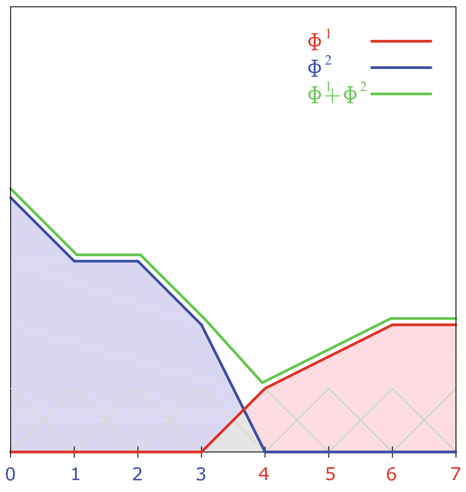


Fig. 6 Local basis functions for the domain decomposition approach. The *green* function is the sum of the local basis function of the *left* subdomain (*blue*) and the local basis function of the *right* subdomain (*red*)

$$(\phi^1)^T(\boldsymbol{A}|_{11}\phi^1\boldsymbol{\alpha}^1 + \boldsymbol{A}|_{12}\phi^2\boldsymbol{\alpha}^2) = (\phi^1)^T\boldsymbol{F}|_1$$
$$(\phi^2)^T(\boldsymbol{A}|_{21}\phi^1\boldsymbol{\alpha}^1 + \boldsymbol{A}|_{22}\phi^2\boldsymbol{\alpha}^2) = (\phi^2)^T\boldsymbol{F}|_2.$$

The off diagonal block matrices correspond to the coupling terms and are null except for the contribution of the unknowns ubicated at the domain interfaces. It can be observed that the cost of computing the ROM system is not larger than in the monolithic approach. However, the size of the reduced system is larger (dimension $2N$).

An important point is that each algebraic local basis function $\boldsymbol{\Phi}_i^k$ arises from a function defined in space. This spatial function is a linear combination of the finite element shape functions of the nodes of subdomain Ω_k. As a consequence, each of the components in $\boldsymbol{\Phi}_i^k$ corresponds to a nodal value of the spatial field to be represented on the finite element mesh. This is illustrated in Fig. 6, where examples of local basis functions for a one-dimensional problem and linear finite elements are shown. Let us also emphasize that, if the original finite element shape functions are continuous, any local (and global) basis function will also be continuous, as a consequence of the definition of the extension of the basis functions to the global domain (15). This will also hold for the combination of local basis functions, even if these belong to different subdomains. In Fig. 6 the blue basis function belongs to the left subdomain, the red basis function belongs to the right subdomain. The green line represents the addition of the blue and the red basis functions. Since both of the original functions are continuous, the green function is also continuous. Note also that there is an overlapping region where both the left and right basis functions are non-zero.

4.2 Local POD (L-POD)

The strategy for building the local POD basis consists in performing a POD for the part of the snapshots corresponding to each of the subdomains. The snapshots are first partitioned according to the domain decomposition strategy and the local basis ϕ^k is obtained from these partitioned snapshots. The global basis is again defined as $\mathbf{\Phi} = \left[\mathbf{\Phi}^1, \mathbf{\Phi}^2\right]$. Note that the number of local basis functions in each subdomain does not necessarily coincide, $N_1 \neq N_2$, $N = N_1 + N_2$.

The main features of these domain-decomposition local POD bases are the following:

- Each local basis can be ensured to be orthonormal at the algebraic level. By construction, each of the basis functions which conform the local POD has unitary norm and is orthogonal to all the basis functions in its subdomain at the algebraic level. Moreover, due to the domain decomposition approach, the projection of a local basis of a given subdomain onto the space conformed by the basis functions of any other subdomain is also zero. This ensures that if we consider the POD decomposition globally, the union of the local bases is also an orthonormal basis.
- The computation of the singular value decomposition of the local snapshots for each subdomain requires less memory than the computation of the singular value decomposition of the global snapshots.

In the case we are using hyper reduced models which require additional POD bases for reconstructing the system matrix and right-hand side, we can proceed in the same way.

Once the localized reduced order bases have been defined, the monolithic domain decomposition reduced order model is obtained by using as reduced basis the union of the local reduced bases. The fact that the basis functions are local makes the computational cost diminish with respect to the global approach with the same number of basis functions, because the operations can be done at the local level. However, the number of functions is usually larger in the domain decomposition approach, because a sufficient number of components needs to be assigned to the reduced basis of each subdomain in order to properly represent the solution in that subdomain.

4.3 Stabilization Through Overlapping and Penalty Terms

The previous domain decomposition strategy for reduced-order models, despite its simplicity, suffers from unstable behavior when it is used in a straightforward manner in the explicit reduced-order model for the stabilized finite element approximation of the incompressible Navier-Stokes equations described in the previous sections. These instabilites can be easily explained taking into account that an explicit time marching scheme is equivalent in this case to an explicit iteration-by-subdomain strategy, which is known to have convergence and stability issues. This is the reason why we propose a domain interface stabilization term, which is obtained by allowing some overlapping between subdomains and enforcing the equality of the unknown values at this overlapping region.

As in classical iteration by subdomain strategies, the overlapping region $\Omega_\cap$ is the part of Ω which belongs to both Ω_1 and Ω_2. In our approach, in which the partitioning is obtained by assigning the nodes of the finite element mesh to Ω_1 and Ω_2, overlapping is achieved by allowing some nodes close to the interface to belong to both Ω_1 and Ω_2. The local reduced bases are computed by performing the POD of the restriction of the snapshots to Ω_k, but the obtained basis functions need to be corrected. Suppose that the original overlapping local POD bases $\mathbf{\Phi}^{01} \in \mathbb{R}^{M\times N_1}$ and $\mathbf{\Phi}^{02} \in \mathbb{R}^{M\times N_2}$ are:

$$\mathbf{\Phi}^{01} = \begin{bmatrix} \phi^1 \\ \phi^{1\cap} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{\Phi}^{02} = \begin{bmatrix} \mathbf{0} \\ \phi^{2\cap} \\ \phi^2 \end{bmatrix}, \tag{20}$$

where now

$$\phi^k \in \mathbb{R}^{M_k\times N_k}, \tag{21}$$

is the restriction of the local basis functions in Ω_k to the part of the subdomain without overlapping (M_k components), and

$$\phi^{k\cap} \in \mathbb{R}^{M_{k\cap}\times N_k}, \tag{22}$$

corresponds to the restriction of $\mathbf{\Phi}^{0k}$ to the overlapping domain $\Omega_\cap$ ($M_{k\cap}$ components). Note that $M_\cap := M_{1\cap} = M_{2\cap}$, and now $M = M_1 + M_2 + M_\cap$.

In this case the corrected bases are:

$$\mathbf{\Phi}^{1} = \begin{bmatrix} \phi^1 \\ \beta\phi^{1\cap} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{\Phi}^{2} = \begin{bmatrix} \mathbf{0} \\ (1-\beta)\phi^{2\cap} \\ \phi^2 \end{bmatrix}, \tag{23}$$

where $\beta \in [0, 1]$ is a weighting parameter. Note that the limits $\beta = 0$ and $\beta = 1$ correspond to the non overlapping case. If several subdomains overlap in a certain region, then each subdomain is assigned a weighting parameter β^k and we must ensure that $\sum \beta^k = 1$. The motivation for this correction is the requirement that the resulting global reduced basis (obtained as the union of the local bases for each subdomain) is capable of representing the global snapshot set if N is equal to the number of snapshots. This is shown in Fig. 7, where some illustrative basis functions for a one-dimensional problem are depicted.

Supposing that the problem defined in (1) allows us to do so, the stabilization penalty term imposes that the solution at the overlapping region recovered from $\phi^{1\cap}$ (prior to the introduction of the weighting parameter β) is equal to the solution recovered from $\phi^{2\cap}$:

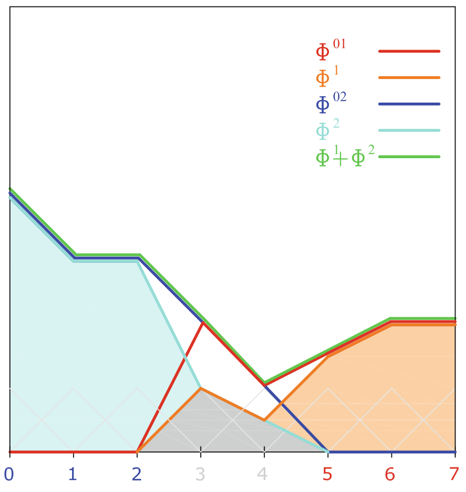


Fig. 7 Overlapping local basis functions. $\beta = 0.5$. The overlapping nodes are depicted in *gray*. In this particular case the value of the basis functions at the overlapping nodes coincides, which is not necessarily the case for L-POD

$$\phi^{1\cap}\boldsymbol{\alpha}^1 = \boldsymbol{U}^{1\cap} = \boldsymbol{U}^{2\cap} = \phi^{2\cap}\boldsymbol{\alpha}^2 \in \mathbb{R}^{M\cap}, \tag{24}$$

where now we take $\boldsymbol{\alpha}^k \in \mathbb{R}^{N_k}$ the ROM degrees of freedom for each subdomain.

This condition can be equivalently written as:

$$\begin{aligned}(\boldsymbol{\Phi}^{1\cap})^T\boldsymbol{\Phi}^{1\cap}\boldsymbol{\alpha}^1 - (\boldsymbol{\Phi}^{1\cap})^T\boldsymbol{\Phi}^{2\cap}\boldsymbol{\alpha}^2 = \mathbf{0},\\ (\boldsymbol{\Phi}^{2\cap})^T\boldsymbol{\Phi}^{1\cap}\boldsymbol{\alpha}^1 - (\boldsymbol{\Phi}^{2\cap})^T\boldsymbol{\Phi}^{2\cap}\boldsymbol{\alpha}^2 = \mathbf{0},\end{aligned} \tag{25}$$

where

$$\boldsymbol{\Phi}^{k\cap} = \begin{bmatrix} \mathbf{0} \\ \phi^{k\cap} \\ \mathbf{0} \end{bmatrix}. \tag{26}$$

Introducing (25) as a penalized constraint in the ROM system we get:

$$\boldsymbol{a}_{11}\boldsymbol{\alpha}^1 + \boldsymbol{a}_{12}\boldsymbol{\alpha}^2 + \frac{1}{\varepsilon}(\boldsymbol{M}_{11}\boldsymbol{\alpha}^1 - \boldsymbol{M}_{12}\boldsymbol{\alpha}^2) = \boldsymbol{f}_1, \tag{27}$$

$$\boldsymbol{a}_{21}\boldsymbol{\alpha}^1 + \boldsymbol{a}_{22}\boldsymbol{\alpha}^2 + \frac{1}{\varepsilon}(\boldsymbol{M}_{21}\boldsymbol{\alpha}^1 - \boldsymbol{M}_{22}\boldsymbol{\alpha}^2) = \boldsymbol{f}_2, \tag{28}$$

where

$$\boldsymbol{M}_{kl} = (\boldsymbol{\Phi}^{k\cap})^T\boldsymbol{\Phi}^{l\cap} \in \mathbb{R}^{N_k\times N_l}. \tag{29}$$

and the definition of $\boldsymbol{\Phi}^k$ for building the $\boldsymbol{a}$ matrices is taken as in (23).

An important property of the block diagonal penalty matrices $\boldsymbol{M}_{kk}$ is that they can only be guaranteed be full-rank matrices if $\Omega_{\cap} = \Omega$. However, this stabilization strategy shows good results in the numerical examples even if $\Omega_{\cap} \neq \Omega$. The introduction of the $\boldsymbol{M}$ matrices to the reduced order formulation allows one to obtain a stable solution in the practical cases. The stabilization parameter ε is chosen so that, on the one hand, the penalty terms are sufficiently large to provide the desired stabilization effects, and on the other, the norm of $\frac{1}{\varepsilon}\boldsymbol{M}$ is proportional to the norm of $\boldsymbol{A}$. In this way we ensure that the resulting system does not become ill-conditioned.

4.4 Full-Order / Reduced-Order Domain Decomposition (FOM-ROM)

Another possibility is the use of a hybrid Full-Order / Reduced-Order (FOM-ROM) approach. This is convenient if a high fidelity model is required in a certain region of the domain, or if the conditions in a certain region strongly depart from the conditions at which the snapshots for building the POD bases were taken. In this cases one can choose to solve the FOM problem in one of the subdomains, while keeping the cheaper ROM approach in the less critical subdomains. Extending the described partitioned ROM strategy to a hybrid FOM-ROM domain decomposition method is straightforward: the FOM-ROM is obtained by taking as local basis for the FOM subdomain Ω_F the nodal shape functions of the finite element space for the unknown. In the ROM subdomain Ω_R a local reduced basis needs to be built. The hybrid FOM-ROM system without overlapping is:

$$\begin{bmatrix} \boldsymbol{A}|_{FF} & \boldsymbol{A}|_{FR}\phi^R \\ (\phi^R)^T\boldsymbol{A}|_{RF} & (\phi^R)^T\boldsymbol{A}|_{RR}\phi^R \end{bmatrix} \begin{bmatrix} \boldsymbol{U}^F \\ \boldsymbol{\alpha}^R \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}|_F \\ (\phi^R)^T\boldsymbol{F}|_R \end{bmatrix}. \tag{30}$$

Let us remark that the time stepping strategies need not to be the same for the full order and the reduced order equations. For instance, if the explicit reduced order model described in the previous sections is used for the incompressible Navier-Stokes equations, the $\boldsymbol{A}$ matrix and the $\boldsymbol{F}$ RHS vector for the reduced order equations are taken from the explicit model, while the equations arising from the implicit time stepping are kept for the full order equations:

$$\begin{bmatrix} \boldsymbol{A}|_{FF} & \boldsymbol{A}|_{FR}\phi^R \\ (\phi^R)^T\boldsymbol{A}^{\exp}|_{RF} & (\phi^R)^T\boldsymbol{A}^{\exp}|_{RR}\phi^R \end{bmatrix} \begin{bmatrix} \boldsymbol{U}^F \\ \boldsymbol{\alpha}^R \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}|_F \\ (\phi^R)^T\boldsymbol{F}^{\exp}|_R \end{bmatrix}. \tag{31}$$

If a Petrov-Galerkin projection is used, this can also be introduced in the ROM equations. For instance, the FOM-ROM system for the Petrov-Galerkin projection described in [11, 13] would result in the following system:

$$\begin{bmatrix} \boldsymbol{A}|_{FF} & \boldsymbol{A}|_{FR}\phi^R \\ \boldsymbol{A}_{RF}^{PG} & \boldsymbol{A}_{RR}^{PG}\phi^R \end{bmatrix} \begin{bmatrix} \boldsymbol{U}^F \\ \boldsymbol{\alpha}^R \end{bmatrix} = \begin{bmatrix} \boldsymbol{F}|_F \\ \boldsymbol{F}_R^{PG} \end{bmatrix}, \tag{32}$$

where

$$\begin{aligned} \boldsymbol{A}_{RF}^{PG} &= (\phi^R)^T \left(\boldsymbol{A}|_{FR}^T \boldsymbol{A}|_{FF} + \boldsymbol{A}|_{RR}^T \boldsymbol{A}|_{RF} \right), \\ \boldsymbol{A}_{RR}^{PG} &= (\phi^R)^T \left(\boldsymbol{A}|_{FR}^T \boldsymbol{A}|_{FR} + \boldsymbol{A}|_{RR}^T \boldsymbol{A}|_{RR} \right), \\ \boldsymbol{F}_R^{PG} &= (\phi^R)^T \boldsymbol{A}|_{FR}^T \boldsymbol{F}|_F + (\phi^R)^T \boldsymbol{A}|_{RR}^T \boldsymbol{F}|_R. \end{aligned}$$

Also, any hyper-reduction technique for efficiently reconstructing the ROM equations can be used. The described overlapping strategies and the use weighting coefficients need to be introduced in the previous formulation. This can be done in a straightforward manner, including the use of different weighting parameters β or ε for the FOM and the ROM equations.

4.5 Particularities of the Application to the Incompressible Navier-Stokes Equations

The use of the domain decomposition ROM strategy to the particular problem of the incompressible Navier-Stokes equations is straightforward if a ROM approach is used in all the subdomains. On the other hand, some care needs to be taken when a FOM approximation is used in one of the subdomains while a ROM approximation is used in its neighbour subdomains. As in the original domain decomposition strategy, a penalization term through overlapping is convenient in this FOM-ROM approach. However, it is necessary to distinguish between the velocity and the pressure unknowns of the incompressible Navier-Stokes equations in this case: only the equality between the FOM and the ROM velocities in the overlapping region is imposed, and no condition is required on the FOM pressure field. This is so because the pressure field can be understood as the Lagrange multiplier enforcing the incompressibility constraint, and as such it is not possible to enforce the pressure value over the overlapping domain.

4.6 Numerical Example. Flow Injection in a Rectangular Cylinder

In this numerical example we show the capability of the proposed FOM-ROM strategy to adapt to flow configurations which were not present in the original snapshot set. The initial problem set is the incompressible flow past a rectangular cylinder at $Re = 100$. The computational domain consists of a 24×12 rectangle with a square cylinder with a side of size 1. The square cylinder is centered at coordinates $(8, 6)$. The horizontal inflow velocity is set to 1. Slip boundary conditions which allow the

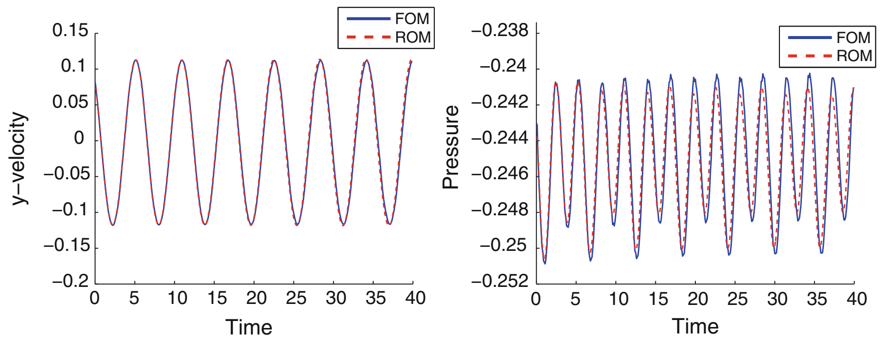


Fig. 8 Comparison of the FOM and ROM velocities at (5,4) for the initial configuration

flow to move in the direction parallel to the walls are set at $y = 0$ and $y = 12$, and velocity is set to $\mathbf{0}$ on the cylinder surface in the direction normal to the surface. A tangential force (computed by using a wall-law approach) is used to model the velocity in the tangential direction. The viscosity has been set to $\nu = 0.01$, which yields a Reynolds number $Re = 100$ based on the dimension of the cylinder and the inflow velocity. A second order backward difference scheme has been used for the time integration with time step $\delta t = 0.1$. In this example, a relatively fine 67224 linear element mesh has been used to solve the problem.

An initial run of the full-order model is performed for the snapshot collection and no domain decomposition strategy is applied in the initial run. The FOM model takes 849.36 s to run.After the snapshot collection procedure, the ROM is capable of reproducing the FOM solution with a good accuracy for the velocity field (2.1 % of relative error in the L^2-norm for the last oscillation period) , the pressure amplitude being underpredicted (but only with 0.8 % of relative error in the last oscillation period), and a very low computational cost (3.07 s, 0.37 % of the original computational cost), as illustrated in Fig. 8. For the ROM run, 10 basis functions are used, which are obtained from the POD decomposition of the original 50 snapshot collection.

As illustrated in Fig. 8, the reduced-order model is capable of reproducing the solution of the full-order model for the configuration in which the snapshots were taken. However, let us now consider the flow injection in the downstream side of the cylinder illustrated in Fig. 9, which is introduced in order to modify the flow. The velocity in the injection region (whose length is 0.2) is 0.1 in the direction normal to the cylinder surface. Figure 10 illustrates the behavior of the reduced order model when the injection is considered. Despite its very low computational cost compared to the FOM model, it is clear that the ROM is incapable of reproducing the new flow configuration; the reason for this is that the snapshot set from which the ROM basis was built does not contain the solution with the flow injection.

Let us now consider the FOM-ROM strategy described in the previous sections. We will decompose the physical domain into two subdomains, based on our a priori knowledge of the boundary conditions of the problem: the first subdomain corre-

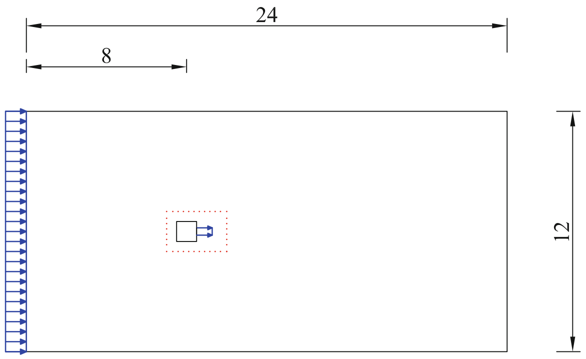


Fig. 9 Flow injection configuration. The *red dotted line* denotes the FOM domain for the FOM-ROM model

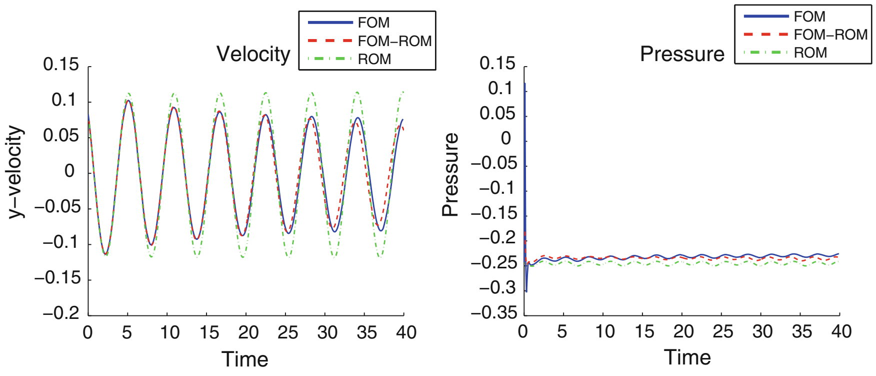


Fig. 10 Comparison of the vertical velocity (*left*) and pressure (*right*) at (5,4) for the FOM, FOM-ROM and ROM models for the injection case

sponds to the region surrounding the square cylinder of the rectangle $(7, 10) \times (5, 7)$. In this subdomain a FOM approach is going to be taken, and the Navier-Stokes equations are going to be solved with full accuracy. The second subdomain covers the rest of the computational domain. Since this region does not involve the critical area where the vortexes are formed, it is going to be solved by means of the less accurate ROM strategy. The ROM basis are obtained from a set of 100 snapshots, from which a L-POD basis of 10 basis functions is obtained. As it will be shown, the combination of both strategies (FOM and ROM) allows us to recover a solution which is close to the full FOM solution, but at a much lower computational cost.

Figure 10 shows a comparison of the vertical velocity and pressure at a point at the wake of the cylinder with coordinates $(5, 4)$, for the FOM, the ROM and the FOM-ROM models. It is interesting to note that the ROM model is not able to capture the physics of the problem; this is natural since the ROM basis does not contain the solution of the injection case. The FOM-ROM model, on the other hand,

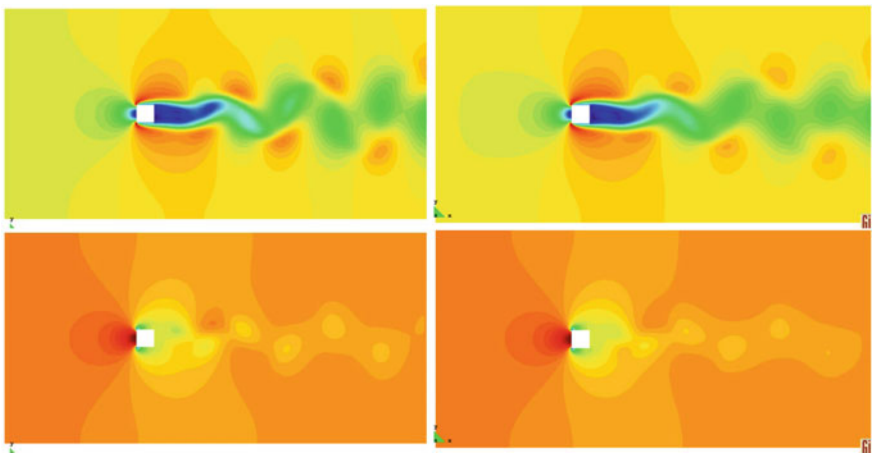


Fig. 11 Comparison of the velocity (*top*) and pressure (*bottom*) fields after 400 steps. *Left* FOM. *Right* FOM-ROM

is capable of a quite accurate solution of the system evolution in the short term in the FOM domain (13.3 % relative error for the velocity time history in the last oscillation period and 4.6 % error in the pressure). Figure 11 compares the velocity and pressure fields of the FOM and the FOM-ROM models. We can observe that in the region surrounding the cylinder (FOM region) the velocity and pressure fields are very similar, in the ROM region the velocity fields slightly differ, with more intense vortexes or bulbs in the FOM simulation. This is due to the difficulties for the ROM model for representing the injected velocity and pressure fields (the used snapshots are *bad* for the injection case). Despite this evident lack of optimality of the snapshot set, the FOM-ROM model is capable of properly representing the solution in the FOM region. Figure 12 shows a comparison between the FOM simulation and FOM-ROM model for several injection velocities. The accuracy of the FOM-ROM model decreases as the absolute value of the injection velocity increases. This is due to the fact that the larger the injection velocity, the more different the flow becomes from the original FOM simulation without injection. Regarding the computational cost, the FOM-ROM approach takes 55.56 s to run, which is only 6.7 % of the original FOM computational cost.

5 Conclusions

In this chapter we have discussed several strategies for dealing with the reduced-order approximation of the incompressible Navier-Stokes equations. We have departed from a stabilized finite element full-order approximation and we have approached the order reduction by using a Proper Orthogonal Decomposition (POD) method.

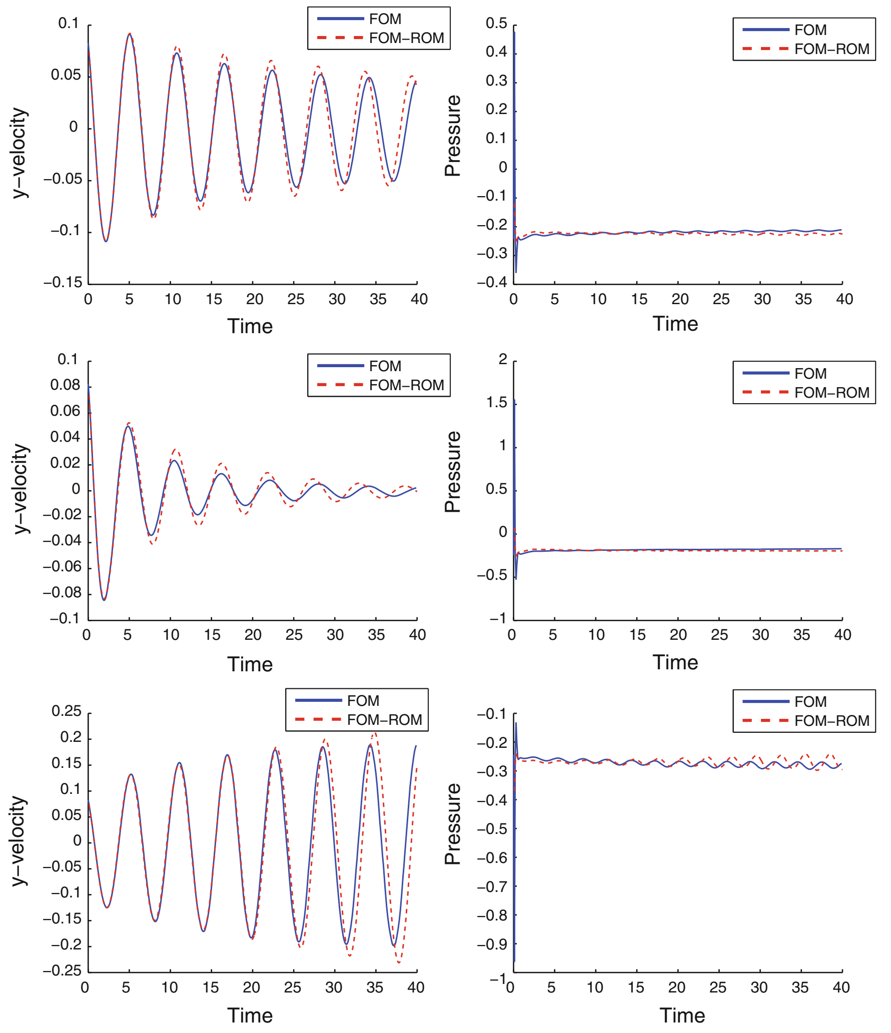


Fig. 12 Comparison of the vertical velocity (*left*) and pressure (*right*) at (5,4) for the FOM, FOM-ROM and ROM models for the injection case. Injection velocities, from *top* to *bottom* 0.2, 0.5, -0.2

In the first part of the chapter, we have focused in the construction of an explicit reduced-order model for the incompressible Navier-Stokes equations, and the application of hyper-reduction techniques to it. The basic idea is to treat all the terms except the mass matrix in the temporal derivative in an explicit way. This includes the non-linear convective term, but also the stabilization terms which can be highly non-linear through the stabilization parameter τ. In order to do so, we take advantage of the fact that the snapshots used for building the reduced-order basis through a singular value decomposition in the POD procedure do already fulfill the stabilized continuity equation. Secondly, we also acknowledge the fact that, if the velocity and pressure are treated jointly, then the pressure can be recovered from the reduced-order basis and the solution coefficients at the end of each time step.

The proposed explicit reduced-order model performs well in practical cases, as illustrated in the numerical examples section. Despite the time-stepping scheme being explicit, the Courant-Friedrichs-Levy condition can be violated, which can be explained because the reduced basis functions expand over the whole computational domain. On the other hand, the reduced model is sensitive to the inclusion of noisy basis functions, which can cause unstable solutions to appear. The sensitivity of the explicit reduced-order model to this issue can be improved by reducing the time step and refining the finite element mesh.

A hyper-reduction strategy for the explicit reduced-order model has also been presented, which is based on the reconstruction of the right-hand-side vector through a gappy-pod procedure. For the selection of the indices of the gappy reconstruction, we use a discrete version of the Best Points Interpolation Method (DBPIM), which uses only values at the nodes of the finite element mesh, with the advantage that the selected points can be guaranteed to be at least locally optimal.

In the second part of the chapter, we have presented a domain decomposition strategy for non-linear hyper-reduced-order models. The method consists of restricting the reduced-order basis functions to the nodes of each subdomain. This definition of the partitioned problem directly ensures the continuity of the recovered solution. The local POD bases are obtained by computing a local POD decomposition for the partitioned snapshots. When applied to the explicit reduced-order model for the incompressible Navier-Stokes equations a stabilizing penalization term is required. This penalty term is defined so that it weakly enforces the equality of the unknown between subdomains in an overlapping region.

The domain decomposition reduced-order model can be extended to a particular case, in which one of the subdomains is solved by using the full-order finite element equations while the other ones are solved using the reduced-order model. This diminishes the computational cost in the low-resolution subdomains, while keeping the high fidelity solution in the domain regions which are subject to more complex physical phenomena.

Numerical examples illustrate the accuracy of the proposed methods for the solution of incompressible flow problems at a low computational cost: the reduced order-model allows us to save up to 65 % of the computational cost, while in the case of the hyper-reduced order models the computational saving is larger than 99 % the original computational cost.

References

1. Akhtar I, Borggaard J, Hay A (2010) Shape sensitivity analysis in flow models using a finite-difference approach. Math Probl Eng 1–23:2010
2. Antil H, Heinkenschloss M, Hoppe RHW, Sorensen DC (2010) Domain decomposition and model reduction for the numerical solution of pde constrained optimization problems with

localized optimization variables. Comput Vis Sci 13(6):249–264
3. Arian E, Fahl M, Sachs EW (2000). Trust-Region proper orthogonal decomposition for flow control. Institute for computers, pp 2000–2101
4. Astrid P (2004). Reduction of process simulation models: a proper orthogonal decomposition approach. PhD thesis, Department of Electrical Engineering, Eindhoven University of Technology
5. Astrid P, Weiland S, Willcox K, Backx T (2008) Missing point estimation in models described by proper orthogonal decomposition. IEEE Trans Autom Control 53:2237–2251
6. Baiges J, Codina R, Idelsohn S (2013) A domain decomposition strategy for reduced order models. Application to the incompressible Navier-Stokes equations. Comput Meth Appl Mech Eng 267:23–42
7. Baiges J, Codina R, Idelsohn S (2013) Explicit Reduced Order Models for the stabilized finite element approximation of the incompressible Navier-Stokes equations. Int J Numer Meth Fluids 72:1219–1243
8. Barrault M, Maday Y, Nguyen NC, Patera AT (2004) An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. Comptes Rendus Mathematique 339(9):667–672
9. Bergmann M, Cordier L, Brancher JP (2007) Drag minimization of the cylinder wake by trust-region proper orthogonal decomposition. Notes on numerical fluid mechanics and multidisciplinary design 95:19
10. Buffoni M, Telib H, Iollo A (2009) Iterative methods for model reduction by domain decomposition. Comput Fluids 38(6):1160–1167
11. Bui-Thanh T, Willcox K, Ghattas O (2008) Model reduction for large-scale systems with high-dimensional parametric input space. SIAM J Sci Comput 30(6):3270
12. Burkardt J, Gunzburger M, Lee H (2006) POD and CVT-based reduced-order modeling of Navier-Stokes flows. Comput Meth Appl Mech Eng 196(1–3):337–355
13. Carlberg K, Bou-Mosleh C, Farhat C (2011) Efficient non-linear model reduction via a least-squares Petrov-Galerkin projection and compressive tensor approximations. Int J Numer Meth Eng 86(2):155–181
14. Chatterjee A (2000) An introduction to the proper orthogonal decomposition. Curr Sci 78(7):808–817
15. Chaturantabut S, Sorensen DC (2009). Discrete empirical interpolation for nonlinear model reduction. Technical Report TR09-05, Rice University, Houston, Texas
16. Codina R (2001) A stabilized finite element method for generalized stationary incompressible flows. Comput Meth Appl Mech Eng 190:2681–2706
17. Drohmann M, Haasdonk B, Ohlberger M (2012) Reduced basis approximation for nonlinear parameterized evolution equations based on empirical operator interpolation. SIAM J Sci Comput 34:937–962
18. Everson R, Sirovich L (1995) Karhunen-Loève procedure for gappy data. J Opt Soc Am A 12:1657–1664
19. Galletti B, Bruneau CH, Zannetti L, Iollo A (2004) Low-order modelling of laminar flow regimes past a confined square cylinder. J Fluid Mech 503:161–170
20. Glaz B, Liu L, Friedmann PP (2010) Reduced-Order nonlinear unsteady aerodynamic modeling using a surrogate-based recurrence framework. AIAA J 48(10):2418–2429
21. Graham WR, Peraire J, Tang KY (1999) Optimal control of vortex shedding using low-order models. Part i-open-loop model development. Int J Numer Meth Eng 44(7):945–972
22. Grepl MA, Maday Y, Nguyen NC, Patera AT (2007) Efficient Reduced-Basis treatment of nonaffine and nonlinear partial differential equations. ESAIM. Math Model Numer Anal 41(03):575–605
23. Holmes P, Lumley JL, Berkooz G (1998) Turbulence, coherent structures. Dynamical systems and symmetry. Cambridge University Press, New York
24. Jacobs EN, Ward KE, Pinkerton RM (1933). The characteristics of 78 related airfoil sections from tests in the variable-density wind tunnel. NACA report, 460

25. Kalashnikova I, Barone MF (2011). Stable and efficient galerkin reduced order models for non-linear fluid flow. In: AIAA-2011-3110, 6th AIAA theoretical fluid mechanics conference, Honolulu
26. Kosambi DD (1943) Statistics in function space. J Indian Math Soc 7:76–88
27. Lassila T, Rozza G (2010) Parametric free-form shape design with PDE models and reduced basis method. Comput Meth Appl Mech Eng 199(23–24):1583–1592
28. LeGresley PA (2005). Application of proper orthogonal decomposition to design decomposition methods. PhD thesis, Department of Aeronautics and Astronautics, Stanford University
29. Lucia DJ, Beran PS (2003) Projection methods for reduced order models of compressible flows. J Comput Phys 188(1):252–280
30. Lucia DJ, King PI, Beran PS (2003) Reduced order modeling of a two-dimensional flow with moving shocks. Comput Fluids 32(7):917–938
31. Nguyen NC, Peraire J (2008) An efficient reduced-order modeling approach for non-linear parametrized partial differential equations. Int J Numer Meth Eng 76(1):27–55
32. Noack BR, Morzynski M, Tadmor G (2011) Reduced-Order modelling for flow control. Springer, Berlin
33. Rabczuk T, Bordas SPA, Kerfriden P, Goury O (2012). A partitioned model order reduction approach to rationalise computational expenses in multiscale fracture mechanics
34. Rozza G, Lassila T, Manzoni A (2011) Reduced basis approximation for shape optimization in thermal flows with a parametrized polynomial geometric map. In: Hesthaven JS, Ranquist EM (eds) Spectral and high order methods for partial differential equations., vol 76Springer, Berlin, pp 307–315
35. Ryckelynck D (2005) A priori hyperreduction method: an adaptive approach. J Comput Phys 202(1):346–366
36. Ryckelynck D (2009) Hyper-reduction of mechanical models involving internal variables. Int J Numer Meth Eng 77(1):75–89
37. Verhoeven A, Voss T, Astrid P, ter Maten EJW, Bechtold T (2007) Model order reduction for nonlinear problems in circuit simulation. PAMM 7(1):1021603–1021604
38. Verhoeven A, Maten J, Striebel M, Mattheij R (2009) Model order reduction for nonlinear ic models. In: Korytowski A, Malanowski K, Mitkowski W, Szymkat M (eds) System modeling and optimization, vol 312, IFIP advances in information and communication technology, Springer, Berlin, pp 476–491
39. Veroy K, Patera AT (2005). Certified real-time solution of the parametrized steady incompressible Navier-Stokes equations: rigorous reduced-basis a posteriori error bounds. Int J Numer Meth Fluids, 47(8–9):773–788
40. Wang Z, Akhtar I, Borggaard J, Iliescu T (2011). Proper orthogonal decomposition closure models for turbulent flows: a numerical comparison. arXiv:1106.3585
41. Wicke M, Stanton M, Treuille A. Modular bases for fluid dynamics. ACM Trans Graph, 28(3):39:1–39:8

A Survey of Hierarchical Model (Hi-Mod) Reduction Methods for Elliptic Problems

Simona Perotto

Abstract In this work, we review the basic aspects of the so called *Hierarchical Model* (*Hi-Mod*) reduction approach, recently advocated to reduce the complexity of models for advection-diffusion-reaction phenomena in pipe-like domains featuring a prevalent axial dynamics. The Hi-Mod approach aims at reducing the computational costs still preserving a reliable approximation of the transverse components of the solution by properly combining finite elements and modal approximations. In particular, we consider the convergence of this approximation to the solution of the full problem and the different ways for selecting the number of transverse modes.

1 Why Hi-Mod Reduction?

Mathematical and numerical models are nowadays a fundamental tool for quantitative analysis in many fields of science and engineering. On the one hand, sophisticated models can be reliably used for complex dynamics (fluid-structure interaction, biochemical reactions, etc.) not only for computing quantities of interest, but also for solving optimization, identification or, more in general, inverse problems. On the other hand, practical use of these tools demands a significant reduction of computational costs. This may be extremely challenging in particular for inverse problems. For this reason, one important recent research line is devoted to the set up of surrogate models and solutions for a particular problem, towards the construction of the best trade-off between reliability and computational efficiency [19]. This can be achieved with a reduction of the size of the (finite dimensional) solution, based on the on-line/off-line paradigm like in the Proper Orthogonal Decomposition approach or in the Reduced Basis method. A different—somehow complementary—approach is

S. Perotto (✉)
MOX, Dipartimento di Matematica "F. Brioschi", Politecnico di Milano,
Piazza Leonardo da Vinci 32, I-20133 Milano, Italy
e-mail: simona.perotto@polimi.it

based on the simplification of the model to be solved, by taking advantage of specific features of the problem. For instance, fluid dynamics in networks of pipes, as in the circulatory system or in internal combustion engines, can be based on the well-known Euler equations that describe the main axial dynamics, dropping the transverse components. Although this is an excellent approach for predicting the dynamics over an entire network, it may fail in describing important local details, where transverse components are important. For this reason, coupling of Euler (in 1D) and incompressible Navier-Stokes equations in (3D) has been addressed in [12], where the so-called *geometrical multiscale approach* has been introduced. A different way for including both axial and transverse components of the dynamics in a dimensionally homogeneous framework has been introduced in [22] for advection-diffusion-reaction problems. According to this approach, the main dynamics, numerically solved with a finite element approximation, is added by transverse components described by a modal or spectral representation. A small number of modes is required when the transverse dynamics is not important, leading to a psychologically 1D model. As a matter of fact, the discrete problem obtained in this way is a coupled system of 1D block problems, with interesting algebraic properties. In addition, the number m of modes can be selected to improve locally the reliability of the model, including, when needed, a more precise description of the transverse components in a hierarchical fashion. In this work, we review the basic aspects of this model reduction technique, called *Hierarchical Model* (*Hi-Mod*) reduction. In particular, we consider the convergence of this approximation to the solution of the full problem and three different ways for selecting the number of transverse modes. In particular, m may be selected uniformly, piecewise constant by subdomains or at each node of the axial finite element discretization. In addition, it can be tuned based on *a priori* as well as *a posteriori* considerations. In the latter case, we naturally obtain a model-adaptive tool selecting automatically the accuracy for the transverse dynamics.

2 The Computational Domain

Problems relevant to the Hi-Mod formulation feature a domain where one direction is prevalent. Thus, we assume that $\Omega \subset \mathbf{IR}^d$ coincides with a d-dimensional fiber bundle, with $d = 2, 3$, so that $\Omega = \bigcup_{x\in\Omega_{1D}}\{x\} \times \gamma_x$, where Ω_{1D} is the *supporting 1D domain* described by only one independent variable x, while $\gamma_x \subset \mathbf{IR}^{d-1}$ denotes the *transverse fiber* which, in general, is a function of x. Thus, we align Ω_{1D} with the dominant dynamics exhibited by the problem at hand and the fibers γ_x with the secondary transverse dynamics. For the sake of simplicity, we choose Ω_{1D} as the interval $]x_0, x_1[$. The more general case of a curved supporting fiber can be considered as well (see Remark 2). Now, we partition the boundary $\partial\Omega$ of Ω into three disjoint sets, $\Gamma_0 = \{x_0\} \times \gamma_{x_0}$, $\Gamma_1 = \{x_1\} \times \gamma_{x_1}$ and $\Gamma_* = \bigcup_{x\in\Omega_{1D}} \partial\gamma_x$, such that $\partial\Omega = \Gamma_0 \cup \Gamma_1 \cup \Gamma^*$. We assume that either homogeneous Dirichlet or homogeneous Neumann boundary conditions can be enforced on Γ_0, Γ_1 and Γ_*, as well as non-homogeneous Dirichlet data can be assigned on Γ_0 and Γ_1.

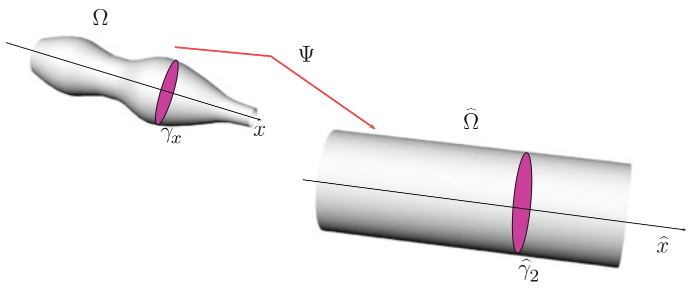


Fig. 1 Example of the map Ψ for a rectilinear three-dimensional domain

For any $x \in \Omega_{1D}$, we introduce the map $\psi_x : \gamma_x \to \widehat{\gamma}_{d-1}$ between the generic fiber γ_x and a reference fiber $\widehat{\gamma}_{d-1}$ of the same dimension. Maps ψ_x induce in turn the general map $\Psi : \Omega \to \widehat{\Omega}$ between the physical domain Ω and the reference domain $\widehat{\Omega} = \bigcup_{x\in\Omega_{1D}}\{x\} \times \widehat{\gamma}_{d-1}$, where the computations are carried out once and for all. A generic point in Ω $(\widehat{\Omega})$ is referred to as $\mathbf{z} = (x, \mathbf{y})(\widehat{\mathbf{z}} = \Psi(\mathbf{z}) = (\widehat{x}, \widehat{\mathbf{y}})$, with $\widehat{x} = x$ and $\widehat{\mathbf{y}} = \psi_x(\mathbf{y}))$. Without loss of generality, we assume Ω_{1D} to be the subset of Ω with $\mathbf{y} = \mathbf{0}$, i.e., Ω_{1D} coincides with the centerline of Ω. We assume that ψ_x is a C^1-diffeomorphism, for all $x \in \Omega_{1D}$, as well as Ψ is differentiable with respect to $\mathbf{z}$. This last assumption essentially excludes the presence of kinks along Γ_*. Figure 1 provides a sketch of the map Ψ for $d = 3$.

In two dimensions, it is always possible to identify the map ψ_x with the linear transformation $\widehat{y} = \psi_x(y) = y/L(x)$, with $L(x) = \text{meas}(\gamma_x)$. In three dimensions this choice is still possible for some configurations, for instance when Ω is a cylindrical domain. In this case $L(x)$ is the diameter of the pipe along the centerline x (as it is in Fig. 1).

Finally, we introduce the Jacobian associated with the map Ψ given by

$$\mathscr{J}(\mathbf{z}) = \frac{\partial \Psi}{\partial \mathbf{z}} = \begin{bmatrix} 1 & 0 \\ \mathscr{D}_1(\mathbf{z}) & \mathscr{D}_2(\mathbf{z}) \end{bmatrix} \in \mathbf{IR}^{d\times d}, \tag{1}$$

with $\mathscr{D}_1(\mathbf{z}) = \partial\psi_x/\partial x \in \mathbf{IR}^{d-1}$, $\mathscr{D}_2(\mathbf{z}) = \nabla_{\mathbf{y}}\psi_x \in \mathbf{IR}^{(d-1)\times(d-1)}$ and where $\nabla_{\mathbf{y}}$ stands for the gradient with respect to $\mathbf{y}$. Notice that the first row in $\mathscr{J}(\mathbf{z})$ is the same as in the identity matrix since the map Ψ does not modify the supporting fiber Ω_{1D}. Moreover, in the 2D case, $\mathscr{D}_1(\mathbf{z})$ and $\mathscr{D}_2(\mathbf{z})$ simplify to $-L'(x)y/L^2(x)$ and $1/L(x)$, respectively.

Remark 1 The geometric framework introduced above remarkably differs with respect to the setting used in [2, 4, 28–30], where 1D models are associated with the transverse directions while the supporting fiber has dimension $(d-1)$.

Remark 2 (Non-rectilinear domains) Curvilinear domains are tackled in [21]. In such a case the supporting fiber Ω_{1D} coincides with a one-dimensional curved domain

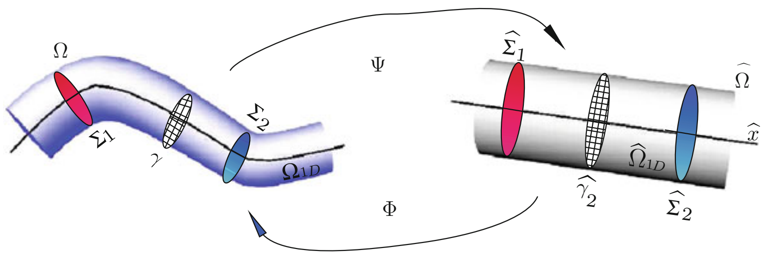


Fig. 2 Maps involved in the Hi-Mod procedure applied to a *curved* three-dimensional domain

as shown in Fig. 2. Now, the map Ψ becomes more complicated since also the deformation of the centerline has to be taken into account. Thus, we have $\widehat{\mathbf{z}} = \Psi(\mathbf{z}) = (\Psi_1(\mathbf{z}), \Psi_2(\mathbf{z}))$, with $\widehat{x} = \Psi_1(\mathbf{z})$ and $\widehat{\mathbf{y}} = \Psi_2(\mathbf{z})$. The definition of the Jacobian (1) accordingly changes in

$$\mathscr{J}(\mathbf{z}) = \frac{\partial \Psi}{\partial \mathbf{z}} = \begin{bmatrix} \frac{\partial \Psi_1}{\partial x} & \nabla_{\mathbf{y}} \Psi_1 \\ \frac{\partial \Psi_2}{\partial x} & \nabla_{\mathbf{y}} \Psi_2 \end{bmatrix}.$$

Moreover, the inverse map between $\widehat{\Omega}$ and Ω does not coincide now with Ψ^{-1} and is defined apart as $\Phi : \widehat{\Omega} \to \Omega$ so that $\mathbf{z} = \Phi(\widehat{\mathbf{z}}) = (\Phi_1(\widehat{\mathbf{z}}), \Phi_2(\widehat{\mathbf{z}}))$, with $x = \Phi_1(\widehat{\mathbf{z}})$ and $\mathbf{y} = \Phi_2(\widehat{\mathbf{z}})$ (see Fig. 2). Finally, we assume that both Ψ and Φ are differentiable with respect to $\mathbf{z}$.

3 Three Different Hi-Mod Techniques

Three different approaches have been proposed so far to perform a Hi-Mod reduction. The common idea is to exploit the fiber structure introduced on Ω by tackling in a different way the dependence of the solution on the dominant and on the transverse directions. In the sections below we present separately the three Hi-Mod techniques by following the chronological order of their proposal in the literature.

For this purpose, we first introduce the *full problem*, i.e., the problem we aim at reducing. We consider a generic second-order elliptic problem in the weak form, given by

$$\text{find } u \in V \quad : \quad a(u, v) = \mathscr{F}(v) \quad \forall v \in V, \tag{2}$$

with $V \subseteq H^1(\Omega)$ a Hilbert space, $a(\cdot, \cdot) : V \times V \to \mathbf{IR}$ a continuous and coercive bilinear form and $\mathscr{F}(\cdot) : V \to \mathbf{IR}$ a continuous linear functional, so that the Lax–Milgram lemma ensures the well-posedness of (2). The choice of the space V takes into account the boundary conditions assigned on $\partial\Omega$. Standard notation for the Sobolev spaces as well as for the spaces of functions bounded a.e. in Ω is adopted (see, e.g., [18]).

3.1 Uniform Hi-Mod Reduction

This model reduction technique is first introduced in [11] and then more rigorously investigated in [22]. The dominant and the trasverse dynamics of the problem are described via two different functional representations, namely

1. a space $V_{1D} \subseteq H^1(\Omega_{1D})$ spanned by functions defined on Ω_{1D} and compatible with the boundary conditions enforced along Γ_0 and Γ_1;
2. a *modal basis* $\{\varphi_j\}_{j\in\mathbb{N}^+} \in H^1(\widehat{\gamma}_{d-1})$ of functions orthonormal with respect to the L^2-scalar product on $\widehat{\gamma}_{d-1}$, properly including the boundary data assigned on Γ_*.

By properly combining the space V_{1D} with the modal basis, we define the uniform hierarchically reduced space

$$V_m = \left\{ v_m(x, \mathbf{y}) = \sum_{j=1}^{m} \widetilde{v}_j(x)\, \varphi_j(\psi_x(\mathbf{y})),\ \text{with}\, \widetilde{v}_j \in V_{1D},\ x \in \Omega_{1D},\ \mathbf{y} \in \gamma_x \right\}, \tag{3}$$

where $m \in \mathbb{N}$ is a given integer, fixed *a priori*. Space V_m identifies a *hierarchy* of models: the number m of included modes determines the accuracy of the reduced model that, in principle, can be tuned arbitrarily close to the full one. We call this approach *uniform* since we use the same value for m over the entire domain. Notice that, due to the orthonormality of the modal functions, the frequency coefficients in (3) are given by $\widetilde{v}_j(x) = \int_{\widehat{\gamma}_{d-1}} v_m\big(x, \psi_x^{-1}(\widehat{\mathbf{y}})\big)\, \varphi_j(\widehat{\mathbf{y}})\, d\widehat{\mathbf{y}}$, with $j = 1, \ldots, m$.

We can now state the *uniform Hi-Mod formulation*: given a modal index $m \in \mathbb{N}^+$,

$$\text{find}\, u_m \in V_m \quad : \quad a(u_m, v_m) = \mathscr{F}(v_m) \quad \forall v_m \in V_m. \tag{4}$$

To guarantee the well-posedness of formulation (4), we introduce a conformity hypothesis on the space V_m (i.e., $V_m \subset V$, for all $m \in \mathbb{N}^+$) and we exploit the well-posedness assumed for problem (2). On the other hand, the convergence of u_m to u is obtained by adding a spectral approximability assumption on V_m, i.e., we demand that, for any $v \in V$, $\lim_{m\to+\infty} \inf_{v_m\in V_m} \|v - v_m\|_V = 0$. Indeed, let $e_m = u - u_m \in V$ be the uniform modeling error which, via the conformity hypothesis, satisfies the modeling orthogonality property $a(e_m, v_m) = 0$, for any $v_m \in V_m$. It can be easily proved that the spectral optimality property holds, i.e., that $\|e_m\|_V \leq C\, \inf_{v_m\in V_m} \|u - v_m\|_V$, with C a constant depending on both the continuity and the coercivity constants of $a(\cdot, \cdot)$. Then, thanks to the spectral approximability hypothesis, the convergence immediately follows.

Remark 3 (Choice of the modal basis) Different choices are possible for the modal basis $\{\varphi_j\}_{j\in\mathbb{N}^+}$. Of course, this choice strictly depends on the boundary conditions assigned along Γ_*. So far we have essentially used trigonometric functions, i.e., we have essentially considered homogeneous Dirichlet boundary conditions on the horizontal sides of Ω. In [22] we have also checked the performances associated with Legendre polynomials with similar results. More recently, in order to deal with more general boundary conditions, we have introduced a new type of modal basis called *educated basis* (see [3]). The idea is to solve an auxiliary Sturm-Liouville problem on the transverse reference fiber $\widehat{\gamma}_{d-1}$ in order to build a modal basis which automatically includes the boundary conditions assigned along Γ_*. This approach has been successfully validated both in 2D and 3D. We finally remark that the choice of the modal basis, together with the regularity of the full solution u, influences also the rate of the modal convergence.

3.1.1 Discrete Uniform Hi-Mod Reduction

To make the uniform Hi-Mod approach useful in practice, we consider the discrete counterpart of formulation (4). Following [11, 22], we discretize the main dynamics via standard 1D finite elements while preserving the modal expansion to describe the transverse features. For this purpose, we consider a subdivision $\mathscr{T}_h$ of Ω_{1D} into subintervals $K_i = (x_{i-1}, x_i)$ of width $h_i = x_i - x_{i-1}$, with $h = \max_i h_i$. Then, we consider a conforming finite element space $V_{1D}^h \subset V_{1D}$ associated with $\mathscr{T}_h$ such that $\dim(V_{1D}^h) = N_h < +\infty$, and we introduce a standard density hypothesis on the space V_{1D}^h. The *discrete uniform Hi-Mod reduction* can thus be stated as: given a modal index $m \in \mathbb{N}$,

$$\text{find } u_m^h \in V_m^h \quad : \quad a(u_m^h, v_m^h) = \mathscr{F}(v_m^h) \quad \forall v_m^h \in V_m^h, \tag{5}$$

where the discrete uniform hierarchically reduced space is

$$V_m^h = \left\{ v_m^h(x, \mathbf{y}) = \sum_{j=1}^{m} \widetilde{v}_j^{\,h}(x)\, \varphi_j(\psi_x(\mathbf{y})), \text{ with } \widetilde{v}_j^{\,h} \in V_{1D}^h,\ x \in \Omega_{1D},\ \mathbf{y} \in \gamma_x \right\} \subset V_m, \tag{6}$$

the last inclusion being guaranteed by the conformity assumption on V_{1D}^h.

In [22] it is proved that the uniform global error $e_m^h = u - u_m^h \in V$ (which includes both the model $(u - u_m)$ and the discretization $(u_m - u_m^h)$ error contributions) vanishes for $m \to \infty$ and $h \to 0$. Some results are available in the literature concerning the rate of convergence of e_m^h (we refer, e.g., to [8, 9, 15]). Moreover, a numerical convergence study of the discrete Hi-Mod formulation (5) is available in Sect. 3.4.1 of [22]. For sufficiently smooth functions, we recover the convergence rate expected from Theorems 2.1 and 3.2 in [8] for e_m^h, namely quadratic for the L^2–norm and linear for the H^1–norm, with respect to both m^{-1} and h, respectively.

From a computational viewpoint, formulation (5) leads to solve a system of coupled 1D problems instead of the full d-dimensional problem. This is expected to be computationally advantageous, in particular if the full problem is three-dimensional and for a modal index m small enough. To detail the effects of a Hi-Mod reduction, let us exemplify the uniform Hi-Mod procedure on the standard Poisson problem completed with full homogeneous Dirichlet boundary conditions.

For the sake of simplicity, we focus on the 2D case. The full space V coincides with $H_0^1(\Omega)$, while $V_{1D} = H_0^1(\Omega_{1D})$. Moreover, the modal functions φ_j vanish on Γ_*. The bilinear and linear forms in (2) are given by $a(u, v) = \int_\Omega \nabla u \cdot \nabla v\, dxdy$ and $\mathscr{F}(v) = \int_\Omega f v\, dxdy$, respectively with $f \in L^2(\Omega)$. Now, we first consider the modal representation $u_m^h(x, y) = \sum_{j=1}^m \widetilde{u}_j^{\,h}(x)\, \varphi_j(\psi_x(y))$ for the Hi-Mod approximation u_m^h; then, we expand each modal coefficient $\widetilde{u}_j^{\,h}$ in terms of the finite element basis $\{\theta_i\}_{i=1}^{N_h}$ as $\widetilde{u}_j^{\,h}(x) = \sum_{i=1}^{N_h} \widetilde{u}_{j,\,i}\, \theta_i(x)$ to get the global expansion

$$u_m^h(x, y) = \sum_{j=1}^{m} \sum_{i=1}^{N_h} \widetilde{u}_{j,\,i}\, \theta_i(x)\, \varphi_j(\psi_x(y)). \tag{7}$$

Thus, the actual unknowns are the coefficients $\widetilde{u}_{j,\,i}$, for $j = 1, \ldots, m$ and $i = 1, \ldots, N_h$. By plugging this representation in (5) after choosing as test function $v_m^h(x, y) = \varphi_k(\psi_x(y))\, \theta_l(x)$, we obtain the following system of 1D coupled problems: for $j = 1, \ldots, m$ and $i = 1, \ldots, N_h$, find the coefficients $\widetilde{u}_{j,\,i}$ such that, for any $k = 1, \ldots, m$ and $l = 1, \ldots, N_h$, it holds

$$\sum_{j=1}^{m} \sum_{i=1}^{N_h} \Bigg\{ \int_{\Omega_{1D}} \Bigg[\; \widehat{r}_{kj}^{\,1,1}(x)\, \frac{d\theta_i(x)}{dx}\, \frac{d\theta_l(x)}{dx} + \widehat{r}_{kj}^{\,1,0}(x)\, \frac{d\theta_i(x)}{dx}\, \theta_l(x) + \widehat{r}_{kj}^{\,0,1}(x)\, \theta_i(x)\, \frac{d\theta_l(x)}{dx}$$
$$+ \; \widehat{r}_{kj}^{\,0,0}(x)\, \theta_i(x)\, \theta_l(x) \Bigg] dx \Bigg\} \widetilde{u}_{j,\,i} = \int_{\Omega_{1D}} \widehat{f}_k(x)\, \theta_l(x)\, dx, \tag{8}$$

where $\widehat{f}_k(x) = \int_{\widehat{\gamma}_1} f(x, \psi_x^{-1}(\widehat{y}))\varphi_k(\widehat{y})\, \left|\mathscr{D}_2^{-1}\big(x, \psi_x^{-1}(\widehat{y})\big)\right| d\widehat{y}$, while we have that $\widehat{r}_{kj}^{\,s,t}(x) = \int_{\widehat{\gamma}_1} r_{kj}^{s,t}(x, \widehat{y})\, \left|\mathscr{D}_2^{-1}\big(x, \psi_x^{-1}(\widehat{y})\big)\right| d\widehat{y}$, for $s, t = 0, 1$, with

$$\begin{aligned}
&r_{kj}^{1,1}(x, \widehat{y}) = \varphi_j(\widehat{y})\, \varphi_k(\widehat{y}), \quad r_{kj}^{1,0}(x, \widehat{y}) = \varphi_j(\widehat{y})\, \varphi_k'(\widehat{y})\, \mathscr{D}_1\big(x, \psi_x^{-1}(\widehat{y})\big),\\
&r_{kj}^{0,1}(x, \widehat{y}) = \varphi_j'(\widehat{y})\, \varphi_k(\widehat{y})\, \mathscr{D}_1\big(x, \psi_x^{-1}(\widehat{y})\big),\\
&r_{kj}^{0,0}(x, \widehat{y}) = \varphi_j'(\widehat{y})\, \varphi_k'(\widehat{y})\, \Big\{\big[\mathscr{D}_1\big(x, \psi_x^{-1}(\widehat{y})\big)\big]^2 + \big[\mathscr{D}_2\big(x, \psi_x^{-1}(\widehat{y})\big)\big]^2\Big\}.
\end{aligned} \tag{9}$$

The quantities $\widehat{r}_{kj}^{\,s,t}$ collect the transverse contributions. Notice that, via the map ψ_x, the reduced system is solved on the reference domain $\widehat{\Omega}$. Coefficients in (9) simplify when the map ψ_x is linear, since $\mathscr{D}_2\big(\mathbf{z}\big)$ reduces to $L(x)^{-1}$.

Analogous computations can be performed, in a straightforward way, starting from a full three-dimensional model and for a generic second-order elliptic problem. We refer the reader, for instance, to [11] for the detailed computations associated with the reduction of a full 2d advection-diffusion-reaction problem.

Remark 4 System (8) shows that a full purely diffusive problem yields low-order contributions in the reduced framework. However, the first-order terms yielded by the reduction procedure are always weighted by the diffusive coefficient. Consequently,

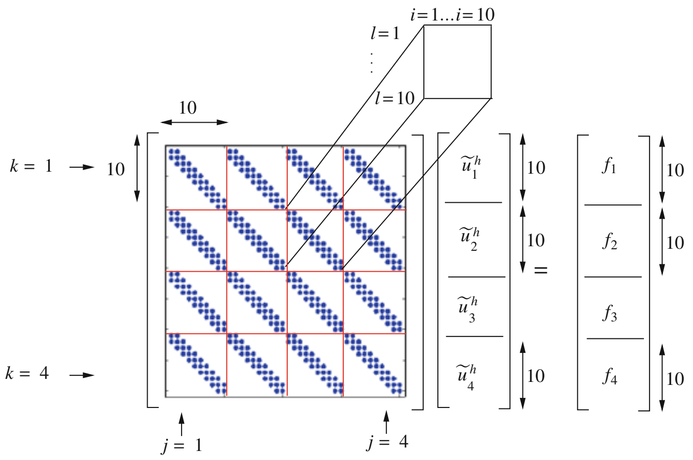


Fig. 3 Sketch of the linear system associated with a uniform Hi-Mod reduction, for $m = 4$ and $N_h = 10$, with $[f_k]_l = \mathscr{F}(\vartheta_l \varphi_k)$, $\widetilde{u}_k^h$ the modal coefficients of u_m^h, for $k = 1, \ldots, 4$ and $l = 1, \ldots, 10$

possible instabilities due to a dominant advection or reaction are in general absent provided that the deformation indices $\mathscr{D}_1(\mathbf{z})$ and $\mathscr{D}_2(\mathbf{z})$ are small enough.

From an algebraic viewpoint, the discrete uniform Hi-Mod formulation leads to solve a linear system characterized by an $mN_h \times mN_h$ block matrix A. With reference to system (8), we distinguish in A a macrostructure associated with the modes and identified by the indices k and j which run on the block rows and block columns, respectively; on the other hand, the indices l and i are related to the finite element basis and span the rows and columns of each block, respectively. Each $N_h \times N_h$ block preserves the sparsity pattern typical of the the adopted finite element discretization. This can be advantageously exploited both in storing and solving the associated system. In particular, the common sparsity pattern can be stored once and for all. In Fig. 3 we show an example of Hi-Mod sparsity pattern for $m = 4$ and $N_h = 10$ for linear finite elements. Blockwise, we recognize the standard tridiagonal pattern.

3.1.2 A Numerical Example

We provide a numerical example to assess the performances of the uniform Hi-Mod reduction. We solve on the domain $\Omega = (0, 7) \times (0, 2)$ the standard advection-diffusion problem $-\Delta u + \mathbf{b} \cdot \nabla u = f$, with $\mathbf{b} = (-20, 0)^T$ a backward field and $f(x, y) = 50\chi_{D_1 \cup D_2}(x, y)$, where $D_1 = \{(x, y) : (x - 4.2)^2 + (y - 1.5)^2 \leq 0.04\}$ and $D_2 = \{(x, y) : (x - 3.7)^2 + (y - 0.5)^2 < 0.03\}$ are two circular misaligned regions with a different radius. Full homogeneous Dirichlet boundary conditions are enforced on $\partial\Omega$. Figure 4, top displays the contour-plots of the full solution computed via a standard 2D linear finite element discretization on a uniform unstructured grid

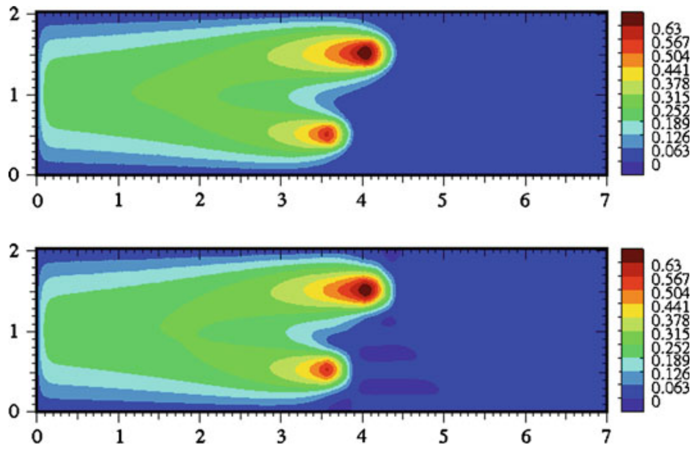


Fig. 4 Full solution (*top*); uniform Hi-Mod solution u_9^h (*bottom*)

with 31,264 elements. Due to the strong advective field and the assigned boundary conditions, the solution is basically flat for $x > 4.5$, while it exhibits large variations in the leftmost part of the domain with boundary layers in correspondence with $\{(x, 0)$ for $0 \leq x \leq 3.5\}$ and $\{(x, 2)$ for $0 \leq x \leq 4\}$ and, more significantly, along $\{(0, y)$ for $0 \leq y \leq 2\}$. In Fig. 4, bottom we display the uniform Hi-Mod solution associated with $m = 9$ modal functions and with a 1D mesh $\mathscr{T}_h$ of uniform size $h = 0.01$. The agreement between the full and the reduced solution is pretty good. Notice that we do not resort to any stabilization scheme: the choice made for h guarantees that the local Péclet number corresponding to the advective field $\mathbf{b}$ is strictly less than one. However, the actual advective term in the Hi-Mod reduced formulation also depends on $\mathscr{D}_1(\mathbf{z})$ (see [11]). This last contribution could make the chosen h locally insufficient to ensure the stability of the discretization scheme. This could explain the negative values of the reduced solution in the darker blue areas near the two sources (see Fig. 4, bottom) in contrast to the minimum value zero assumed by the full solution.

3.2 The Piecewise Hi-Mod Reduction

Figure 4, bottom clearly shows the main limit of a uniform Hi-Mod reduction. To accurately approximate a full solution with *local* strong transverse components, we need to employ a large number of modes over the whole domain, i.e., also where the transverse dynamics are not relevant. This implies a waste in terms of computational cost. The piecewise Hi-Mod formulation aims at improving the computational efficiency of a Hi-Mod reduction by employing a different number of modes in different parts of Ω. Large values are associated with the zones where the transverse dynamics

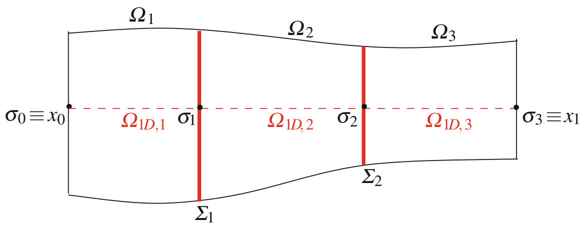


Fig. 5 Example of 2D partition $\mathscr{T}_\Omega$, for $s = 3$

are important, while small values are selected where the 1D behavior is dominant. As a consequence, the modal index m becomes a vector, called modal multi-index, which collects the number of modes used in the different portions of Ω. This justifies the name of the approach.

In order to formulate the piecewise counterpart of (4), we need to introduce a number of definitions. To simplify the discussion, we focus on the 2D case and we assume to identify, via some criterion, three areas Ω_1, Ω_2 and Ω_3 in Ω where a different number m_i of modal functions, for $i = 1, 2, 3$, is employed. In particular, we denote by Σ_1 and Σ_2 the interface between $\Omega_1 - \Omega_2$ and $\Omega_2 - \Omega_3$, respectively and by $\Omega_{1D,i} = \Omega_{1D} \cap \Omega_i = (\sigma_{i-1}, \sigma_i)$ the subinterval of Ω_{1D} associated with the subdomain Ω_i so that $\Omega_i = \bigcup_{x\in\Omega_{1D,i}} \{x\} \times \gamma_x$, $\cup_{i=1}^{3} \overline{\Omega}_{1D,i} = \overline{\Omega}_{1D}$, $\Omega_{1D,i} \cap \Omega_{1D,\tilde{i}} = \emptyset$ for $i \neq \tilde{i}$ and $i, \tilde{i} = 1, 2, 3$, and where $\sigma_0 \equiv x_0$, $\sigma_3 \equiv x_1$ (see Fig. 5). Finally, we introduce the two-dimensional broken Sobolev space $H^1(\Omega, \mathscr{T}_\Omega)$ associated with the partition $\mathscr{T}_\Omega = \{\Omega_i\}_{i=1}^{3}$ of Ω, properly modified according to the boundary conditions assigned on $\partial\Omega$ [17]. The inclusion $V \subset H^1(\Omega, \mathscr{T}_\Omega)$ holds. We can define now the piecewise hierarchically reduced space

$$
\begin{aligned}
V_{\mathbf{m}}(\mathscr{T}_\Omega) = \Big\{ & v_{\mathbf{m}} \in L^2(\Omega) : v_{\mathbf{m}}|_{\Omega_i}(x, y) = \sum_{j=1}^{m_i} \widetilde{v}_j^{\,i}(x)\, \varphi_j(\psi_x(y)),\ i = 1, 2, 3, \\
& \widetilde{v}_j^{\,i} \in H^1(\Omega_{1D,i}) \ :\ \forall \tilde{j} = 1, \ldots, m_\perp^p \text{ with } p = 1, 2, \qquad (10) \\
& \int_{\widehat{\gamma}_1} \big(v_{\mathbf{m}}|_{\Omega_{p+1}}(\sigma_p, \psi_{\sigma_p}^{-1}(\widehat{y})) - v_{\mathbf{m}}|_{\Omega_p}(\sigma_p, \psi_{\sigma_p}^{-1}(\widehat{y})) \big)\, \varphi_{\tilde{j}}(\widehat{y})\, d\widehat{y} = 0 \Big\},
\end{aligned}
$$

with $\mathbf{m} = \{m_i\}_{i=1}^{3} \in [\mathbb{N}^+]^3$ a given modal multi-index and $m_\perp^p = \min(m_p, m_{p+1})$. The reduced space $V_{\mathbf{m}}(\mathscr{T}_\Omega)$ is a subset of $H^1(\Omega, \mathscr{T}_\Omega)$. The ingredients used to define space $V_{\mathbf{m}}(\mathscr{T}_\Omega)$ are essentially the same characterizing the uniform space V_m, i.e., a one-dimensional space to describe the main stream and a modal basis for the transverse dynamics, even though now the 1D space has to be localized to the different subdomains Ω_i. The interface condition in (10) weakly enforces the continuity of the minimum number $m_{\min}$ of transverse modes in the whole Ω. Different strategies can be pursued to enforce this condition. In [11] we resort to an iterative substructuring Dirichlet/Neumann method (see, e.g., [26, 27]). Alternatively, different domain decomposition algorithms as well as techniques based on an appropriate definition of the reduced space can be pursued, as, for instance, in [2, 4, 30].

The *piecewise Hi-Mod formulation* can thus be stated: given a modal multi-index $\mathbf{m} \in [\mathbb{N}^+]^3$,

$$\text{find } u_{\mathbf{m}} \in V_{\mathbf{m}}(\mathscr{T}_\Omega) \quad : \quad a_{\mathscr{T}_\Omega}(u_{\mathbf{m}}, v_{\mathbf{m}}) = \mathscr{F}_{\mathscr{T}_\Omega}(v_{\mathbf{m}}) \quad \forall v_{\mathbf{m}} \in V_{\mathbf{m}}(\mathscr{T}_\Omega), \tag{11}$$

with $a_{\mathscr{T}_\Omega}(u_{\mathbf{m}}, v_{\mathbf{m}}) = \sum_{i=1}^{3} a_i(u_{\mathbf{m}}|_{\Omega_i}, v_{\mathbf{m}}|_{\Omega_i})$ and $\mathscr{F}_{\mathscr{T}_\Omega}(v_{\mathbf{m}}) = \sum_{i=1}^{3} \mathscr{F}_i(v_{\mathbf{m}}|_{\Omega_i})$, where $a_i(\cdot,\cdot)$ and $\mathscr{F}_i(\cdot)$ are the restrictions to the subdomain Ω_i of the bilinear and linear form in (2), respectively for $i = 1, 2, 3$. We remark that the piecewise Hi-Mod formulation is well-posed in $V_{\mathbf{m}}(\mathscr{T}_\Omega)$ with respect to the broken energy norm $\|v_{\mathbf{m}}\|_{\mathscr{T}_\Omega} = \left(\sum_{i=1}^{3} \|v_{\mathbf{m}}|_{\Omega_i}\|^2_{H^1(\Omega_i)}\right)^{1/2}$ [17]. As trivial subcase, formulation (11) admits the uniform Hi-Mod reduction when $\mathbf{m} = (\widetilde{m}, \widetilde{m}, \widetilde{m})^T$ for a certain $\widetilde{m} \in \mathbb{N}$. The weak imposition of the continuity does not necessarily guarantee an H^1-conforming approximation $u_{\mathbf{m}}$ to the full solution u in (2): the continuity on Ω of both the trace and the flux of $u_{\mathbf{m}}$ is ensured to the first $m_{\min}$ modal components only. A global conforming approximation is yielded only if $m_i \geq m_{i+1}$, for $i = 1, 2$ [22].

The extension of the picewise Hi-Mod formulation to a generic number s of subdomains is immediate.

3.2.1 Discrete Piecewise Hi-Mod Reduction

We consider now the discrete couterpart of formulation (11) by referring to a generic partition $\mathscr{T}_\Omega = \{\Omega_i\}_{i=1}^{s}$ of $\Omega \subset \mathbf{IR}^d$, $d = 2, 3$. We introduce the subdivision $\mathscr{T}_h^i$ of $\Omega_{1D,i}$ into the subintervals $K_l^i = (x_{l-1}^i, x_l^i)$ of width $h_l^i = x_l^i - x_{l-1}^i$ for $l = 1, \ldots, n_i$, $i = 1, \ldots, s$ and $n_i \in \mathbb{N}^+$. The discrete piecewise hierarchically reduced space is

$$V_{\mathbf{m}}^h(\mathscr{T}_\Omega, \{\mathscr{T}_h^i\}) = \Big\{ v_{\mathbf{m}}^h \in V_{\mathbf{m}}(\mathscr{T}_\Omega) \ : \ v_{\mathbf{m}}^h|_{\Omega_i}(x, \mathbf{y}) = \sum_{j=1}^{m_i} \widetilde{v}_j^{\,i,h}(x)\, \varphi_j(\psi_x(\mathbf{y})) \tag{12}$$
$$\forall i = 1, \ldots, s,\ \widetilde{v}_j^{\,i,h} \in V_{1D}^{i,h} \Big\} \subset V_{\mathbf{m}}(\mathscr{T}_\Omega),$$

where $V_{1D}^{i,h} \subset H^1(\Omega_{1D,i})$ is a finite element space associated with $\mathscr{T}_h^i$, such that $\dim(V_{1D}^{i,h}) = N_h^i < +\infty$. A standard density assumption is postulated on the spaces $V_{1D}^{i,h}$. The *discrete piecewise Hi-Mod reduction* reads: given a modal multi-index $\mathbf{m} \in [\mathbb{N}^+]^s$,

$$\text{find } u_{\mathbf{m}}^h \in V_{\mathbf{m}}^h(\mathscr{T}_\Omega, \{\mathscr{T}_h^i\}) \ : \ a_{\mathscr{T}_\Omega}(u_{\mathbf{m}}^h, v_{\mathbf{m}}^h) = \mathscr{F}_{\mathscr{T}_\Omega}(v_{\mathbf{m}}^h) \quad \forall v_{\mathbf{m}}^h \in V_{\mathbf{m}}^h(\mathscr{T}_\Omega, \{\mathscr{T}_h^i\}). \tag{13}$$

The actual unknowns are the modal coefficients $\widetilde{u}_j^{i,h}$ of $u_{\mathbf{m}}^h$, for $j = 1, \ldots, m_i$ and $i = 1, \ldots, s$.

Despite the possible non-conformity of the reduced solution, we can state also for this formulation a Galerkin orthogonality property, simply by subtracting (13) from (11) for $v_{\mathbf{m}} = v_{\mathbf{m}}^h$, to get

$$a_{\mathscr{T}_\Omega}(\varepsilon_{\mathbf{m}}^h, v_{\mathbf{m}}^h) = 0 \quad \forall v_{\mathbf{m}}^h \in V_{\mathbf{m}}^h(\mathscr{T}_\Omega, \{\mathscr{T}_h^i\}), \tag{14}$$

with $\varepsilon_{\mathbf{m}}^h = u_{\mathbf{m}} - u_{\mathbf{m}}^h$ the piecewise discretization error associated with (13).

From a computational viewpoint, the discrete piecewise formulation is associated with the resolution of s linear systems of coupled 1D problems, one for each subdomain. As in Fig. 3, each system is characterized by a sparse $m_i \times m_i$ block matrix and each $N_h^i \times N_h^i$ block exhibits the sparsity pattern typical of the chosen 1D finite element approximation. In particular, since we employ an iterative substructuring scheme to impose the interface condition between the subdomains, we solve the s linear systems at each iteration of the Dirichlet/Neumann algorithm. Of course, the factorization of the correspoding matrices is stored once and for all at the first iteration. Particular attention has to be paid to the well-posedness of each subproblem which means to properly assign the boundary conditions on each Ω_i.

An *a priori* analysis for the piecewise global error $e_{\mathbf{m}}^h = u - u_{\mathbf{m}}^h = e_{\mathbf{m}} + \varepsilon_{\mathbf{m}}^h \in H^1(\Omega, \mathscr{T}_\Omega)$, with $e_{\mathbf{m}} = u - u_{\mathbf{m}}$ the piecewise modeling error and $\varepsilon_{\mathbf{m}}^h$ defined as in (14), can be found in [22], under the assumption that the Dirichlet-Neumann scheme converges. In particular, as previously pointed out, if $\mathbf{m}$ is a strictly decreasing multi-index, i.e., $m_i > m_{\tilde{i}}$ for any $i, \tilde{i} = 1, \ldots, s$ with $i > \tilde{i}$, a conforming approximation $u_{\mathbf{m}}^h$ is guaranteed and the error can be bounded via the standard Céa lemma. On the other hand, for an increasing multi-index $\mathbf{m}$, we get an error bound consisting of the usual best approximation term plus an additional correction due to the high-frequency components near the interfaces, which are responsible for the nonconformity (see [22, Sect. 4.2.2] for a detailed discussion).

3.2.2 A Numerical Example

We apply the piecewise Hi-Mod reduction to the same test case as in Sect. 3.1.2. By exploiting the intrinsic heterogeneity of the full solution u, we divide Ω into three subdomains $\Omega_1 = (0, 2.5) \times (0, 2)$, $\Omega_2 = (2.5, 4.5) \times (0, 2)$ and $\Omega_3 = (4.5, 7) \times (0, 2)$ and we resort to $m_1 = 2$, $m_2 = 7$ and $m_3 = 1$ modal functions, respectively. This choice is suggested by the behaviour itself of the full solution since the most complex dynamics take place in the central part of Ω, while the solution u is essentially flat in the rightmost part of the domain. At both the interfaces $\Sigma_1 = \{2.5\} \times (0, 2)$ and $\Sigma_2 = \{4.5\} \times (0, 2)$ we apply a Dirichlet/Neumann scheme with relaxation (i.e., Dirichlet conditions are prescribed when solving the leftmost subdomain, Neumann conditions for the rightmost one), by setting the relaxation parameter to 0.5. This choice guarantees the well-posedness of each problem in Ω_i since the field $\mathbf{b}$ is backward. Notice that, in the presence of a forward advective

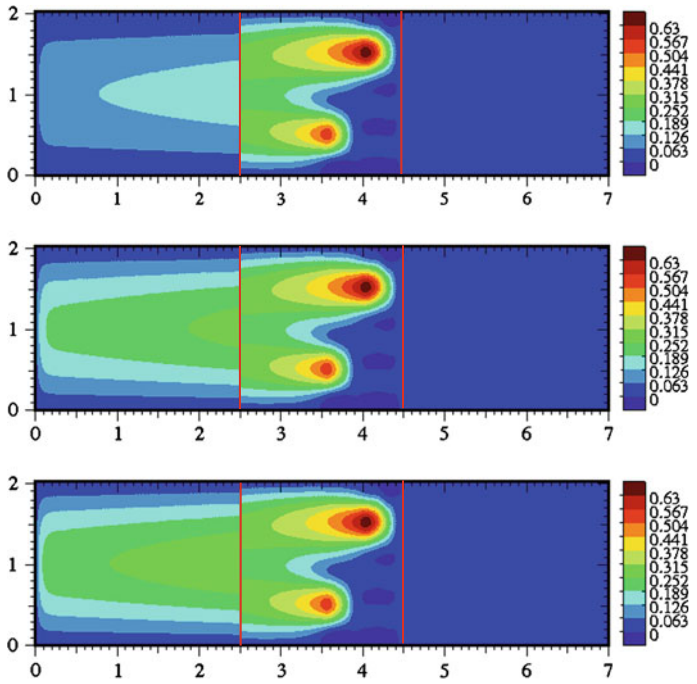


Fig. 6 Piecewise Hi-Mod solution $u^h_{2,7,1}$ at the second (*top*), fourth (*middle*) and seventh (*bottom*) iteration of the domain decomposition approach for $h = 0.01$

field, Neumann/Dirichlet conditions have to be imposed at both Σ_1 and Σ_2 to get well-posed subproblems (see Sect. 4.2.2 for an example).

Concerning the parameters characterizing the domain decomposition scheme, we select both the initial guesses at the interfaces identically equal to zero and we settle the convergence tolerance for the relative errors to 0.01. Finally, the same discretization step $h = 0.01$ is employed along all the three subintervals $\Omega_{1D,1} = (0, 2.5)$, $\Omega_{1D,2} = (2.5, 4.5)$, $\Omega_{1D,3} = (4.5, 7)$.

The chosen data lead the Dirichlet/Neumann scheme to converge after seven iterations. Figure 6 collects the contour plots of the piecewise Hi-Mod solution associated with the second, the fourth and the last iteration. For the sake of comparison, we employ the same colour map used for the uniform case in Fig. 4. A considerable model discontinuity can be observed at $x = 2.5$. The model discontinuity is due to the fact that $m_1 < m_2$. Moreover, the first interface Σ_1 is located in an area of Ω where the transverse dynamics are important. This makes even more relevant the change in the number of modes. On the contrary, no model discontinuity appears at $x = 4.5$, where the solution is completely flat and since $m_2 > m_3$. As already remarked in [22], it can be verified that a finer discretization step does not significantly reduce the model jump.

3.3 The Pointwise Hi-Mod Reduction

As shown in the previous section, the piecewise Hi-Mod reduction may lead to a computational improvement with respect to the uniform approach when dealing with phenomena with significant transverse dynamics localized in a certain portion of the domain. In this perspective, the best computational advantage is attained when one can calibrate the subdomain with the largest modal index exactly to fit the region with significant transverse dynamics. This is in some sense the spirit behind a pointwise Hi-Mod reduction [24]. In this case, the modal functions are pointwise-tuned, which justifies the name assigned to this method. Now, the modes are associated with the nodes of the finite element partition, in contrast to the piecewise approach where the subdomain Ω_i shares the same number of modal functions. Due to the association mode-node, the pointwise Hi-Mod reduction makes sense only in a discrete context.

To settle the pointwise formulation, we move from the discrete modal expansion (7) that we properly rewrite as

$$u_m^h(x, y) = \sum_{i=1}^{N_h} \Big[\sum_{j=1}^{m} \widetilde{u}_{j,i}\, \varphi_j(\psi_x(y)) \Big] \theta_i(x). \tag{15}$$

We remark that, in this expansion, the leading role is taken by the sum on the finite element nodes while in (7) by the one on the modes. Inspired by representation (15), we introduce a new definition for the discrete Hi-Mod space, where we allow the number m of the modal basis functions to vary on each finite element node x_i. Thus, the discrete pointwise hierarchically reduced space is

$$V_{\mathbf{M}}^h = \Big\{ v_{\mathbf{M}}^h(x, y) = \sum_{i=1}^{N_h} \Big[\sum_{j=1}^{m_i^N} \widetilde{v}_{j,i}^{\,h}\, \varphi_j(\psi_x(y)) \Big] \theta_i(x), \text{ with } x \in \Omega_{1D},\ y \in \gamma_x \Big\}, \tag{16}$$

where $\mathbf{M} = \{m_i^N\}_{i=1}^{N_h} \in [\mathbb{N}^+]^{N_h}$ is the modal multi-index collecting the number of modes for each finite element node.

The *pointwise Hi-Mod formulation* is given by: for a certain modal multi-index $\mathbf{M} \in [\mathbb{N}^+]^{N_h}$,

$$\text{find } u_{\mathbf{M}}^h \in V_{\mathbf{M}}^h \quad : \quad a(u_{\mathbf{M}}^h, v_{\mathbf{M}}^h) = \mathscr{F}(v_{\mathbf{M}}^h) \quad \forall v_{\mathbf{M}}^h \in V_{\mathbf{M}}^h, \tag{17}$$

where $a(\cdot, \cdot)$ and $\mathscr{F}(\cdot)$ coincides with the bilinear and linear forms in (2). From definition (16), it follows immediately that the nodewise Hi-Mod solution $u_{\mathbf{M}}^h$ is always H^1-conformal in Ω in contrast to the piecewise approach.

From an algebraic viewpoint, we have to solve a linear system whose coefficient matrix has a structure similar to that of the uniform case (with $m = \max_i m_i^N$), except for the fact that the rows and the columns associated with the finite element nodes x_i such that $m_i^N < \max_i m_i^N$ are deleted. This leads to solve a reduced system with

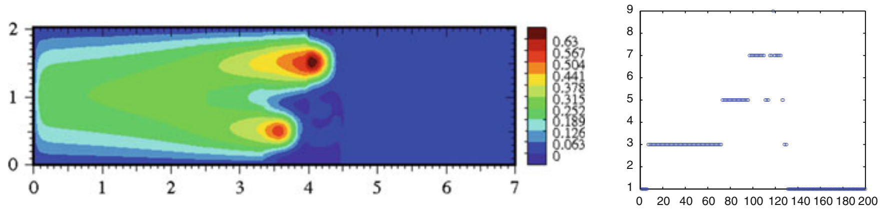


Fig. 7 Pointwise Hi-Mod solution (*left*); corresponding modal distribution (*right*)

respect to the uniform approach with a consequent computational saving. A further significative improvement in terms of computational costs is due to the fact that the nodewise formulation completely relieves us from using a domain decomposition scheme in the presence of a different number of modal functions in Ω. No iterative procedure is now demanded to get the reduced solution. On the contrary, a domain decomposition scheme could be advantageously exploited to deal with complex geometries, such as bifurcations, not taken into account by the standard setting in Sect. 2.

3.3.1 A Numerical Example

We assess the pointwise Hi-Mod procedure on the same test case used to validate both the uniform and the piecewise approaches. For this purpose, we introduce a finite element partition $\mathscr{T}_h$ of the supporting fiber $\Omega_{1D} = (0, 7)$ constituted by 200 equispaced nodes. Then, starting from the results in Figs. 4 and 6, we adopt the corresponding modal distribution shown in Fig. 7, right. In particular, we employ seven modes in the area around the two sources, except for a single node, close to the center of D_1, where nine modal functions are used and for few nodes in the zone between the two sources where only five modes are switched on.

Figure 7, left shows the contour plots of the pointwise Hi-Mod solution $u_{\mathbf{M}}^h$. It is fully comparable with the full solution in Fig. 4, top despite the presence of some negative areas as already detected in the uniform and in the piecewise reduced solutions. As expected, solution $u_{\mathbf{M}}^h$ is an H^1-conforming approximation, i.e., no model discontinuity is present.

4 How to Select the Model in the Hierarchy?

An appropriate choice of the modal index m in (5) or of the modal multi-indices $\mathbf{m}$ and $\mathbf{M}$ in (13) and (17), respectively is certainly the most critical issue of a Hi-Mod reduction procedure. In this section, we provide two different approaches to select the number of modal functions in Ω. The first method coincides essentially with a user-dependent criterion; the second technique is based on an *ad hoc* theoretical tool and makes the user free from any arbitrary choice.

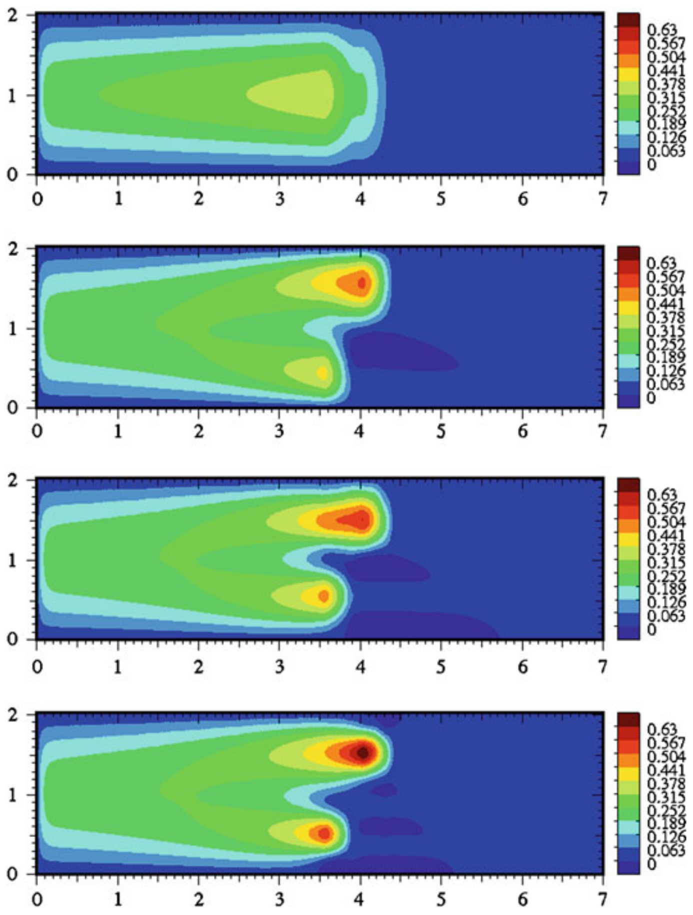


Fig. 8 Uniform Hi-Mod solutions: u_1^h, u_3^h, u_5^h, u_7^h (*top–bottom*)

4.1 An a Priori Approach

This approach is the most straightforward and the first strategy that we have pursued to select the number of modes to be used in Ω [11, 22].

We essentially distinguish the case when the user has no information about the dynamics of the problem to be reduced from the case when some hints on this problem are available. If no information is available, we can resort to a "trial and error" approach: we move from the computationally cheapest choice for m, i.e., $m = 1$. Then, we gradually increase such a value and we stop when the addition of the successive modal function does not significantly improve the accuracy of the reduced solution. This is the procedure that we have followed to get the uniform Hi-Mod solution u_9^h in Fig. 4, bottom. Figure 8 shows the contour plots of some

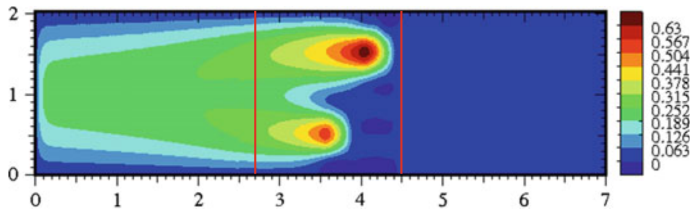


Fig. 9 Piecewise Hi-Mod solution $u^h_{4,7,1}$ at the last iteration of the domain decomposition approach for $h = 0.01$

of the reduced solutions yielded by this trial and error approach. It is evident that the accuracy of u^h_m increases as m gets larger. The presence of the two localized sources demands a rather large number of modes overall. While the behaviour of u is correctly described by a single mode on the rightmost part of Ω, at least five modes are necessary to recognize the sources at D_1 and D_2. Solution u^h_7 is very close to u^h_9 in Fig. 4, bottom. The only difference is a slight reduction of the negative areas for the choice $m = 9$.

When we have some knowledge about the full solution or about some previous run of the Hi-Mod procedure, we may fix directly the number of modes instead of gradually varying it. This is the approach used in the piecewise Hi-Mod reduction of Sect. 3.2.2. Indeed, Fig. 8 suggests that few modes are sufficient in Ω_1 and Ω_3, while at least 7 modes are demanded in Ω_2. Of course, other choices are allowed, driven, e.g., by specific user demands. For instance, when we aim at reducing the model discontinuity in Fig. 6, bottom, we simply increase the number of modes in Ω_1. If we employ, e.g., four instead of two modes in Ω_1 while preserving the same values for all the parameters involved in the domain decomposition scheme, we obtain, after 7 iterations, the Hi-Mod solution in Fig. 9. It is difficult to appreciate in this case the model discontinuity occurring along the interface $\Sigma_1 = \{2.5\} \times (0, 2)$.

The pointwise Hi-Mod reduction is the setting where an *a priori* choice of the modal multi-index is less immediate. This is essentially due to the large variability that can be assigned to the modal indices m^N_i, which now may vary nodewise. Such a variability represents a huge richness to optimize the reduced model selection provided that the user has a precise idea on the trend of the full solution u . Figure 7 provides an example in such a direction.

4.2 An a Posteriori Approach

The main purpose of an *a posteriori* technique is to devise an automatic procedure to select the modal distribution in Ω. This automatic choice is performed via a specific theoretical tool, represented by an *a posteriori* modeling error estimator. Following [23], we focus on the piecewise Hi-Mod approach. Moreover, we assume that the finite element partition is sufficiently fine to neglect the discretization error.

According to a goal-oriented analysis (see, e.g., [5, 14, 16, 20]), we measure the accuracy of the reduced model via a goal functional $J : H^1(\Omega, \mathscr{T}_\Omega) \to \mathbf{IR}$ which represents a physical quantity of interest to be measured (e.g., mean or local values, the lift and drag coefficients around bodies in external flows, convective or diffusive fluxes). In particular, we assume J linear. We estimate the unknown value $J(u)$ via the computable value $J(u_{\mathbf{m}})$, with u and $u_{\mathbf{m}}$ solution to (2) and (11), respectively, with the purpose of keeping the goal error $J(u - u_{\mathbf{m}})$ below a desired tolerance. The automatic procedure aims at identifying the subdomains Ω_i and the associated modal multi-index $\mathbf{m}$ to guarantee such a goal.

To estimate the goal error, we define the *piecewise Hi-Mod dual problem*: given a modal multi-index $\mathbf{m} \in [\mathbb{N}^+]^s$,

$$\text{find } z_{\mathbf{m}} \in V_{\mathbf{m}}(\mathscr{T}_\Omega) \quad : \quad a_{\mathscr{T}_\Omega}(v_{\mathbf{m}}, z_{\mathbf{m}}) = J(v_{\mathbf{m}}) \quad \forall v_{\mathbf{m}} \in V_{\mathbf{m}}(\mathscr{T}_\Omega), \tag{18}$$

where $J(\cdot) = \sum_{i=1}^{s} J_i(\cdot)$ with $J_i = J|_{\Omega_i}$ for $i = 1, \ldots, s$. Moreover, we introduce an auxiliary hierarchically reduced space $V_{\mathbf{m}^+}(\mathscr{T}_\Omega)$, defined exactly as $V_{\mathbf{m}}(\mathscr{T}_\Omega)$ with $\mathbf{m}^+ \in [\mathbb{N}^+]^s$ such that $\mathbf{m}^+ > \mathbf{m}$ (i.e., $m_i^+ > m_i$, for any $i = 1, \ldots, s$). The inclusions $V_{\mathbf{m}}(\mathscr{T}_\Omega) \subset V_{\mathbf{m}^+}(\mathscr{T}_\Omega) \subset H^1(\Omega, \mathscr{T}_\Omega)$ trivially hold, while we refer to space $V_{\mathbf{m}^+}(\mathscr{T}_\Omega)$ as the *enriched* piecewise hierarchically reduced space. Finally, we associate with $V_{\mathbf{m}^+}(\mathscr{T}_\Omega)$ the *enriched piecewise Hi-Mod primal and dual formulations*: given a modal multi-index $\mathbf{m}^+ \in [\mathbb{N}^+]^s$,

$$\text{find } u_{\mathbf{m}^+} \in V_{\mathbf{m}^+}(\mathscr{T}_\Omega) \ : \ a_{\mathscr{T}_\Omega}(u_{\mathbf{m}^+}, v_{\mathbf{m}^+}) = \mathscr{F}_{\mathscr{T}_\Omega}(v_{\mathbf{m}^+}) \ \forall v_{\mathbf{m}^+} \in V_{\mathbf{m}^+}(\mathscr{T}_\Omega), \tag{19}$$

and

$$\text{find } z_{\mathbf{m}^+} \in V_{\mathbf{m}^+}(\mathscr{T}_\Omega) \ : \ a_{\mathscr{T}_\Omega}(v_{\mathbf{m}^+}, z_{\mathbf{m}^+}) = J(v_{\mathbf{m}^+}) \ \forall v_{\mathbf{m}^+} \in V_{\mathbf{m}^+}(\mathscr{T}_\Omega). \tag{20}$$

We recall now the main proposition in [23] which represents the reference result in the proposal of the *a posteriori* modeling error estimator:

Proposition 1 *Let us assume that there exists a positive constant $\beta_{\mathbf{m}} < 1$ and a modal multi-index $\mathbf{M}_0 \in [\mathbb{N}^+]^s$, such that, for $\mathbf{m}, \mathbf{m}^+ \in [\mathbb{N}^+]^s$ with $\mathbf{m}^+ > \mathbf{m} \geq \mathbf{M}_0$,*

$$|J(e_{\mathbf{m}^+})| \leq \beta_{\mathbf{m}} \, |J(e_{\mathbf{m}})|, \tag{21}$$

with $e_{\mathbf{m}} = u - u_{\mathbf{m}}$, $e_{\mathbf{m}^+} = u - u_{\mathbf{m}^+} \in H^1(\Omega, \mathscr{T}_\Omega)$ the piecewise modeling error associated with formulation (11) and (19), respectively. Then,

$$\frac{|J(u_{\mathbf{m}^+} - u_{\mathbf{m}})|}{1 + \beta_{\mathbf{m}}} \leq |J(e_{\mathbf{m}})| \leq \frac{|J(u_{\mathbf{m}^+} - u_{\mathbf{m}})|}{1 - \beta_{\mathbf{m}}}. \tag{22}$$

Moreover, the following identity holds

$$J(u_{\mathbf{m}^+} - u_{\mathbf{m}}) = a_{\mathscr{T}_\Omega}(u_{\mathbf{m}^+} - u_{\mathbf{m}}, z_{\mathbf{m}^+} - z_{\mathbf{m}}). \tag{23}$$

Relation (23), combined with the two-sided bound (22), leads to identify the *a posteriori* modeling error estimator for the goal error $|J(e_{\mathbf{m}})|$ with the quantity

$$\eta_{\mathbf{m}\mathbf{m}^+} = |a_{\mathscr{T}_\Omega}(u_{\mathbf{m}^+} - u_{\mathbf{m}}, z_{\mathbf{m}^+} - z_{\mathbf{m}})|. \tag{24}$$

The lower and the upper bound in (22) represent the corresponding efficiency and reliability estimate, respectively. $\eta_{\mathbf{m}\mathbf{m}^+}$ is a goal-oriented hierarchical estimator which combines the easy computability typical of a hierarchical estimator with the high versatility proper of a goal functional analysis.

From a computational viewpoint, to evaluate $\eta_{\mathbf{m}\mathbf{m}^+}$ we replace the piecewise hierarchically reduced primal and dual solutions with the corresponding discrete approximations. Thus, via (24), we simply evaluate the quantity $(z^h_{\mathbf{m}^+} - z^h_{\mathbf{m}})^T K (u^h_{\mathbf{m}^+} - u^h_{\mathbf{m}})$, where K is the stiffness matrix associated with the enriched formulation (19), which is already available and does not need any additional assembly. Alternative procedures to evaluate estimator $\eta_{\mathbf{m}\mathbf{m}^+}$ are proposed in [23].

Remark 5 Hypothesis (21) represents a *saturation assumption*. It is rather usual in the context of a hierarchical *a posteriori* analysis for the discretization error associated with a finite element approximation [1, 6, 10]. Requirement (21) essentially generalizes a standard saturation hypothesis to the context of a modeling error analysis, by including the functional used to measure the error. A numerical check of such a hypothesis is provided in [23, Sect. 3.3].

4.2.1 The Modeling Adaptive Procedure

The adaptive procedure proposed in [23] consists of two phases. The first phase identifies the number s and the location of the subdomains Ω_i. In the second phase we find the modal multi-index $\mathbf{m}$ to satisfy the requirement $\eta_{\mathbf{m}\mathbf{m}^+} \leq \texttt{TOL}$, with $\texttt{TOL}$ a user-defined tolerance.

Let us focus on the first phase. To select the subdomains Ω_i, we employ a thresholding technique (see Fig. 10, left). We compute the estimator $\eta_{\mathbf{m}\mathbf{m}^+}$ in a uniform context, i.e., by employing $\widetilde{m}$ and $\widetilde{m}^+$ modes on the whole Ω, with $\widetilde{m}^+ > \widetilde{m}$. We usually choose small values for both $\widetilde{m}$ and $\widetilde{m}^+$ to contain the computational costs. In particular, we compute the estimator normalized by its maximum value, $\check{\eta}_{\widetilde{m}\widetilde{m}^+}$, so that we may assume the estimate to be in the range $[\check{\eta}^{\min}_{\widetilde{m}\widetilde{m}^+}, 1]$, with $\check{\eta}^{\min}_{\widetilde{m}\widetilde{m}^+}$ the minimum value assumed by the normalized estimator on Ω. Now, we pick a threshold $\phi \in (0, 1)$. Then, after introducing a uniform finite element partition $\{K_l\}$ on Ω_{1D}, we assign the value $\check{\eta}_{\widetilde{m}\widetilde{m}^+}\big|_{K_l}$ to the barycenter of K_l. We denote by ζ_j, with $j = 1, \ldots, \check{s}$, the intersections between $\check{\eta}_{\widetilde{m}\widetilde{m}^+}$ and ϕ, and by σ_j the corresponding closest finite element node. The set $\{\sigma_j\}$ identifies the partition $\mathscr{T}_\Omega = \{\Omega_i\}_{i=1}^s$, with $s = \check{s} + 1$, $\sigma_0 = x_0$, $\sigma_s = x_1$, and with $\Sigma_j = \{\sigma_j\} \times \gamma_{\sigma_j}$ the interface between Ω_j and Ω_{j+1}. We highlight that subdomains Ω_i are fixed once and for all at the end of this phase and do not change anymore during the adaptive procedure.

As byproduct of the first phase, we also get the initial guess $\mathbf{m}^{(0)} = \{m_i^{(0)}\}_{i=1}^s \in [\mathbb{N}^+]^s$ for the modal multi-index to start the second phase of the modeling adaptive procedure. We set

$$m_i^{(0)} = \begin{cases} \widetilde{m} & \text{if } \check{\eta}_{\widetilde{m}\widetilde{m}^+}\big|_{K_l} < \phi, \ \forall K_l \text{ s.t. } K_l \cap \Omega_{1D,i} \neq \emptyset, \\ \widetilde{m} + \delta & \text{if } \check{\eta}_{\widetilde{m}\widetilde{m}^+}\big|_{K_l} \geq \phi, \ \forall K_l \text{ s.t. } K_l \cap \Omega_{1D,i} \neq \emptyset, \end{cases} \tag{25}$$

where $\delta \in \mathbb{N}^+$ denotes the modal update. We define the initial guess $\mathbf{m}^{+\,(0)}$ for the multi-index $\mathbf{m}^+$ in an analogous way.

Remark 6 Some crucial situations may occur when selecting the threshold ϕ. If ϕ is less than $\check{\eta}^{\min}_{\widetilde{m}\widetilde{m}^+}$, it means that the initial guess for the modal truncation is too coarse. We consequently refine both $\widetilde{m}$ and $\widetilde{m}^+$ and recompute the normalized estimator. The algorithm above fails even when a root of the equation $\phi - \check{\eta}_{\widetilde{m}\widetilde{m}^+} = 0$ has multiplicity strictly greater than one. A possibility to avoid this is to check also the first derivative of $\check{\eta}_{\widetilde{m}\widetilde{m}^+}$ before selecting ϕ. Finally, if $\check{\eta}_{\widetilde{m}\widetilde{m}^+}$ exhibits several oscillations around a range of values, these values are not eligible as threshold, since this would lead to identify too many subdomains, making the numerical procedure completely ineffective. We refer the interested reader to Remark 4 in [23] to get more details and some example for such critical situations.

We move now to the second phase of the modeling adaptive procedure. Starting from the initial guesses $\mathbf{m}^{(0)}$ and $\mathbf{m}^{+\,(0)}$ defined according to (25), we apply a standard equidistribution criterion to the subdomains Ω_i, i.e, we demand that $\eta^i_{\mathbf{mm}^+} = \mathtt{TOL}/s$, where $\eta^i_{\mathbf{mm}^+} = \eta_{\mathbf{mm}^+}|_{\Omega_i}$ denotes the modeling error estimator associated with the subdomain Ω_i for $i = 1, \ldots, s$. In particular, to make the adaptive procedure more efficient, we include both the model refinement and coarsening. The main steps of the mode adaptivity are summarized in the following pseudocode:

```
1. set $\mathbf{m} = \mathbf{m}^{(0)}$ and $\mathbf{m}^+ = \mathbf{m}^{+\,(0)}$;
2. compute $\eta_{\mathbf{mm}^+}$;
3. while ($\eta_{\mathbf{mm}^+} > \mathtt{TOL}$ & k $\leq$ Nmax) {
   4. for i=1:s
      5. compute $\eta^{\mathrm{i}}_{\mathbf{mm}^+}$;
      6. if $\eta^{\mathrm{i}}_{\mathbf{mm}^+} >$ delta1 $\frac{\mathrm{TOL}}{\mathrm{s}}$
         7. $m_{\mathrm{i}}^{(\mathrm{k})} = m_{\mathrm{i}}^{(\mathrm{k}-1)} + \delta$; $m_{\mathrm{i}}^{+\,(\mathrm{k})} = m_{\mathrm{i}}^{+\,(\mathrm{k}-1)} + \delta$;
      8. elseif $\eta^{\mathrm{i}}_{\mathbf{mm}^+} \leq$ delta2 $\frac{\mathrm{TOL}}{\mathrm{s}}$
         9. $m_{\mathrm{i}}^{(\mathrm{k})} = \max(1, m_{\mathrm{i}}^{(\mathrm{k}-1)} - \delta)$, $m_{\mathrm{i}}^{+,(\mathrm{k})} = \max(1, m_{\mathrm{i}}^{+,(\mathrm{k}-1)} - \delta)$;
         end
     end
   10. compute $\eta_{\mathbf{mm}^+}$;
   11. k = k + 1; }
```

Some remarks are in order. We introduce a maximum number `Nmax` of allowed iterations to ensure that the procedure stops. If the algorithm terminates within this number, the modal multi-index predicted by the procedure identifies a piecewise hierarchically reduced solution $u_{\mathbf{m}}$ such that $J(u_{\mathbf{m}}) \simeq J(u)$ within tolerance `TOL`.

The parameters `delta1` and `delta2` limit the occurrence of model refinement and coarsening, respectively by improving the algorithm robustness as well as the computational efficiency. In particular, during the model coarsening, we force the minimal value of admissible modes to be at least equal to one.

Finally, the control of the fulfillment of the desired tolerance for the estimator is performed twice: at step `3.`, when we check if the initial guess $\mathbf{m}^{(0)}$ predicts a sufficiently rich model (in such a case, no loop is needed); then, we perform the same check after the new prediction for $\mathbf{m}$ at steps `7.` and `9.`.

4.2.2 A Numerical Example

We assess the reliability of both the modeling error estimator $\eta_{\mathbf{mm}^+}$ and of the adaptive procedure above on the same test case tackled in the previous numerical sections. In particular, as goal quantity we choose the mean value of the solution on Ω. This leads us to identify the functional J in (18) with $J(v) = [\text{meas}(\Omega)]^{-1} \int_\Omega v(x, y)\, d\mathbf{z}$. Notice that the dual problem still coincides with an advection-diffusion problem, but with a forward advective field. Full homogeneous Dirichlet boundary conditions complete the dual formulation.

We make the following choices for the input paramentes of the modeling adaptive procedure: `TOL`$= 2 \cdot 10^{-3}$, $\widetilde{m} = 3$, $\widetilde{m}^+ = 5$, $\phi = 0.1$, $\delta = 1$, `delta1` $= 0.5$, `delta2` $= 1.5$, `Nmax` $= 10$, while we introduce a uniform finite element partition of size $h = 0.05$ to discretize $\Omega_{1D} = (0, 7)$. The adaptive procedure detects the three subdomains $\Omega_1 = (0, 2.7) \times (0, 2)$, $\Omega_2 = (2.7, 4.4) \times (0, 2)$ and $\Omega_3 = (4.4, 7) \times (0, 2)$, while the initial guesses predicted for the modal multi-indices are $\mathbf{m}^{(0)} = \{3, 4, 3\}$, $\mathbf{m}^{+\,(0)} = \{5, 6, 5\}$. The domain Ω_2 associated with the two sources is immediately identified as the most troublesome.

Concerning the domain decomposition algorithm, due to the advective field $\mathbf{b}$, we have to pay attention in selecting the Dirichlet and the Neumann interfaces for the primal and dual problems to guarantee the well-posedness of each subproblem on Ω_i. In particular, a Dirichlet/Neumann condition is assigned at $\Sigma_1 = \{2.7\} \times \gamma_{2.7}$ and $\Sigma_2 = \{4.4\} \times \gamma_{4.4}$ when solving the primal problem; conversely, a Neumann/Dirichlet condition is enforced on Σ_1 and Σ_2 to solve the dual problem. We fix a tolerance equal to 10^{-2} for the domain decomposition algorithm at both Σ_1 and Σ_2. The average number of iterations demanded to ensure this accuracy is eight for the primal problem and nine for the dual one.

Three model adaptive iterations allow to reach the desired tolerance `TOL`, with a final prediction for the modal multi-index $\mathbf{m} = \mathbf{m}^{(3)} = \{3, 7, 1\}$. While the initial number $m_1^{(0)} = 3$ of modes is preserved on Ω_1, we have a gradual increase of the number of modal functions in Ω_2 and a model coarsening occurs in Ω_3. Table 1

Table 1 Quantitative information on the second phase of the modeling adaptive procedure

k	$\mathbf{m}$	$\mathbf{m}^+$	$\eta_{\mathbf{m}\mathbf{m}^+}$
0	$\{3, 4, 3\}$	$\{5, 6, 5\}$	$1.81 \cdot 10^{-2}$
1	$\{3, 5, 2\}$	$\{5, 7, 4\}$	$4.72 \cdot 10^{-3}$
2	$\{3, 6, 1\}$	$\{5, 8, 3\}$	$4.56 \cdot 10^{-3}$
3	$\{3, 7, 1\}$	$\{5, 9, 3\}$	$1.31 \cdot 10^{-3}$

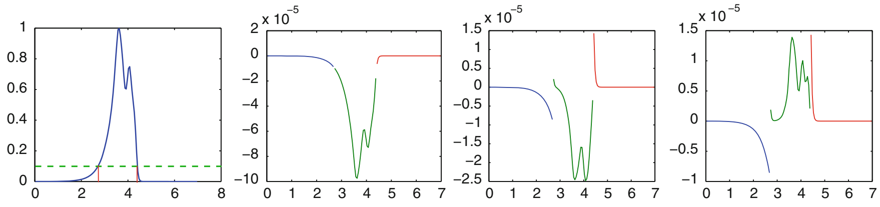


Fig. 10 Normalized estimator $\check{\eta}_{\widetilde{m}\widetilde{m}^+}$; distribution of the the local quantities $J_i(u_{\mathbf{m}^+} - u_{\mathbf{m}})$, with $i = 1, 2, 3$, for the starting guess and for the odd iterations of the adaptive procedure (*left–right*)

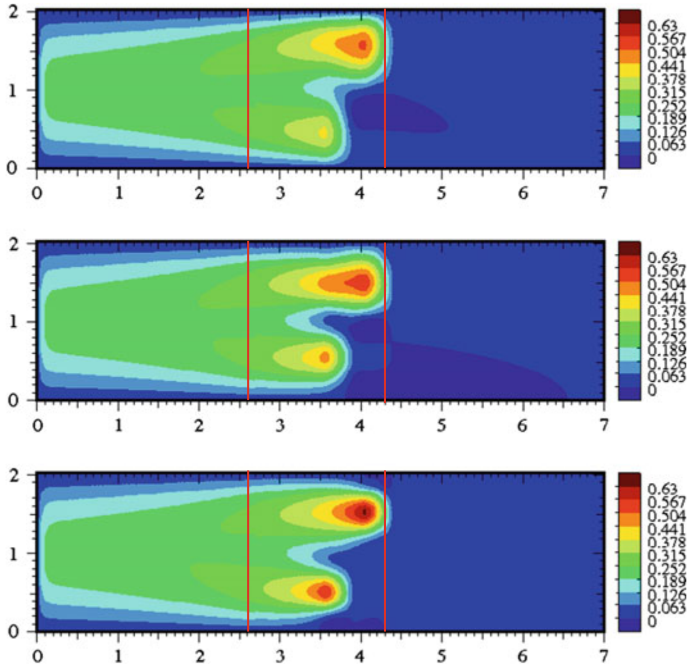


Fig. 11 Piecewise Hi-Mod solutions: $u^h_{3,4,3}$, $u^h_{3,5,2}$, $u^h_{3,7,1}$ (*top–bottom*)

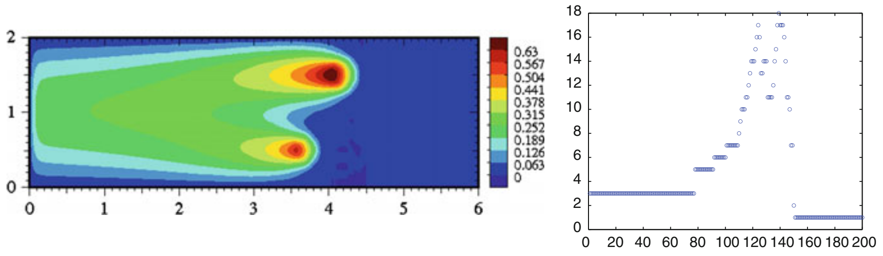


Fig. 12 Pointwise Hi-Mod solution (*left*); corresponding modal distribution (*right*)

provides more quantitative information about the adaptive procedure. The sequence of columns gathers the iteration number k (k= 0 refers to the initial configuration predicted by the first phase), the modal multi-indices $\mathbf{m}$, $\mathbf{m}^+$ and the value of the estimator $\eta_{\mathbf{mm}^+}$. For the sake of simplicity, we set the saturation constant $\beta_{\mathbf{m}}$ in (21) to zero. Figure 11 collects the piecewise Hi-Mod solutions for the starting guess and for the odd iterations of the adaptive procedure, while Fig. 10 shows the associated error estimators: in particular, the three lines correspond to the local quantities $J_i(u_{\mathbf{m}^+} - u_{\mathbf{m}})$, for $i = 1, 2, 3$. The model discontinuity occurring at the interface Σ_1 is almost imperceptible.

Remark 7 The *a posteriori* modeling error analysis and the adaptive procedure here presented are generalizable to both a uniform and a pointwise setting. Figure 12 shows the Hi-Mod solution predicted by the error estimator derived for a pointwise reduction to control the energy norm of the modeling error [25]. Notice the high number of modes used in the central part of the domain. Moreover, in [23] the model adaptivity is successfully combined with an adaptive prediction of the finite element partition along Ω_{1D}.

5 Conclusions and Perspectives

We have presented three different approaches to model phenomena characterized by strong horizontal dynamics, even though in the presence of meaningful transverse dynamics. The pointwise Hi-Mod procedure seems to be the most flexible approach since it is suited to deal with both localized and widespread dynamics. On the contrary the uniform approach better deals with diffused dynamics while the piecewise approach is ideal to manage confined transverse dynamics.

Concerning the future developments, our last goal is to employ Hi-Mod reduction for the simulation of the blood flow in the whole arterial system. Mandatory next steps will be the generalization of the Hi-Mod procedure to unsteady nonlinear problems, such as the Navier-Stokes equations, as well as the combination of the Hi-Mod procedure with approaches already validated in a haemodynamics setting, such as the geometrical multiscale formulation [7, 12, 13].

Acknowledgments The author thanks Massimiliano Lupo Pasini for Figs. 7 and 12 and Alessandro Veneziani for the suggestions during the preparation of the manuscript.

References

1. Achchab B, Achchab S, Agouzal A (2004) Some remarks about the hierarchical a posteriori error estimate. Numer Meth Partial Differ Equ 20(6):919–932
2. Ainsworth M (1998) A posteriori error estimation for fully discrete hierarchic models of elliptic boundary value problems on thin domains. Numer Math 80:325–362
3. Aletti M, Perotto S, Veneziani A (2014) Educated bases for hierarchical model reduction in 2D and 3D. (in preparation)
4. Babuška I, Schwab C (1996) A posteriori error estimation for hierarchic models of elliptic boundary value problems on thin domains. SIAM J Numer Anal 33:221–246
5. Bangerth W, Rannacher R (2003) Adaptive finite element methods for differential equations. Birkhauser, Basel
6. Bank RE, Smith RK (1993) A posteriori error estimates based on hierarchical bases. SIAM J Numer Anal 30:921–935
7. Blanco PJ, Leiva JS, Feijóo RA, Buscaglia GC (2011) Black-box decomposition approach for computational hemodynamics: one-dimensional models. Comput Methods Appl Mech Eng 200(13–16):1389–1405
8. Canuto C, Maday Y, Quarteroni A (1982) Analysis of the combined finite element and Fourier interpolation. Numer Math 39:205–220
9. Canuto C, Maday Y, Quarteroni A (1984) Combined finite element and spectral approximation of the Navier-Stokes equations. Numer Math 44:201–217
10. Dörfler W, Nochetto RH (2002) Small data oscillation implies the saturation assumption. Numer Math 91:1–12
11. Ern A, Perotto S, Veneziani A (2008) Hierarchical model reduction for advection-diffusion-reaction problems. In: Kunisch K, Of G, Steinbach O (eds) Numerical mathematics and advanced applications, pp 703–710. Springer, Heidelberg
12. Formaggia L, Nobile F, Quarteroni A, Veneziani A (1999) Multiscale modelling of the circulatory system: a preliminary analysis. Comput Visual Sci 2:75–83
13. Formaggia L, Quarteroni A, Veneziani A (eds) (2009) Cardiovascular mathematics, modeling, simulation and applications, vol 1. Springer, Berlin
14. Giles MB, Süli E (2002) Adjoint methods for pdes: a posteriori error analysis and postprocessing by duality. Acta Numer 11:145–236
15. Heinrich B (1996) The Fourier-finite-element method for Poisson's equation in axisymmetric domains with edges. SIAM J Numer Anal 33:1885–1911
16. Johnson C (1993) A new paradigm for adaptive finite element methods. In: Whiteman J (eds) Proceedings of MAFELAP, vol 93. Wiley, New York
17. Lasis A, Süli E (2003) Poincaré-type inequalities for broken Sobolev spaces. Technical Report 03–10. Oxford University Computing Laboratory
18. Lions J-L, Magenes E (1972) Non-homogeneous boundary value problems and applications. Springer, Berlin
19. Lorenz B, Biros G, Ghattas O, Heinkenschloss M, Keyes D, Mallick B, Tenorio L, van Bloemen Waanders B, Willcox K, Marzouk Y (eds) (2011) Large-scale inverse problems and quantification of uncertainty, vol 712. Wiley, Chichester
20. Oden JT, Prudhomme S (2001) Goal-oriented error estimation and adaptivity for the finite element method. Comput Math Appl 41:735–756
21. Perotto S (2014) Hierarchical model (Hi-Mod) reduction in non-rectilinear domains. In: Erhel J, Gander M, Halpern L, Pichot G, Sassi T, Widlund O (eds) Lecture Notes in Computer Science Engineering. Springer, Berlin,pp 407–414

22. Perotto S, Ern A, Veneziani A (2010) Hierarchical local model reduction for elliptic problems: a domain decomposition approach. Multiscale Model Simul 8(4):1102–1127
23. Perotto S, Veneziani A (2014) Coupled model and grid adaptivity in hierarchical reduction of elliptic problems. J Sci Comput. doi:10.1007/s10915-013-9804-y
24. Perotto S, Zilio A (2013) Hierarchical model reduction: three different approaches. In: Cangiani A, Davidchack R, Georgoulis E, Gorban A, Levesley J, Tretyakov M (eds) Numerical mathematics and advanced applications, pp 851–859. Springer, Berlin
25. Perotto S, Zilio A (2014) Hierarchical model reduction for time dependent problems. (in preparation)
26. Quarteroni A, Valli A (1999) Domain decomposition methods for partial differential equations. In: Numerical mathematics and scientific computation. Oxford University Press, New York
27. Toselli A, Widlund O (2005) Domain decomposition methods-algorithms and theory. Springer, Berlin
28. Vogelius M, Babuška I (1981) On a dimensional reduction method I. the optimal selection of basis functions. Math Comput 37:31–46
29. Vogelius M, Babuška I (1981) On a dimensional reduction method II. some approximation-theoretic results. Math Comput 37:47–68
30. Vogelius M, Babuška I (1981) On a dimensional reduction method III. a posteriori error estimation and an adaptive approach. Math Comput 37:361–384

Part V
Multi-fluid Flows

On the Application of Two-Fluid Flows Solver to the Casting Problem

K. Kamran, R. Rossi, P. Dadvand and S. R. Idelsohn

Abstract Two-fluid modeling of the casting process is important as it particularly provides insight to the air behavior during the filling process. Large deformations of the material-air front require an interface capturing technique to detect it on the fixed Eulerian meshes. On the other hand, if sharp interface is considered, jumps in the material properties along one single element causes severe instabilities in the solution. We review various techniques developed for both conservative level set method to capture the interface, and enrichment techniques to cure the instabilities. The coupling of the level set method with one of these enrichment techniques is applied to the mold filling process.

1 Introduction

Multi-fluid flow simulation with large deformations at the interface has to deal with two main challenges. The first one is accurately follow or capture the interface between the phases and the second one is treating jumps in the material properties

K. Kamran (✉) · R. Rossi · P. Dadvand · S. R. Idelsohn
Centre Internacional de Métodes Numérics en Enginyeria (CIMNE),
Gran Capitán s/n,
08034 Barcelona, Spain
e-mail: kazem@cimne.upc.edu

R. Rossi · P. Dadvand
Universitat Politècnica de Catalunya, Barcelona, Spain
e-mail: rrossi@cimne.upc.edu

P. Dadvand
e-mail: pooyan@cimne.upc.edu

S. R. Idelsohn
Instituciò Catalana de Recerca i Estudis Avançats (ICREA), Barcelona, Spain
e-mail: sergio@cimne.upc.edu

at the interface, that result in kink or jumps in the unknown fields. The most popular technique to capture the interface is the level set method that can deal with the large deformations of the interface. its main deficiency is the gain/loss of the material at sharp interfaces or during the reinitialization step. On the other hand, for applications like mold filling that one of the fluids has large density, aluminum or steel, and the other one small density, air, instabilities appear at the interface that can totally destroy the solution accuracy. The main reason for such behavior is the poor quality of the linear elements to capture kink or jumps in the pressure-velocity pair. Various *enrichment* techniques are provided to improve the approximation properties at the elements cut by the interface. In the following, we first review some main algorithms developed for the interface capturing approach by means of level set method. Then, various enrichment techniques to treat kink and/or jumps in the unknown fields at the interface are presented. Finally we loosely couple the level set method and one of the enrichment techniques to model the mold filling process.

2 Level Set Method

The underlying idea behind level set method is to represent an interface as the zero-level set of a higher dimensional function $\phi(x,t)$. This function is scalar and substantially reduces the complexity of describing the interface, especially when undergoing topological changes such as pinching and merging. The level set function $\phi(x,t)$ is defined to be a smooth function that is positive in one region and negative in the other.

The motion of the interface is determined by a velocity field, **u**, which can depend on a variety of things including position, time, geometry of the interface, or be given externally for instance as the solution of the Navier-Stokes equations in a fluid flow simulation. The advection equation for interface evolution is,

$$\phi_t + \mathbf{u} \cdot \nabla\phi = 0. \tag{1}$$

This level set equation is a first order hyperbolic PDE and only needs to be solved near the interface. Geometrical quantities related to the interface ($\phi = 0.0$), i.e. unit normal **N** and curvature κ, can be calculated from the level set function by:

$$\mathbf{N} = \frac{\nabla\phi}{\parallel \nabla\phi \parallel} \quad \kappa = \nabla \cdot \mathbf{N} \tag{2}$$

The most common choice for the level set function ϕ is the signed distance to the interface so that $|\nabla\phi| = 1$. This ensures that the level set is a smoothly varying function well suited for high order accurate numerical methods. There are several techniques to solve the level set Eq. (1) in space and time. One of the most popular on structured meshes is the high order Hamilton-Jacobi ENO method (HJ-ENO) combined with a Runge-Kutta method [1, 2]. Despite the high order temporal and spatial approximations of the level set equation, instabilities may appear when the level set cease to be a signed distance function. This situation occurs at the presence of large topological changes at the interface vicinity, which are quite common in practical problems. One solution is to reshape the level set function to a distance function. This method, called *reinitialization*, has been shown to stabilize those numerical instabilities. Reinitialization algorithms maintain the signed distance property by solving to steady state (as fictitious time $\tau \to \infty$) the equation

$$\phi_\tau + sgn(\phi_0)(\| \nabla\phi \| - 1) = 0. \tag{3}$$

where $sgn(\phi_0)$ is a one-dimensional smeared out signum function [3]. Equation (3) only needs to be solved near the interface and not in the whole domain. Efficient ways to solve this equation to steady state via fast marching methods are discussed in [4]. This equation can also be written in a classical Hamilton-Jacobi form as:

$$\phi_\tau + \mathbf{v}(\phi) \cdot \nabla\phi = sgn(\phi_0) \quad and \quad \mathbf{v}(\phi) = sgn(\phi_0)\frac{\nabla\phi}{\| \nabla\phi \|}$$

Unfortunately, one of the major drawbacks of the reinitialization process is the difficulty in preserving the original location of the interface, often leading to breakdown in the conservation of mass. To overcome this problem of mass loss with the level set method, various solutions have been proposed:

- Particle level set methods
- Volume of fluid methods
- Geometric mass-preserving redistance methods
- Discontinuous Galerkin level set methods.

2.1 Particle Level Set

Particle Level Set (PLS) [5] uses Lagrangian marker particles to rebuild the level set in regions which are under-resolved. This is often the case for flows undergoing stretching and tearing. Two sets of massless marker particles are placed near the interface with one set, the *positive* particles, in the $\phi > 0$ region and the other set, the *negative* particles in the $\phi < 0$ region. It is unnecessary to place particles far from the interface and this greatly reduces the number of particles needed in a given simulation. The region near the interface could be considered as the region covered by all elements that has at least one corner with the distance inferior to 3 element size. The number of particle per cell is set to 4^d where d is the spatial dimension. Figure 1a shows the zero level-set for the 2D Zalesak's disk after one revolution. Placement of the massless particles around the zero-level set for this test can be seen in Fig. 1b. In Fig. 1c the PLS solution after one revolution is shown.

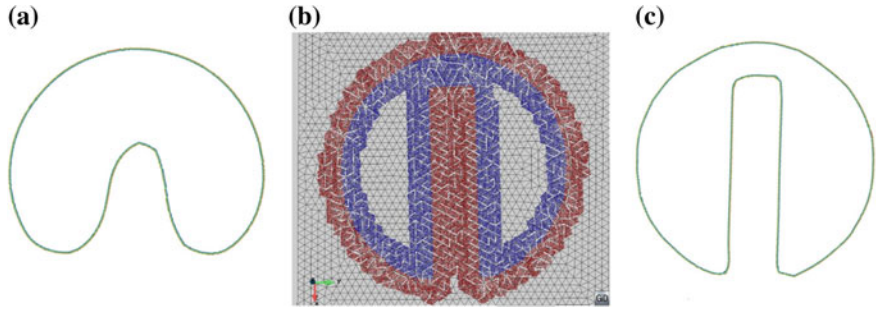


Fig. 1 Particle Level Set (PLS) method [5], **a** mass gain/loss of the standard level set solution after one revolution, **b** placement of massless *positive* (*blue*) and *negative* (*red*) particles around the interface for the Zalesak test, **c** PLS solution after one revolution

For the purpose of interface reconstruction a radius r_p is assigned to each particle as the function of its distance to the interface. This radius is bounded (on structured grids) by minimum and maximum values based upon the gird spacing ($0.1d < r_p < 0.5d$, d $= \min\{\Delta x, \Delta y, \Delta z\}$). The particles are then advected with the evolution equation

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{u}(\mathbf{x}_p), \tag{4}$$

where $\mathbf{x}_p$ is the position of the particle and $\mathbf{u}(\mathbf{x}_p)$ is its velocity. The particle velocities are interpolated from the velocities on the underlying grid. The marker particles (4) and the level set Eq. (1) are separately integrated forward in time. After each complete time cycle, the particles are used to locate possible errors in the level set function. Particles that are on the wrong side of the interface by more than their radius, as determined by the interpolated distance $\phi(\mathbf{x}_p)$, are considered to have escaped.

A local level set function is defined for each particle by means of the radius associated to the particle,

$$\phi_p(\mathbf{x}) = s_p(r_p - \parallel \mathbf{x} - \mathbf{x}_p \parallel) \tag{5}$$

where s_p is the sign of the particle, i.e. ± 1. These level sets are only defined locally on the corners of the cell containing the particle and can be seen as the particle predictions of the values of the level set function on the corners of the cell. Any variation of ϕ from ϕ_p indicates possible errors in the level set solution. The escaped positive particles are used to rebuild the $\phi > 0$ region and the escaped negative particles to rebuild the $\phi \leq 0$ region. For example take the $\phi > 0$ region and an escaped positive particle. Using Eq. (5), the ϕ_p values of the grid points on the boundary of the cell containing the particle are calculated. Each ϕ_p is compared to the local values of ϕ and the maximum of these two values is taken as ϕ^+. This is done for all escaped positive particles creating a reduced error representation of the $\phi > 0$ region. That is, given a level set ϕ and a set of escaped positive particles E^+, we initialize ϕ^+ with ϕ and then calculate

$$\phi^+ = \max_{\forall p \in E^+} (\phi_p, \phi^+)$$

Similarly for the negative region $\phi \leq 0$, we initialize ϕ^- with ϕ and then calculate

$$\phi^- = \min_{\forall p \in E^-} (\phi_p, \phi^-).$$

We merge ϕ^+ and ϕ^- back into a single level set value by setting ϕ to the value of ϕ^+ or ϕ^- which is least in magnitude at each grid point.

The Particle level set method relies on ϕ being a signed distance function. This implies that after each time marching step, a reinitialization of the level set function using Eq. (3) is necessary. Unfortunately, the reinitialization may cause the zero level set to move, which is not desirable, so the particle level set method is employed to correct these errors as well. During the reinitialization step the particles are not moved to keep the zero level set, and then any error in the reinitialization scheme is corrected by the particles. After reinitialization and correction step the radii of the particles are adjusted according to the current $\phi(\mathbf{x}_p)$ regarding their distance to the zero level set.

In summary the order of operation in PLS is: evolve both particles and level set function in time, correct errors in the level set function using particles, reinitialize the distance function, again correct the errors using particles and finally adjust the particle radii. A final task to complete the PLS is the particle reseeding, i.e. in flows with the the interface stretching and tearing we need to periodically readopt the particle distribution to the deformed interface [5].

2.2 Volume of Fluid

Coupled level set/volume-of-fluid (CLSVOF) method combines some of the advantages of the volume-of-fluid method with the level set method to obtain a method which is generally superior to either method alone. In the VOF [6, 7], the volume fraction $F(\Omega, t)$ is used to represent the free surface. Typically, Ω represents a computational cell and if $F(\Omega, t) = 1$, then the region Ω is all liquid, if $F(\Omega, t) = 0$ then the region Ω is all gas and for the intermediate values, Ω contains both gas and liquid.

One can define the volume fraction function $F(\Omega, t)$ in terms of the level set function ϕ. Since we have $\phi > 0$ in one domain (liquid) and $\phi < 0$ in the other domain (air), one can define $F(\Omega, t)$ as

$$F(\Omega, t) = \frac{1}{|\Omega|} \int_{\Omega} H(\phi(\mathbf{x}, t))\, d\Omega, \tag{6}$$

where H is the Heaviside function,

$$H(\phi) = \begin{cases} 1 & \text{if } \phi > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

The main advantage of representing the free surface with the volume fractions is that one can write accurate algorithms for advecting the volume fraction function so that mass is conserved while still maintaining a sharp representation of the interface. However, a disadvantage of the VOF method is the fact that it is difficult to compute accurate local curvatures from volume fractions. This is mainly due to the sharp transitions of the volume fractions at the interface. One remedy is to smooth the H function at the interface. If one smooths too much then the numerical method will not detect changes in curvature along the interface. In the CLSVOF method, the curvature is not smoothed at all; instead the curvature is obtained via finite differences of the level set function which in turn is derived from the level set function and volume-of-fluid function at the previous time step.

The equation governing the evolution of the VOF is:

$$F_t + \mathbf{u} \cdot \nabla F = 0. \tag{8}$$

The coupling between the level set function ϕ and the volume-of-fluid F occurs when the interface normals computed from the level set are used in the interface reconstruction at each cell that in turn are used by the VOF function. Much of the research on VOF methods has focused on obtaining higher accuracy and better representations of the interface geometry. The original piecewise linear interface reconstruction technique (PLIC) [20] has been improved upon using parabolic (PROST) [21] and least squares [22] techniques.

The CLSVOF can be summarized as, first update level set function ϕ^{n+1}, then reconstruct the interface at each cell by one of the linear or higher order techniques and finally update the volume-of-fluid function F^{n+1}. Note that reinitializing the ϕ^{n+1} as the exact signed distance function to the reconstructed interface, is another point where coupling between the LS and VOF occurs. Remind that numerically, the smoothed Heaviside function $H_\epsilon(\phi)$ is substituted for the sharp Heaviside function $H(\phi)$. The smoothed Heaviside function is defined as,

$$H_\epsilon(\phi) = \begin{cases} 1 & \text{if } \phi < -\epsilon \\ \frac{1}{2}[1 + \frac{\phi}{\epsilon} + \frac{1}{\pi}\sin(\pi\phi/\epsilon)] & \text{if } |\phi| \leq \epsilon \\ 0 & \text{if } \phi > \epsilon. \end{cases} \tag{9}$$

2.3 Geometric Mass-Preserving Redistance Methods

The advection distorts the initial shape of the level set function, which needs to be reinitialized to a smooth function preserving the position of the zero-level set.

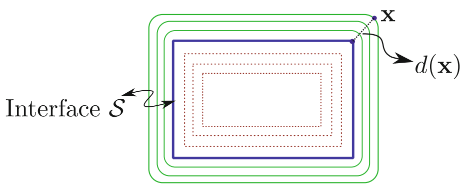


Fig. 2 Contours of the distance function d to a rectangle. Example showing that the distance to the interface from outside of the rectangle region (*solid line* contours) does not belong to the finite element space [9]

Efficient algorithms for level-set redistancing on Cartesian meshes have been developed but few methods are available for unstructured meshes. Geometric mass-preserving redistance method [8, 9], developed for unstructured meshes, can be localized on a narrow band close to the interface, saving computing effort. Almost all redistancing algorithms involves some sort of mass-rebalancing step. Geometric mass-preserving redistance includes one such step that is local and involves no adjustable parameter.

Consider an arbitrary triangulation of the domain Ω, and the associated space V_h of continuous function that are linear inside each simplex. Let $\phi_h \in V_h$ be a function, and let $\mathcal{S}$ be its zero-level set. We look for a function $\bar{\phi}_h \in V_h$ which approximates the signed distance function d to $\mathcal{S}$. This function satisfies $\|\nabla d\| = 1$ almost everywhere in Ω but does *not*, in general, belong to V_h (see Fig. 2).

Let P be the set of nodal points that are adjacent to the zero-level set of ϕ, in the sense that they are vertices of simplicis inside which ϕ_h changes sign. If one makes the simple assignment $\bar{\phi}_h(\mathbf{X}) = d(\mathbf{X})$ for all $\mathbf{X} \in P$, there is a volume loss (or gain) which could render the algorithm for practical simulations. The values of $\bar{\phi}_h$ at nodes adjacent to the zero-level set must thus be "adjusted" so as to preserve volume, and the function $\bar{\phi}_h$ must be calculated at the remaining nodes using the adjusted values at P.

Let us define $\mathcal{K}(\phi_h)$ as the set of simplices in which ϕ_h changes sign. The objective is thus to calculate $\bar{\phi}_h$ such that it approximates the signed distance d while at the same time preserving the volume,

$$V(\phi_h) = \int\limits_{\mathcal{K}(\phi_h)} H(\phi_h(\mathbf{x}))\, d\mathbf{x},$$

where H is the Heaviside function defined in (7). The contribution to the volume of each simplex $K \in \mathcal{K}(\phi_h)$ is

$$V_K(\phi_h) = \int\limits_K H(\phi_h(\mathbf{x}))\, d\mathbf{x}.$$

The algorithm to compute the conserved values $\bar{\phi}_h$ is as follows,

1. *Initialization*: The function $\bar{\phi}_h$ is initialized over the nodes in P (cut elements), to a first estimate $\bar{\phi}_h^0$, calculated as the true signed distance to the interface.
2. *Simplex-wise correction*: In general, the initialization ends up with a function $\bar{\phi}_h^0$ for which $V_K(\bar{\phi}_h^0) \neq V_K(\phi_h)$, though the difference is quite small. The idea here is to determine a constant value, Δ_K, to add to $\bar{\phi}_h^0$ over cut element to achieve local volume preservation. The nonlinear system to be solved is,

   $$R_K(\Delta_K) = V_K(\bar{\phi}_h^0 + \Delta_K) - V_K(\phi_h) = 0$$

   The values Δ_K are computed using a simple secant algorithm $\Delta_K^{i+1} = \Delta_K^i - R_K^i(\Delta_K^i - \Delta_K^{i-1})/(R_K^i - R_K^{i-1})$, which converges in very few steps.

3. *Node-wise correction*: From the previous step simplex-wise correction to the $\bar{\phi}_h^0$ to preserve volume on all cut elements are computed. A node-wise correction ψ_h is defined by averaging over the simplices that share a node. The value of $\bar{\phi}_h$ on P is finally calculated as

   $$\bar{\phi}_h = \bar{\phi}_h^0 + C\psi_h,$$

   where C is computed such that conserve the volume over the band of elements cut by the interface,

   $$\int_{\mathcal{K}(\phi_h)} H\left(\bar{\phi}_h^0 + C\psi_h\right) d\mathbf{x} = \int_{\mathcal{K}(\phi_h)} H\big(\phi_h(\mathbf{x})\big)\, d\mathbf{x},$$

   This nonlinear system for C is again solved by a simple secant method and converges in very few iterations.

Note that considering a uniform mass-conserving correction by choosing $\psi_h = 1$ is not optimal since volume loss/gain is not uniform over the interface. In fact the loss/gain in volume tends to concentrate in regions of higher curvature. Figure 3 shows the application of geometric mass-preserving redistance method to the Zalesak test.

2.4 Discontinuous Galerkin Level Set Methods

A quadrature-free discontinuous Galerkin method (DGM) has been developed for the conservative form of the level-set equation in [10]. This method has two main advantages that makes it interesting comparing to the other methods. First, it is not necessary to compute gradients of the level set function ϕ and second, the scheme remains stable even if ϕ diverges from a signed distance function. It follows that in this method the reinitialization step is omitted and therefore many problems of the mass conservation are disappeared.

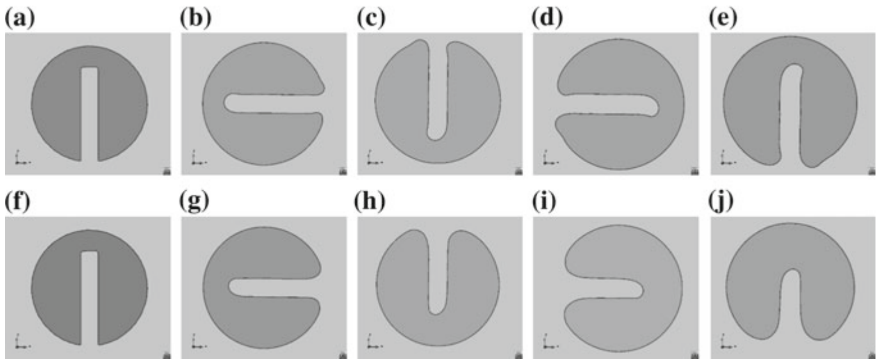


Fig. 3 Geometric mass-preserving redistance method [8] applied to the Zalesak test. Interface position at different instances during one revolution and with redistancing after each step, **a**–**e** mass-preserving redistance method and **f**–**j** level set method. **a** t = 0 s, **b** t = 157 s, **c** t = 314 s, **d** t = 471 s, **e** t = 628 s, **f** t = 0 s, **g** t = 157 s, **h** t = 314 s, **i** t = 471 s, **j** t = 628 s

The *conservative* form of the level set Eq. (1) is:

$$\phi_t + \nabla \cdot (\mathbf{u}\phi) = \phi \nabla \cdot \mathbf{u}.$$

In case of incompressible flows, $\nabla \cdot \mathbf{u} = 0$, this equation reduces to

$$\phi_t + \nabla \cdot (\mathbf{u}\phi) = 0. \tag{10}$$

Within this framework, only possible for incompressible flow, it is not needed to compute $\nabla\phi$ anymore. The variational form of the Eq. (10) is written as;

$$\int\limits_{\Omega} \phi_t w dv = \int\limits_{\Omega} \mathbf{u}\phi \cdot \nabla w dv - \int\limits_{\partial\Omega} f\, w ds \tag{11}$$

where $f(\phi) = \phi\mathbf{u}\cdot\mathbf{n}$ is the normal trace of the fluxes and $\mathbf{u}$ is the velocity field. Since the DGM allows discontinuities at the interface, the flux is not uniquely determined on it and a flux formula has to be supplied to complete the discretization process. In the simple advection case the *upwind flux* is chosen to be the value of ϕ on $\partial\Omega$:

$$\phi^{UP} = \begin{cases} \phi^{+} & \text{if } \mathbf{u}\cdot\mathbf{n} \leq 0, \quad \textit{i.e. The flow goes inside the domain} \\ \phi^{-} & \text{if } \mathbf{u}\cdot\mathbf{n} > 0, \quad \textit{i.e. The flow goes outside the domain}. \end{cases}$$

In each element ϕ is discretized using piecewise continuous approximations and an important assumption is considered to evaluate the nonlinear term $F(\phi)$;

Table 1 Different level set techniques applied to the 2D Zalesak test

Method	h	Area	% Area loss (%)	L_1 error	CPU time(s)
Exact	–	582.2	–	–	–
WENO [5]	1.0	613.0	-5.3	0.61	–
PLS [5]	1.0	580.4	0.31	0.07	134.043
PLS [5]	0.5	581.7	0.08	0.031	840.038
Geometric [8]	0.5	579.1	-0.55	0.17	–
DG TRI(4) [10]	4.0	583.27	-0.18	0.05	101.22
DG TRI(4) [10]	2.0	582.17	-0.005	0.011	955.5

$$\phi = \sum_{k=1}^{d} N_k \phi_k^e$$

$$F(\phi) = F(\sum_{k=1}^{d} N_k \phi_k^e) \simeq \sum_{k=1}^{d} N_k F(\phi_k^e) \tag{12}$$

In [10] Lagrangian shape functions are used to approximate ϕ and in each element e, d Lagrangian points is considered. As the interpolations are being disconnected, the integral form (11) is written for each element e of the mesh. Note that for elements with straight line edges the Jacobian matrix is constant and also the matrices related to (11), once written in the reference coordinates, are independent of each element. In this way the DGM is quadrature-free. In [10] a TVD-Runge-Kutta of order $p + 1$, where p is the polynomial interpolation order, is used for the time stepping. The RK of order $p + 1$ with DG at order p can be proven to be stable under the CFL condition $\Delta t < \frac{h}{c(2p+1)}$, where h is the element size and c is the norm of maximum velocity on element e. Table 1 summarizes the comparison between, DG for various order of interpolation and two other methods mentioned earlier, VOF and PLS, for the 2D Zalesak's disk.

3 Two Phase Flows

Two types of methods are generally distinguished for resolving the interface between the two phases: *interface tracking* in which the mesh explicitly represents the interface and follows its movements [11–13] and *interface capturing* in which the interface is described implicitly as the zero level-set of an auxiliary function [6, 14–16] defined on the fixed mesh. The interface-capturing method is more convenient in case that large topological changes occur at the interface. The main issue that need to be addressed in two-phase flow problems is the possible jumps or kinks in the unknown fields i.e. velocity and pressure, due to the jumps in material properties, i.e. density and viscosity, or the surface tension. In this view, two class of methods can be recognized in the interface-tracking category. In the first one [17] a numerical thickness is given to the interface by means of the smoothed Heaviside function (9). The material properties, ρ and μ, are then defined for the two-fluid system at the discrete level as,

$$\rho = \rho_w H_\epsilon(\phi^h) + \rho_a[1 - H_\epsilon(\phi^h)]$$
$$\mu = \mu_w H_\epsilon(\phi^h) + \mu_a[1 - H_\epsilon(\phi^h)]$$

A typical stabilized form of the incompressible Navier-Stokes equations are considered to complete the simulation. Note that this family of methods require constant interface thickness during the time and therefore it is crucial to frequently reinitialize the distance function.

In the second class of methods, the interface is considered sharp and no smoothing of the interface is considered. As the result, cut element with sharp jumps of several orders of magnitudes in material properties results in kink and/or discontinuities in the solution fields. Piecewise linear continuous approximations are too poor to capture these phenomena. Different enrichment techniques are then developed to treat this problem, local enrichments, that are condensed at element level, and global enrichments.

3.1 Local Enrichments

The incompressible Navier-Stokes equations governing the two-phase motion in domain Ω is written as:

$$\begin{aligned}&\rho(\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u}) - \nabla \cdot (2\mu \nabla^s \mathbf{u}) + \nabla p = \rho \mathbf{b}\\ &\nabla \cdot \mathbf{u} = 0\end{aligned} \tag{13}$$

This set of equations is completed by the appropriate Dirichlet and Neumann boundary conditions. The domain Ω is divided into two parts by the zero-level of the level set function denoted by $+$ and $-$ and with following material properties;

$$(\rho(\mathbf{x}), \mu(\mathbf{x})) = \begin{cases} (\rho^+, \mu^+) & \text{if } \mathbf{x} \in \Omega^+ \\ (\rho^-, \mu^-) & \text{if } \mathbf{x} \in \Omega^-. \end{cases}$$

Note that a jump in density at the interface Γ produces a discontinuity at the pressure gradient, or a jump in viscosity causes a discontinuity in pressure. Surface tension can also produces a jump in pressure field at the interface.

The balance of the interface and internal forces at the interface implies that,

$$(\sigma^- - \sigma^+).\mathbf{n} = \mathbf{f}_\Gamma.$$

Here we only consider the case that interface force $\mathbf{f}_\Gamma$ is surface tension and therefore is normal to the interface. The internal stress σ at each domain has the form $-pI + 2\mu\nabla^s\mathbf{u}$.

The variational equivalent of (13) is to find $(\mathbf{u}, p) \in V \times Q$ such that

$$\int_\Omega \rho(\partial_t \mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u})\cdot\mathbf{v}d\Omega + \int_\Omega 2\mu\nabla^s\mathbf{u} : \nabla^s\mathbf{v}d\Omega$$
$$-\int_\Omega p\nabla\cdot\mathbf{v}d\Omega = \int_\Omega \rho\mathbf{b}\cdot\mathbf{v}d\Omega + \int_\Gamma v\cdot f_\Gamma$$
$$\int_\Omega q\nabla\cdot\mathbf{u}d\Omega = 0$$

$\forall(\mathbf{v}, q) \in V \times Q$. A stabilized variational form can be obtained based on the Algebraic Subgrid Scale (ASGS) method. Trapezoidal rule for temporal discretization is considered and nonlinearities are best handled in an implicit fashion using a Newton-Raphson scheme. The discrete variational form of this problem then reads,

$$\mathcal{R}_\mathbf{u} = \int_\Omega \mathcal{G}_\mathbf{u}\cdot\mathbf{v}_h d\Omega + \int_\Omega 2\mu\nabla^s\mathbf{u}_h^{n+1} : \nabla\mathbf{v}_h d\Omega - \int_\Omega p_h^{n+1}\nabla\cdot\mathbf{v}_h d\Omega + \int_\Gamma f_{\Gamma_h}^{n+1}\cdot\mathbf{v}_h d\Gamma$$
$$+\sum_e \tau_1\int_{\Omega_e}\mathbf{u}_h^n\cdot\nabla\mathbf{v}_h\cdot(\mathcal{G}_\mathbf{u} + \nabla p_h^{n+1})d\Omega_e + \sum_e \tau_2\int_{\Omega_e}\nabla\cdot\mathbf{u}_h^{n+1}\nabla\cdot\mathbf{v}_h d\Omega_e = 0$$
$$\mathcal{R}_p = \int_\Omega q_h\nabla\cdot\mathbf{u}_h^{n+1}d\Omega + \sum_e\int_{\Omega_e}\frac{\tau_1}{\rho}(\mathcal{G}_\mathbf{u} + \nabla p_h^{n+1})\cdot\nabla q_h d\Omega_e = 0 \tag{14}$$

$\forall(\mathbf{v}_h, q_h) \in V_h \times Q_h$. In (14) the term $\mathcal{G}_\mathbf{u}$ is given by,

$$\mathcal{G}_\mathbf{u} = \rho(\frac{\mathbf{u}_h^{n+1} - \mathbf{u}_h^n}{\Delta t} + \mathbf{u}_h^{n+1}\cdot\nabla\mathbf{u}_h^{n+1} - \mathbf{b}^{n+1})$$

and the stabilization parameters given by,

$$\tau_1 = \frac{h_e^2}{4\nu + 2h_e|\mathbf{u}_e|} \quad \textit{and} \quad \tau_2 = \mu + 0.5\rho h_e|\mathbf{u}_e|$$

where h_e and $\mathbf{u}_e$ are elemental length and velocity, respectively. Solving (14) accurately requires the finite element space to be appropriately chosen to capture the possible kink or jumps that are expected at the solution.

When the interface cut elements on the fixed mesh, the jump in density causes a kink in pressure field and therefore a jump in its gradient in the cut elements (see Fig. 4). It is clear that for simple elements (triangles in 2D and tetrahedra in 3D) linear approximation of the pressure can not represent the kink in the cut element and in the same way the jump in pressure gradients. This pressure field in the cut elements does not belong to the standard pressure space made up of linear nodal interpolation and one way to capture this pressure filed is to add *enrichment* to the pressure field of the cut elements. One of such modifications proposed in [18] is to interpolate pressure as:

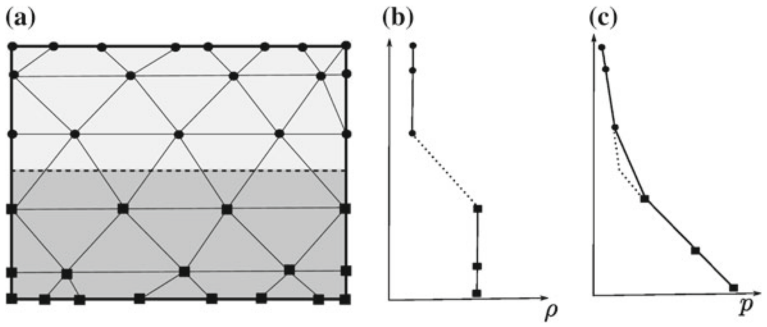


Fig. 4 Two-fluid hydrostatic flow with a jump in density. Linear elements can not represent kink in pressure field inside an element. **a** Interface pass through the elements, **b** jump in density in elements cut by the interface, **c** exact solution (*dotted line*) and FE solution (*solid line*)

$$p_h^e = \sum_i^{Nnode} N_i^e p_i^e + N_{enr}^e p_{enr}^e$$

$Nnode$ is the number of nodes per element and $Nenr$ is the new enrichment function added just for the elements cut by the interface. This function has the constant gradient at each side of the interface and is designed to be local to each element and therefore has zero value at the element nodes. Figure 5a shows a sketch of the enrichment function for an element cut by the inteface in the 2D case. Node a belongs to the Ω^- and nodes b and c to the Ω^+. N_{enr} is easily defined by the level set values ϕ at the element nodes as;

$$N_{enr} = 0.5(-|\sum_i^{Nnode} N_i^e \phi_i| + \sum_i^{Nnode} N_i^e |\phi_i|)$$

In order to capture the discontinuities and take advantage of the enrichment functions used, the integration rules need to be modified in elements cut by the front. To this end, each tetrahedral (triangular in 2D) element is divided into up to six tetrahedra (three triangular in 2D) sub elements. For each sub element the same integration rule as for the non-cut elements is used (see Fig. 5b). Further enrichments are needed in case that surface tension is present or viscosity jump arises at the interface. In this case the pressure solution belongs to a space that is discontinuous at the interface. Figure 6 shows one type of this enrichment for the 2D case that is introduced in [19]. The pressure is enriched at the cut elements by one additional local pressure at each side of the interface. This new space has the property that can capture a constant solution at each side with a possible jump at the interface. Note that both enrichment functions are zero at the nodes of the mesh. These enrichment functions has the following form;

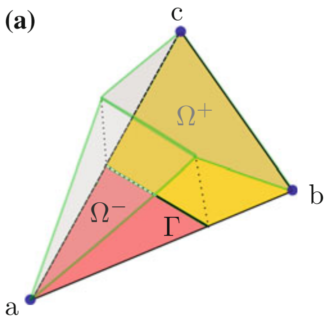


Fig. 5 Interface divides an element in 2D. **a** N_{enr} proposed in [18] to capture discontinuous pressure gradient in the cut elements. **b** Triangular sub elements used for the numerical integration. $\times$ shows the integration points

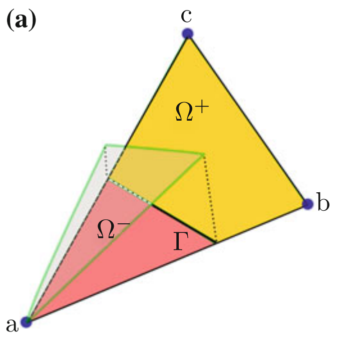


Fig. 6 Enrichment shape functions proposed in [19] to capture jump in the pressure field. Note that they are not continuous at the interface position Γ. **a** N^1_{enr}, **b** N^2_{enr}

$$N^1_{enr}(\mathbf{x}) = (1 - S(\mathbf{x}))\chi^+(\mathbf{x})$$
$$N^2_{enr}(\mathbf{x}) = S(\mathbf{x})\chi^-(\mathbf{x})$$

where the function S is given in terms of the standard P_1 functions as,

$$S = \sum_{i \in I^+} N^e_i$$

with $I^+ = \{i \in I, \mathbf{x} \in \Omega^+_e\}$, and χ^+ and χ^- the characteristic functions for the positive and negative sides. Note that the additional shape functions are local to each element crossed by the interface and therefore can be condensed prior to the assembly to maintain the size of the system matrix and its graph.

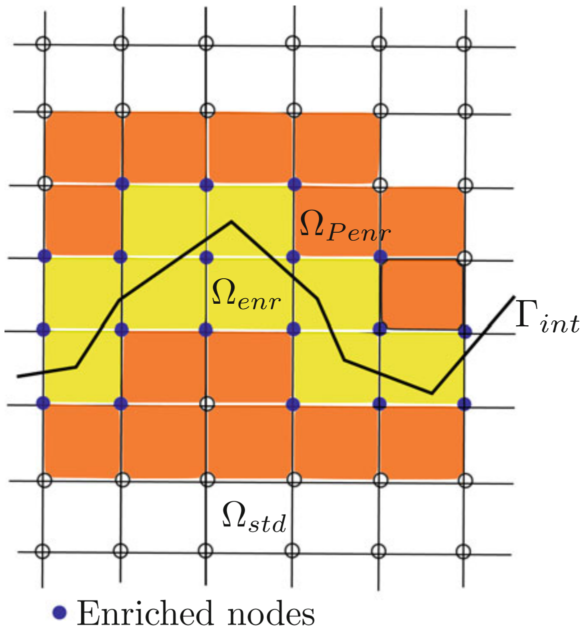


Fig. 7 Subdomains containing enriched, partially enriched and standard (non-enriched) elements as well as the enriched nodes [23]

3.2 Global Enrichments

By global enrichment we mean those enrichment techniques that add new degrees of freedom to the solution space, not condensed at the element level, and therefore change the graph of the matrix. In this way, extended finite element method (XFEM) is considered a global enrichments although the enrichment (extension) is local to the interface zone. One exception to our classification is the intrinsic XFEM method [20] that does not introduce any additional unknowns. All other standard XFEM methods developed for the two-phase flows [21–24], add new DOFs to the system. Similar to the local enrichment techniques, when XFEM is used for the simulation of two-phase flows, several enrichment schemes can be employed: velocity and/or pressure fields may be enriched and enrichments for kinks or jumps may be used. Numerical studies in [24] reveals that it is not advisable to enrich the velocity approximation space as it does not improve the results significantly, but may lead to severe convergence problems. Furthermore, the required number of iterations for the solution of the governing equations may increase considerably. On the other hand, the enrichment of the pressure field is essential.

Figure 7 shows three types of subdomains that can be distinguished in an XFEM enrichment scheme. The first one Ω_{enr} is composed of elements cut by the interface and are those that have all nodes enriched. The second subdomain, Ω_{penr}, has all elements that have at least one node enriched and the last subdomain is the collection of elements with standard degrees of freedom and without enrichment. All nodes belong to the elements cut by the interface are enriched. Let us define I^* as the collection of all nodes i that are enriched. The enrichment function M_i for these nodes are written as [24],

$$M_i(\mathbf{x}, t) = N_i(\mathbf{x}, t) \cdot [\psi(\mathbf{x}, t) - \psi(\mathbf{x}_i, t)] \qquad \forall i \in I^*$$

with $\psi(\mathbf{x}, t)$ being the global enrichment function and $N_i(\mathbf{x}, t)$ is the standard FE shape function for node i. $M_i(\mathbf{x}, t)$ is the so called shifted enrichment which ensures Kronecker-δ property of the overall approximation.

A global enrichment function that is typically chosen for strong discontinuities (jump in pressure) is the sign enrichment:

$$\psi_{sign}(\mathbf{x}, t) = sign(\phi(\mathbf{x}, t)) = \begin{cases} -1 & \text{if } \phi(\mathbf{x}, t) < 0 \\ 0 & \text{if } \phi(\mathbf{x}, t) = 0 \\ 1 & \text{if } \phi(\mathbf{x}, t) > 0. \end{cases}$$

where $\phi(\mathbf{x}, t)$ is the level set function. For weak discontinuities (kink in pressure fields), Moës et al. [25] proposed abs-enrichment shape function that has an important property of being zero on standard element (i.e. Ω_{penr} in Fig. 7). It is defined as;

$$\psi(\mathbf{x}, t) = \psi_{abs}(\mathbf{x}, t) = \sum_i^{Nnode} |\phi_i| N_i(\mathbf{x}, t) - |\sum_i^{Nnode} \phi_i N_i(\mathbf{x}, t)|$$

Note that this shape function is quite similar to the one proposed in [18] and shown in Fig. 5. Figure 8 shows the enrichment shape functions, M_i, for the weak and strong discontinuities, as mentioned above, in a quadrilateral element.

Note that in general, enrichment shape functions with small support can occur (when the interface is so close to a node) which lead to an ill-conditioned system matrix. Two approaches to treat this problem is found in [26, 27].

4 Application to Mold Filling Process

Simulation of mold filling process is very complex not only because of the variety of phenomena that appear, but also because of the high level of interactions between them. Fluid flow, heat transfer and phase change effects take place at the same time and with a complex coupling pattern. Different defects may arise during the mold filling stage like, air entrapment, slag inclusion, formation of cold shuts and cold laps and solidification. Heat transfer with the mold and the air can dramatically changes the material properties and therefore change the course of the flow. It may cause solidification in some material front while on the other zones the material is still

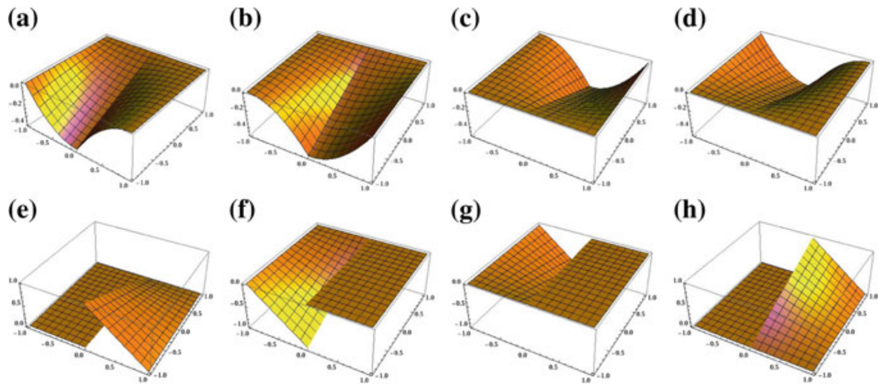


Fig. 8 Enrichment functions proposed by the XFEM method [23, 24] for, **a**–**d** weak discontinuities ($\psi = \psi_{abs}$) and, **e**–**f** strong discontinuities ($\psi = \psi_{sign}$). The interface is considered to pass through the middle of the element. **a** M_1^{abs}, **b** M_2^{abs}, **c** M_3^{abs}, **d** M_4^{abs}, **e** M_1^{sign}, **f** M_2^{sign}, **g** M_3^{sign}, **h** M_4^{sign}

expanding. Cold shuts arise when the metal flowing in a section solidifies and the flow is blocked before the region is completely filled. Cold laps are formed when the metal stream separates and subsequently joins in a region where the metal has already solidified. Both cold shuts and cold laps are formed due to excessive heat loss from the flowing metal to the mold.

The flow regime in filling stage is turbulent in most cases. The standard turbulence models require a very fine grid near the mold walls and it is rather impractical to achieve such a fine grid in all the sections of a complex mold. Hence, only laminar flow computations are usually done for mold filling predictions. Some authors have used upwinding algorithms and/or turbulence models to obtain convergent results on practically feasible grids. These models effectively increase the viscosity of the fluid and make it possible to achieve stable solutions. The other important aspect in filling process is the boundary condition. If no-slip boundary conditions are enforced along the wall, the predicted filling pattern is unrealistic as the metal flowing adjacent to the wall tends to stick to the wall itself. It can be understood as the convective velocity becomes zero, for these nodes, in the level set Eq. (1). Slip boundary condition is therefore prescribed for the fluid all along the mold wall. This may be seen as a simple turbulence model and is feasible for complex mold geometries; the velocity profile obtained in thin sections is almost uniform across the cross-section and thus mimics the turbulent flow profile. The near-wall (viscous sublayer) region is not included in the flow computation as in any other high Reynolds number turbulent flow model. Instead, tangential traction forces can be prescribed on the slip boundaries to model the effect of wall friction. This stress can be tuned by empirical factors to mimic the mold wall material friction and therefor get more realistic flow pattern. For high pressure die casting process, that the mold is usually made of still and with less friction, this scaling factor is chosen much less than the gravity die casting in which the mold is made of sand and therefor is much more rougher.

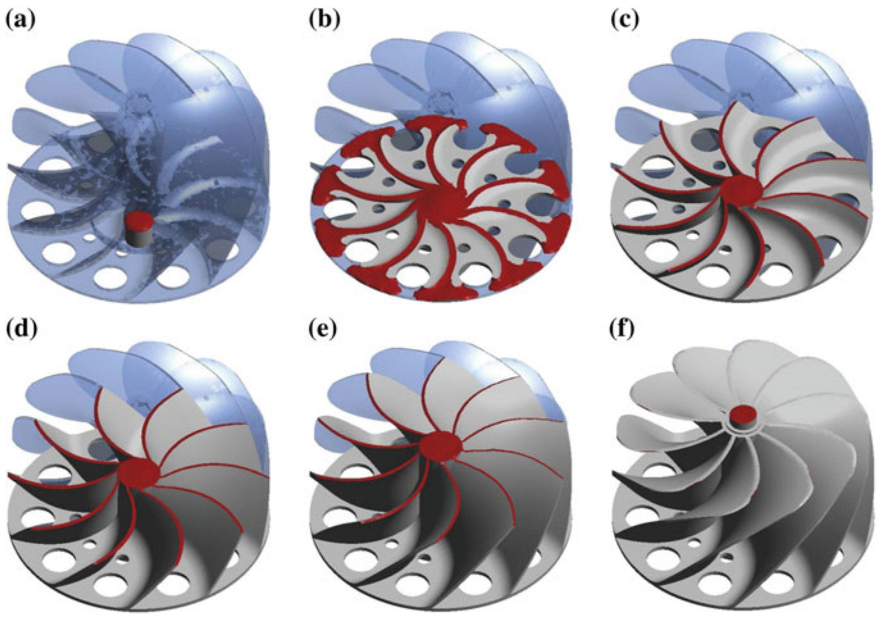


Fig. 9 Filling of a turbine blade. *Red* color shows the aluminum-air interface at different instances. The inlet velocity is 0.3 m/s and the blade is filled from the *bottom*. **a** t= 0.2 s, **b** t= 4 s, **c** t= 6 s, **d** t= 9 s, **e** t= 12 s, **f** t= 15 s

Another important aspect in mold filling simulation is the air exits. It is known that in the sand molds the porous texture of the mold provide air exits. In the two-fluid simulation that the air inside the mold is modeled as incompressible, it is essential to provide holes or exits in the mold wall to vent the air as the metal fills the cavity. This is very important particularly when coarse grids are used. Otherwise, air tends to recirculate in thin sections, corners or closed zones and prevents the metal reaching those regions and therefore unrealistically large pockets of air are entrapped. Once the fluid has reached the vent zones they are considered "closed" to avoid the excessive loss of material.

Figure 9 shows various instances during the filling of a turbine blade. Level set is used to detect the interface position and the two-fluid multi-scale stabilized incompressible formulation (14) is used. Among the different enrichments presented in Sect. 3, the discontinuous pressure gradient is applied. The filling material is Aluminum with density equal to 2600 kg/m^3 and the air has density of 1 kg/m^3. In order to avoid unnecessary jumps in pressure the same dynamic viscosity, μ, equal to $1e-5$ $\frac{\text{kg}}{\text{m}}$s is considered. Boundaries are slip and a friction force of the following form is applied in the direction opposite to the velocity at the boundary nodes,

$$\mathbf{f}_{wall} = C\rho \parallel \mathbf{u} \parallel (\mathbf{I} - \mathbf{n} \otimes \mathbf{n}) \cdot \mathbf{u}$$

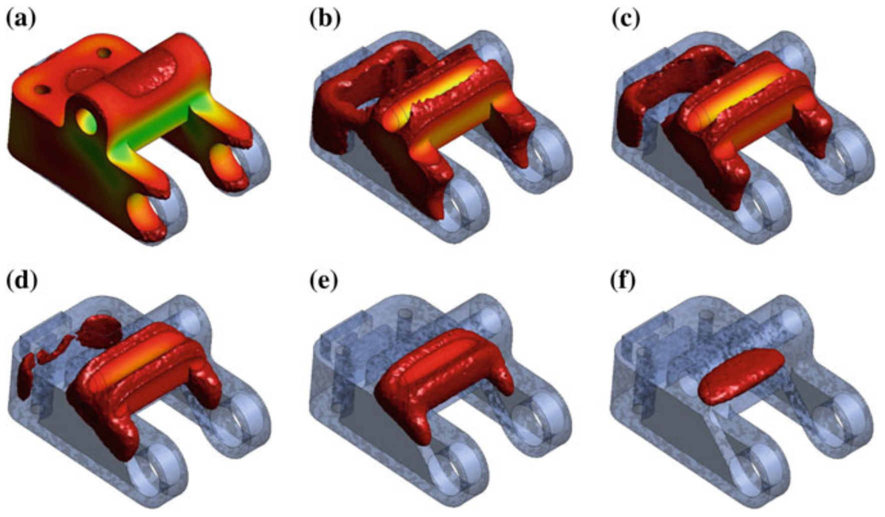


Fig. 10 Post-filling solidification of a mechanical part. *Red* zones are the liquid zones and time is measured from the moment that filling is completed. **a** $t = 7$ s. **b** $t = 12$ s. **c** $t = 14$ s. **d** $t = 16$ s. **e** $t = 19$ s. **f** $t = 25$ s

here C is a constant that is chosen between $\{0.005, 0.05\}$, $\mathbf{u}$ is the velocity at the Gauss point and $\mathbf{n}$ is the exterior normal. ρ for the mixed element is chosen as metal material.

One of the major components of the post-filling simulation is the solidification analysis. Solidification is accompanied by the release of latent heat at the solid-liquid interface. Heat transfer between the filling material and the mold causes colling from a liquid state until the material begins to solidify at the liquidus temperature, T_l, and solidifies completely at the solidus temperature, T_s. The solid and liquid phases are separated by a transition mixture region called *mushy zone*. One way to model the solidification process is the *effective specific heat* method in which the energy equation is written in terms of an effective specific heat $\bar{c}$;

$$\rho\bar{c}\frac{\partial T}{\partial t} = \nabla \cdot (k\nabla T)$$

where T and k are temperature and thermal conductivity, respectively. The parameter $\bar{c}$ is the slope of the enthalpy-temperature curve and is equal to the specific heat, c, in the solid and liquid region. In the mushy zone its value is given by

$$\bar{c} = c - \frac{df_s}{dT}L$$

here L is the latent heat and f_s the solid fraction. When the solid fraction, f_s, is assumed to vary linearly with temperature (i.e. $f_s = (T_l - T)/(T_l - T_s)$), the value of $\bar{c}$ is constant in the mushy zone as given below:

$$\bar{c} = c + \frac{L}{T_l - T_s}$$

Figure 10, shows the post-filling solidification process of a mechanical part. Solidus temperature, T_s, and liquidus temperature, T_s are taken as 630 and 542 °C, respectively. The latent heat varies linearly with respect to temperature and between 2000–3000 kj/kg. The volume of the mechanical part is 215 cm^3 and it is solidified in 35 s.

Our formulation is implemented in the free source parallel multi-physics software platform of KRATOS [28, 29] developed at CIMNE.

5 Conclusion

The main challenge in the application of level set method to capture the interface, is the mass conservation during both the convection and reinitialization step. To treat this problem different level set techniques have been developed. The particle level set method, PLS, and the coupled level set/volume-of-fluid method, CLSVOF, have been mainly developed for the structural quadrilateral meshes and works quite efficiently in this kind of meshes. On the other hand, geometric mass-preserving redistance method is developed for both the structured and unstructured meshes and is particularly useful in case that redistancing error is dominant i.e. fine meshes and small Δt. Discontinuous Galerkin level set method (DGLSM) is quadrature-free, works on structured and unstructured meshes and as the polynomial order increases ($p > 4$) its superiority to other method is evident even on very coarse meshes (see Table 1). The convective velocity in in the level set equation comes form the solution of Navier-Stokes equations. In case that linear elements are used, enrichment is necessary at the interface level to improve and stabilize the velocity-pressure pairs at the interface. Various enrichment techniques are developed to treat kink or discontinuities due to the jump in material properties at the element level. We presented them as the local and non-local, though both of them add enrichments at the interface zone. Local enrichment add DOFs that can be condensed at the element level and therefore does not change the graph of the global matrices as the interface moves, and the non-local enrichment, XFEM, add global DOFs that change the graph of the global matrices and can be quite costly for the practical applications. In both of these techniques stability issues have to be taken into account in case that the interface passes so close to the element nodes.

References

1. Osher S, Shu C-W (1991) High-order essentially nonoscillatory schemes for hamilton-jacobi equations. SIAM J Numer Anal 28(4):907–922

2. Osher S, Fedkiw R (2003) Level set methods and dynamic implicit surfaces, vol 153. Springer, New York
3. Sussman M, Fatemi E (1999) An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. SIAM J Sci Comput 20(4):1165–1191
4. Sethian JA (1999) Fast marching methods. SIAM Rev 41(2):199–235
5. Enright D, Fedkiw R, Ferziger J, Mitchell I (2002) A hybrid particle level set method for improved interface capturing. J Comput Phys 183(1):83–116
6. Sussman M, Puckett EG (2000) A coupled level set and volume-of-fluid method for computing 3d and axisymmetric incompressible two-phase flows. J Comput Phys 162(2):301–337
7. Sussman M (2003) A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. J Comput Phys 187(1):110–136
8. Mut F, Buscaglia GC, Dari EA (2004) A new mass-conserving algorithm for level set redistancing on unstructured meshes. Mecanica Computacional 23:1659–1678
9. Ausas RF, Dari EA, Buscaglia GC (2011) A geometric mass-preserving redistancing scheme for the level set function. Int J Numer Meth Fluids 65(8):989–1010
10. Marchandise E, Remacle J-F, Chevaugeon N (2006) A quadrature-free discontinuous galerkin method for the level set equation. J Comput Phys 212(1):338–357
11. Idelsohn S, Mier-Torrecilla M, Oñate E (2009) Multi-fluid flows with the particle finite element method. Comput Methods Appl Mech Eng 198(33):2750–2767
12. Kamran K, Rossi R, Oñate E, Idelsohn SR (2012) A compressible lagrangian framework for the simulation of the underwater implosion of large air bubbles. Comput Methods Appl Mech Eng 225(1):210–225
13. Bonet J, Kulasegaram S (2000) Correction and stabilization of smooth particle hydrodynamics methods with applications in metal forming simulations. Int J Numer Meth Eng 47(6):1189–1214
14. Sunitha N, Jansen KE, Lahey RT Jr (2005) Computation of incompressible bubble dynamics with a stabilized finite element level set method. Comput Methods Appl Mech Eng 194(42):4565–4587
15. Kees CE, Akkerman I, Farthing MW, Bazilevs Y (2011) A conservative level set method suitable for variable-order approximations and unstructured meshes. J Comput Phys 230(12):4536–4558
16. Rossi R, Larese A, Dadvand P, Oñate E (2012) An efficient edge-based level set finite element method for free surface flow problems. Int J Numer Methods Fluids 33:737–766
17. Sussman M, Smereka P, Osher S (1994) A level set approach for computing solutions to incompressible two-phase flow. J Comput phys 114(1):146–159
18. Coppola-Owen AH, Codina R (2005) Improving eulerian two-phase flow finite element approximation with discontinuous gradient pressure shape functions. Int J Numer Methods Fluids 49(12):1287–1304
19. Ausas RF, Buscaglia GC, Idelsohn SR (2012) A new enrichment space for the treatment of discontinuous pressures in multi-fluid flows. Int J Numer Methods Fluids 70(7):829–850
20. Fries T-P, Belytschko T (2006) The intrinsic xfem: a method for arbitrary discontinuities without additional unknowns. Int J Numer Meth Eng 68(13):1358–1385
21. Chessa J, Belytschko T (2003) An extended finite element method for two-phase fluids: flow simulation and modeling. J Appl Mech 70(1):10–17
22. Groß S, Reusken A (2007) An extended pressure finite element space for two-phase incompressible flows with surface tension. J Comput Phys 224(1):40–58
23. Rasthofer U, Henke F, Wall WA, Gravemeier V (2011) An extended residual-based variational multiscale method for two-phase flow including surface tension. Comput Methods Appl Mech Eng 200(21):1866–1876
24. Sauerland H, Fries T-P (2011) The extended finite element method for two-phase and free-surface flows: a systematic study. J Comput Phys 230(9):3369–3390
25. Moës N, Cloirec M, Cartraud P, Remacle J-F (2003) A computational approach to handle complex microstructure geometries. Comput Methods Appl Mech Eng 192(28):3163–3177

26. Reusken A (2008) Analysis of an extended pressure finite element space for two-phase incompressible flows. Comput Vis Sci 11(4–6):293–305
27. Béchet E, Minnebo H, Moës N, Burgardt B (2005) Improved implementation and robustness study of the x-fem for stress analysis around cracks. Int J Numer Meth Eng 64(8):1033–1056
28. Dadvand P, Rossi R, Oñate E (2010) An object-oriented environment for developing finite element codes for multi-disciplinary applications. Arch Comput Methods Eng 17(3):253–297
29. Dadvand P, Rossi R, Gil M, Martorell X, Cotela J, Juanpere E, Idelsohn SR, Oñate E (2012) Migration of a generic multi-physics framework to hpc environments. Comput fluids 80:301–309

Recent Advances in the Particle Finite Element Method Towards More Complex Fluid Flow Applications

Norberto M. Nigro, Juan M. Gimenez and Sergio R. Idelsohn

Abstract This paper presents a state of the art in the Particle Finite Element Method, normally called PFEM, its emphasis in the new ideas oriented to extend its application not only to solve fluid structure interaction and multifluid problems, also bring new opportunities to shorten the gap between engineering design times and computational simulation times for general problems when Eulerian formulation were typically chosen. In order to reduce the long history of this method here the starting point begins with the reformulation of the method to solve academic and real problems in real time or at least in drastically reduced computational times. The main topics involved in this paper are around the stability and the accuracy of Lagrangian formulations against its Eulerian counterpart shown through several academic benchmarks and a deep analysis of the efficiency revealing that the original method needs some new features. The former brought out a new integration method called X-IVAS and the later has produced a new version of the method called PFEM in fixed Mesh. Once the method had shown its good performance and how the new features impact on the final efficiency the last developments had been done in extending the application of this new method in multifluids and other complex fluid mechanics problems like turbulence and reactive flows.

N. M. Nigro (✉) · J. M. Gimenez
Centro de Investigacion en Metodos Computacionales (CIMEC), Consejo Nacional de Investigaciones Cientificas y Tecnicas (CONICET), Santa Fe, Argentina
e-mail: nnigro@intec.unl.edu.ar

N. M. Nigro
Centro de Investigacion en Metodos Computacionales (CIMEC), Universidad Nacional del Litoral (UNL), Santa Fe, Argentina

S. R. Idelsohn
Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE), Institució Catalana de Recerca i Estudis Avançats (ICREA), Barcelona, Spain

S. R. Idelsohn
Centro de Investigaciones en Mecánica Computacional (CIMEC), Universidad Nacional del Litoral (UNL), Santa Fe, Argentina

1 Introduction

Standard formulations for the solution of the incompressible Navier-Stokes equations may be split in two classes depending on the approach chosen for the description of the inertial terms, namely Eulerian and Lagrangian approaches. In the first class, the acceleration is described as the sum of the spatial derivative of the velocity plus the convective term. In the second approach, the acceleration is simply described as the total derivative of the velocity. Over the last 30 years, computer simulation of incompressible flows has been mainly based on the Eulerian formulation (see [18] for references on this topic). However, with this formulation, it is still difficult to analyze large 3D problems in which the shape of the free-surfaces or internal interfaces changes continuously or in fluid structure interactions where complex contact problems are involved. In all these problems the computing time is sometimes so high that makes the method unpractical. In the last few years, several solutions using the Lagrangian formulation to solve the compressible and incompressible Navier-Stokes equations have been developed [3, 9, 11]. The advantages of these solutions to solve problems with free-surfaces or multi-fluids with complicated internal interfaces have been demonstrated [15]. In general, these formulations are more expensive than a standard Eulerian approaches if they are used in homogeneous flows, but they justify their popularity in solving free- surface flows or complicated multi-fluids flows in which the standard Eulerian formulations are inaccurate or, sometimes, impossible to be used [15].

When attempting to compare the efficiency of Eulerian codes against Lagrangian ones the conclusions were not so definite. Even though Lagrangian solvers are simpler than Eulerian ones the very small time steps normally employed in the former has reduced its application only to specific examples.

Only few attempts in the past thought in using Lagrangian formulation for homogeneous fluid flow. To cite here only a few contributions we can mention the work of Joe Stam [23]. In these work the author solve the Navier-Stokes equations in the context of video games leaving the message that it is possible to design simpler numerical methods that may be applied on this challenge context where the efficiency is the key point.

One of the reasons why there are so few jobs using Lagrangian methods for homogeneous fluid flow applications may be the important computational cost involved in the remeshing. Lagrangian solvers are principally based on moving particles, and after that some sort of mesh should be built depending on the specific method the way to do that. *PFEM* is one of the most popular Lagrangian methods with the particular fact that the moving particles define a mesh where the discrete equations are solved. Its origins go back to the early 2,000. For brevity reasons its state of the art is given up here. Readers interested in the basis of the method may see http://www.cimne.com/pfem/ where there are most of the publications of the method. Among the most cited publications here we can mention [3, 12, 13, 19].

This feature obliged the method to deform the mesh up to certain limit where for geometric reasons some sort of remeshing should be done. As the remeshing was only limited to some special time intervals the deforming mesh added another ingredient to the time step selection, to avoid the mesh inversion. This severe limitation together with another imposed for the non-linearities and those proper of explicit schemes made the efficiency of original *PFEM* a serious problem. Lately the method evolved thanks to the progress done in parallel mesh generation and remeshing avoiding this serious limitation in some measure.

Even though these limitations and the large community that normally employ Eulerian codes the Lagrangian formulation contains some nice features that need to be reviewed here.

One of the most important rests on the missing of the convective term in the balance equations, converting the non-symmetric equations in symmetric and positive definite. For Navier-Stokes equations this fact has a by-product, converting in linear the original non-linear momentum equation. These two facts avoid the usage of stabilization terms with the strong consequence of not adding the typical numerical diffusion needed to stabilize it. Not having convective terms, for constant coefficient problems as for laminar and homogeneous fluid flow and also for DNS (Direct Numerical Simulation), the system matrices may be factorized at the beginning and reusing them all the time steps, with an important saving in cpu-time. For convection dominated flows the time step in Eulerian formulations needs to be limited attending non-linearities and stability reasons. On the contrary, the Lagrangian formulations do not suffer from this inconvenience when the equations are integrated with good accuracy. This is a key point that deserves much more attention.

In particular *PFEM* has evolved considerably over the past few years, incorporating new ideas seeking enlarge the time steps largely in stable and accurate way.

In this sense *PFEM* has incorporated a novel time integration scheme called X-IVS and its extension X-IVAS. This form of integrating based in following the streamlines of the flow in the present time step is to some extent a better way to solve the non-linearities of the equations of the flow.

In this way it is possible to solve the complex flow situations allowing to extend the time steps in a significant way.

On the other hand being the information carried by the particles using the mesh only for computing secondary fields confers to the method of high accuracy.

Therefore the goals of accuracy, robustness (stability) and efficiency are significantly improved by these new ideas included in the last version of *PFEM*, called Fixed Mesh PFEM.

One of the main target of this work is to show that Lagrangian formulations are not only valuable to solve heterogeneous fluid flows with free-surfaces. We will prove on the contrary that even for homogeneous fluid flows, without free-surfaces or internal interfaces, they are able to yield accurate solutions while being competitively fast when compared to state-of-the-art eulerian solvers. Also, another goal of this paper is to update the state of the art of *PFEM* joining some basis published before [6, 7, 10, 11, 22], with new findings discovering more and more nice features of the method to become a competitive tool in the future for high performance computations.

The paper starts with a mathematical review of the problems to be treated writing them in an Eulerian and a Lagrangian formalism.

Next, the time integration schemes are presented where it is possible to understand the novelty introduced by X-IVS and X-IVAS.

It is followed by a section dedicated to two examples that have served as inspiring muses for the development of new ideas which were then applied to PFEM. In these examples may be understood the benefits of using Lagrangian solvers. While these examples solved by Eulerian codes needs a lot of numerical artifacts, they are trivially solved by Lagrangian ones. The next section presents the two versions of PFEM. The first called Mobile Mesh Version is an extension to the original PFEM with permanent remeshing and a X-IVAS time integration scheme included. Showing the pros and cons the rest of the section is devoted to the novel idea of mixing two view points, one based on particles and the other based on the background and fixed mesh. This idea allowed to increase the efficiency in a very important way. Even though some earlier attempts had been done in using the duality of particle and mesh, for example [8], at the moment of designing the idea this information was not on the knowledge of the authors and moreover, both ideas have only few things in common.

The next section presents some details about the Fixed Mesh version of PFEM, how to manage the particle inventory, how to share the information between particles and mesh. It is followed by a section where the focus is on the treatment of the diffusive terms. Contrary to what may be a prior assumed, the Lagrangian behavior has been superior to the Eulerian one, in regard to precision being that this part of the calculation is of Eulerian nature. The last section is devoted to show some examples solved numerically by PFEM where it is possible to realize that in the present status PFEM is able to solve turbulent flows, multifluids and multiphase flows, general multiphysics, among others. Finally some conclusions are included.

2 X-IVAS: A New Integration Method to Enhance Accuracy and Stability

In this section the emphasis is placed on the main features that produce the big advantages of *PFEM* against any other method. In general the interesting problem to be solved is the general transport equation that is very widespread in the engineering applications. Both, the passive scalar transport equation and the incompressible flow represented by the Navier-Stokes equations will be considered in the rest of the paper.

In order to understand the evolution of *PFEM* the Eulerian and Lagrangian formulations are introduced first.

2.1 Scalar Transport Equation

In the Eulerian framework a fixed coordinate system is considered as the reference for the physical quantities. The scalar transport equation is written as:

$$\frac{\partial T}{\partial t} + \nabla \cdot (\mathbf{v}T) = \nabla \cdot (\alpha \nabla T) + Q \tag{1}$$

where $T(\mathbf{x}, t)$ is the dependent variable (passive scalar), for example the temperature, $\mathbf{v}$ is the velocity vector, for this problem a given data and α the diffusivity, with $\nabla\cdot$ the divergence operator, ∇ the gradient operator and $\frac{\partial}{\partial t}$ the temporal derivative. In this problem $\mathbf{x}$ represents a fixed coordinate.

Normally this equation may be rewritten in the following form:

$$\frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T = \nabla \cdot (\alpha \nabla T) + Q - (\nabla \cdot \mathbf{v})T \tag{2}$$

where the first order derivative is split in two terms, one for the convective transport and the other for the source term generated by the non free divergence velocity field. Normally the incompressible flow satisfies the free divergence and in this case this source term may be neglected.

On the other hand in the Lagrangian framework the same problem is written as:

$$\frac{DT}{Dt} = \nabla \cdot (\alpha \nabla T) + Q \tag{3}$$

where $\frac{DT}{Dt} = \frac{\partial T}{\partial t} + \mathbf{v} \cdot \nabla T$ is the material derivative. The convective term works like a variable transformation between that measured in a fixed coordinate system and that measured in the moving coordinate system that travels with the fluid velocity $\mathbf{v}$. In this transformation the velocity field is incorporated in the dependent variable itself being the unknown variable $T = T(\mathbf{x}_p, t)$ with $\mathbf{x}_p$ the location of a fluid parcel included within the material volume. This location is at the same time another variable, so it is needed to solve an additional equation like:

$$\frac{D\mathbf{x_p}}{Dt} = \mathbf{v} \tag{4}$$

Finally the problem in Lagrangian framework is:

$$\begin{aligned} \frac{DT}{Dt} &= \nabla \cdot (\alpha \nabla T) + Q \\ \frac{D\mathbf{x_p}}{Dt} &= \mathbf{v} \end{aligned} \tag{5}$$

2.2 Incompressible Viscous Fluid Flow: Navier-Stokes Equations

The other problem that in this paper deserves special attention is the fluid dynamics of an incompressible and viscous flow. It is very well known that this problem is governed by the Navier-Stokes equations that presents the balance of the linear momentum equation and the continuity equation or the mass balance.

Both equations normally written together in an Eulerian framework look like:

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\mathbf{v}\rho) &= 0 \\ \frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v}) &= \nabla \cdot (\sigma) + \mathbf{F} \end{aligned} \tag{6}$$

Being σ the stress tensor which definition may be split in two parts, one for the spherical (isotropic) component being proportional to the fluid pressure and the other, the deviatoric or viscous component normally written as τ. The operator $\otimes$ means the tensor or dyadic product between two vectors. **F** represents the external force, for example the gravity, and finally ρ is the density. For incompressible flows the density is constant, therefore, the continuity equation becomes a constrain over the velocity field, as:

$$\nabla \cdot (\mathbf{v}) = 0 \tag{7}$$

Applying the above restriction also in the linear momentum equation produces a simplified and non-conservative version like

$$\begin{aligned} \nabla \cdot (\mathbf{v}) &= 0 \\ \rho(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v}) &= \nabla \cdot (\sigma) + \mathbf{F} \end{aligned} \tag{8}$$

For the Lagrangian formulation the above system is written as:

$$\begin{aligned} \nabla \cdot (\mathbf{v}) &= 0 \\ \rho \frac{D\mathbf{v}}{Dt} &= \nabla \cdot (\sigma) + \mathbf{F} \\ \frac{D\mathbf{x_p}}{Dt} &= \mathbf{v} \end{aligned} \tag{9}$$

2.3 Time Integration

In this section the time integration of both frameworks, the Eulerian and the Lagrangian is presented.

For simplicity the scalar transport equation is chosen first leaving for some special topics the extension to the vector equation system governing the fluid dynamics of one phase incompressible viscous flow.

2.3.1 Scalar Transport Equation

For the Eulerian framework represented by Eq. (2) the integration is normally done as

$$\begin{aligned}\int_{t^n}^{t^{n+1}} \frac{\partial T}{\partial t} dt &= \int_{t^n}^{t^{n+1}} (-\mathbf{v}\cdot\nabla T + \nabla\cdot(\alpha\nabla T) + Q)dt \\ T^{n+1}(\mathbf{x}) - T^n(\mathbf{x}) &= \int_{t^n}^{t^{n+1}} (-\mathbf{v}\cdot\nabla T(\mathbf{x}) + \nabla\cdot(\alpha\nabla T(\mathbf{x})) + Q(\mathbf{x},t))dt \\ T^{n+1}(\mathbf{x}) - T^n(\mathbf{x}) &= (-\mathbf{v}\cdot\nabla T(\mathbf{x}) + \nabla\cdot(\alpha\nabla T(\mathbf{x})) + Q(\mathbf{x},t))^{n+\theta}\Delta t\end{aligned} \tag{10}$$

For some $\theta \in (0, 1)$ the last expression in (10) gives the exact solution. As this parameter is unknown and problem dependent some fixed values for θ are adopted, $\theta = 0$ for explicit schemes, $\theta = 1$ for implicit schemes and $\theta = \frac{1}{2}$ for Crank-Nicholson among others.

$$\begin{aligned}(-\mathbf{v}\cdot\nabla T + \nabla\cdot(\alpha\nabla T) + Q)^{n+\theta} = \theta\Big(-\mathbf{v}\cdot\nabla T + \nabla\cdot(\alpha\nabla T) + Q\Big)^{n+1} \\ + (1-\theta)\Big(-\mathbf{v}\cdot\nabla T + \nabla\cdot(\alpha\nabla T) + Q\Big)^{n}\end{aligned} \tag{11}$$

Replacing (11) in (10)

$$\begin{aligned}T^{n+1}(\mathbf{x}) - T^n(\mathbf{x}) = \theta\Big(-\mathbf{v}\cdot\nabla T + \nabla\cdot(\alpha\nabla T) + Q\Big)^{n+1}\Delta t \\ + (1-\theta)\Big(-\mathbf{v}\cdot\nabla T + \nabla\cdot(\alpha\nabla T) + Q\Big)^{n}\Delta t\end{aligned} \tag{12}$$

The right hand side in (12) is evaluated using only the information of the nodal point $\mathbf{x}$ at the two extremes of the time interval, t^n and $t^{n+1} = t^n + \Delta t$.

For the Lagrangian framework a similar integration scheme is applied.

$$
\begin{aligned}
\int_{t^n}^{t^{n+1}} \frac{DT}{Dt} dt &= \int_{t^n}^{t^{n+1}} (\nabla \cdot (\alpha \nabla T) + Q)^t dt \\
\int_{t^n}^{t^{n+1}} \frac{D\mathbf{x_p}}{Dt} dt &= \int_{t^n}^{t^{n+1}} \mathbf{v}^t dt \\
T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^n, t^n) &= \int_{t^n}^{t^{n+1}} \Big(\nabla \cdot (\alpha \nabla T) + Q\Big)^t dt \\
\mathbf{x_p}^{n+1} - \mathbf{x_p}^n &= \int_{t^n}^{t^{n+1}} \mathbf{v}^t \; dt
\end{aligned}
\tag{13}
$$

In a straightforward way it is possible to apply (11) in (13) producing the following standard Lagrangian form:

$$
\begin{aligned}
T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^n, t^n) &= \int_{t^n}^{t^{n+1}} \Big(\nabla \cdot (\alpha \nabla T) + Q\Big)^t dt \\
&= \theta\Big(\nabla \cdot (\alpha \nabla T) + Q\Big)^{n+1} \Delta t + \\
&(1-\theta)\Big(\nabla \cdot (\alpha \nabla T) + Q\Big)^{n} \Delta t \\
\mathbf{x_p}^{n+1} - \mathbf{x_p}^n &= \theta \mathbf{v}^{n+1} \Delta t + (1-\theta)\mathbf{v}^n \Delta t
\end{aligned}
\tag{14}
$$

2.3.2 Navier-Stokes

The extension of the time discretization to the Navier-Stokes equations needs to solve the pressure-velocity coupling.

It is well known that the velocity vector unknown arises solving the vector momentum equation. Being the pressure the scalar unknown for which the continuity equation might be the natural choice, in this equation the pressure does not appear. Moreover, this equation is not a time evolution equation, it works like a constraint over the velocity field, choosing only those velocity field that satisfy a free divergence. To discover the equation associated with the pressure several alternatives are possible. Among them, segregated or projection methods like fractional step appear as good candidates. The idea behind the fractional step is to write the momentum equation discretized in time in such a way to firstly predict a velocity using the old value of the pressure (the pressure at the old time step) and after correcting it with the updated pressure that arises from applying the divergence operator to the correction equation getting a Poisson like equation for the pressure.

In synthesis the fractional step may be viewed as:

$$
\begin{aligned}
\mathbf{v}^{n+1} - \mathbf{v}^{n} &= \Big(\nabla\cdot\boldsymbol{\sigma} + \mathbf{f}\Big)^{n+\theta}\Delta t\\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^{n} &= \theta\Big(\nabla\cdot\boldsymbol{\sigma} + \mathbf{f}\Big)^{n+1}\Delta t + (1-\theta)\Big(\nabla\cdot\boldsymbol{\sigma} + \mathbf{f}\Big)^{n}\Delta t\\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^{n} &= \theta\Big(-\nabla p + \nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n+1}\Delta t +\\
&\qquad (1-\theta)\Big(-\nabla p + \nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t\\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^{n} &= \theta(-\nabla p^{n+1} + \nabla p^{n})\Delta t - \theta\nabla p^{n}\Delta t + \theta\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n+1}\Delta t\\
&\quad + (1-\theta)\Big(-\nabla p + \nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t\\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^{n} &= \theta(-\nabla p^{n+1} + \nabla p^{n})\Delta t - \nabla p^{n}\Delta t\\
&\quad + \theta\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n+1}\Delta t\\
&\quad + (1-\theta)\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t\\
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} + \widehat{\mathbf{v}}^{n+1} - \mathbf{v}^{n} &= \theta(-\nabla p^{n+1} + \nabla p^{n})\Delta t - \nabla p^{n}\Delta t + \theta\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n+1}\Delta t\\
&\quad + (1-\theta)\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t\\
\underbrace{\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1}}_{CORRECTOR} + \underbrace{\widehat{\mathbf{v}}^{n+1} - \mathbf{v}^{n}}_{PREDICTOR} &= \underbrace{\theta(-\nabla p^{n+1} + \nabla p^{n})\Delta t}_{CORRECTOR} - \underbrace{\nabla p^{n}\Delta t}_{PREDICTOR}\\
&\quad + \underbrace{\theta\Big(\nabla\cdot\mathbf{v}(\widehat{\mathbf{v}}^{n+1}) + \mathbf{f}\Big)^{n+1}\Delta t + (1-\theta)\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t}_{PREDICTOR}
\end{aligned}
\tag{15}
$$

Spiting the predictor and corrector parts of the equation in two steps and applying the divergence to the corrector step using the constraint that $\nabla\mathbf{v}^{n+1} = 0$,

PREDICTOR

$$
\begin{aligned}
\widehat{\mathbf{v}}^{n+1} - \mathbf{v}^{n} &= -\nabla p^{n}\Delta t + \theta\Big(\nabla\cdot\mathbf{v}(\widehat{\mathbf{v}}^{n+1}) + \mathbf{f}\Big)^{n+1}\Delta t\\
&\quad + (1-\theta)\Big(\nabla\cdot\mathbf{v} + \mathbf{f}\Big)^{n}\Delta t
\end{aligned}
$$

PRESSURE EQUATION

$$
\begin{aligned}
\nabla\cdot\Big(\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} &= -\theta\Delta t(\nabla p^{n+1} - \nabla p^{n})\Big)\\
\nabla\cdot\widehat{\mathbf{v}}^{n+1} &= \theta\Delta t\nabla\cdot(\nabla\delta p) = \theta\Delta t\nabla^{2}\delta p
\end{aligned}
$$

CORRECTOR

$$
\mathbf{v}^{n+1} - \widehat{\mathbf{v}}^{n+1} = -\theta\Delta t\nabla\delta p \tag{16}
$$

with $\delta p = p^{n+1} - p^n$ and ∇^2 the Laplacian operator. The equation for the pressure is interposed between the predictor and corrector equations for the momentum equation as it is normally found in the algorithm.

For the Eulerian formulation the above three steps may be applied straightforward only changing $\mathbf{v}$ by $\mathbf{v}(\mathbf{x})$ and p by $p(\mathbf{x})$.

For the Lagrangian formulation the above algorithm may be summarized as:

PREDICTOR
Explicit part

$$\begin{aligned}
\mathbf{x_p}^{n+1} &= \mathbf{x_p}^{n} + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x_p}^{\tau})d\tau \\
\hat{\hat{\mathbf{v}}}^{n+1}(\mathbf{x_p}^{n+1}) - \mathbf{v}^n(\mathbf{x_p}^{n}) &= \\
&\int_{t^n}^{t^{n+1}} -\nabla p^n(\mathbf{x_p}^{\tau}) + (1-\theta)\Big(\nabla \cdot \tau_{\mathbf{v}}^n(\mathbf{x_p}^{\tau}) + \mathbf{f}^n(\mathbf{x_p}^{\tau})\Big)d\tau
\end{aligned}$$

Implicit part

$$\hat{\mathbf{v}}^{n+1}(\mathbf{x}) - \hat{\hat{\mathbf{v}}}^{n+1}(\mathbf{x}) = \theta\Big(\nabla \cdot \tau_{\mathbf{v}}(\hat{\mathbf{v}}^{n+1}(\mathbf{x})) + \mathbf{f}^{n+1}\Big)\Delta t$$

PRESSURE EQUATION

$$\nabla \cdot \hat{\mathbf{v}}^{n+1}(\mathbf{x}) = \theta \Delta t \nabla^2 \delta p(\mathbf{x})$$

CORRECTOR

$$\mathbf{v}^{n+1}(\mathbf{x_p}) - \hat{\mathbf{v}}^{n+1}(\mathbf{x_p}) = -\theta \Delta t \nabla \delta p(\mathbf{x_p}) \tag{17}$$

It should be noted that for the whole procedure of Lagrangian formulation two coordinates are used, one for the particles ($\mathbf{x_p}$) and the other for the mesh ($\mathbf{x}$). The relation between them is presented in a next section.

2.4 X-IVS [X-IVAS]: Explicit Integration Velocity [Acceleration] Scheme

In the scalar transport equation the velocity field is a given data, known not only for its spatial variation also for its time variation. Therefore it is possible to include this information explicitly in the Lagrangian formulation. Using a high accurate particle tracking integration scheme it is possible to solve simple and complex pathlines normally present in fluid flows (Figs. 1 and 2).

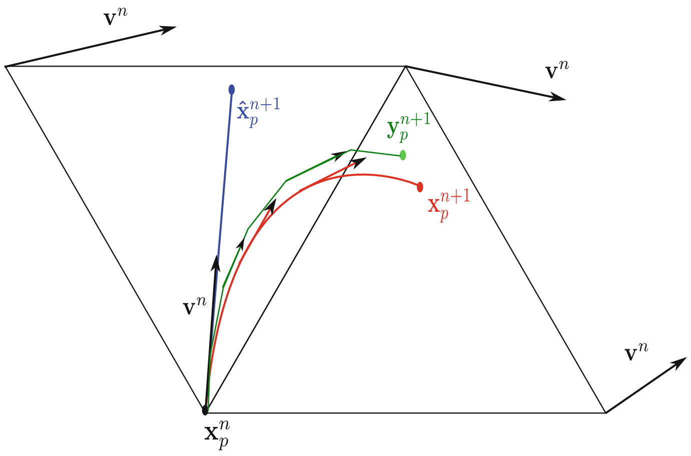


Fig. 1 Time integration

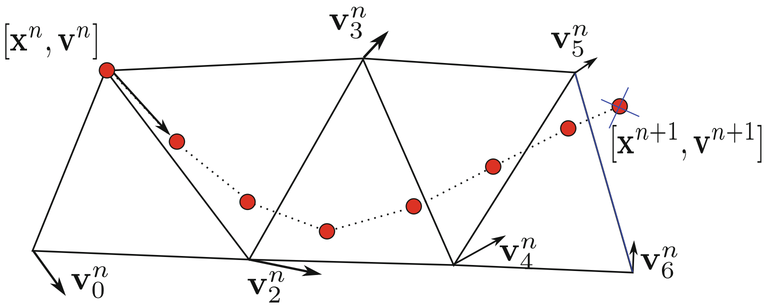


Fig. 2 Streamline integration updating the particle position and state

$$\mathbf{x_p}^{n+1} = \mathbf{x_p}^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^\tau(\mathbf{x_p}^\tau) d\tau \tag{18}$$

- Real trajectory:

$$\mathbf{x_p}^{n+1} = \mathbf{x_p}^n + \int_{n}^{n+1} \mathbf{v}^\tau(\mathbf{x_p}^\tau)\, d\tau. \tag{19}$$

- Simple approximation:

$$\widehat{\mathbf{x_p}}^{n+1} = \mathbf{x_p}^n + \mathbf{v}^n(\mathbf{x_p}^n)\Delta t \tag{20}$$

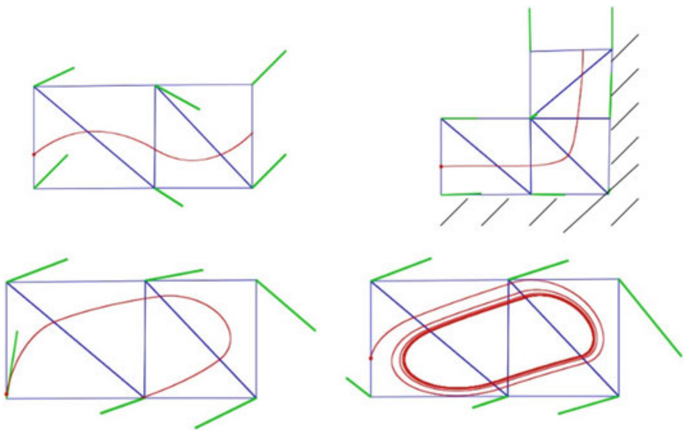


Fig. 3 Streamline integration updating the particle position and state

- Streamlines approximation:

$$\mathbf{y}_p^{n+1} = \mathbf{x_p}^n + \sum_{i=0}^{N-1} \mathbf{v}^n(\mathbf{y}_p^{n+\frac{i}{N}})\,\delta t \tag{21}$$

2.4.1 Remarks

- Adaptive substep: $\delta t = \frac{\Delta t}{N} \propto Co_{ELE}$. These integration substeps may be replaced by an almost exact integration [18]. However for practical applications a very little difference in accuracy is observed for the former with an extra cost for the later that push the decision on the former.
- For vector systems where normally the velocity field is also an unknown the particle velocity $\mathbf{v_p}$ may also be updated following the same ideas changing the name of the method to **X-IVAS**.
- for fluxes depending on the spatial derivatives (like diffusive) **X-IVAS** method is also applied.
- This novel integration in Lagrangian context allows to a significantly better particle trajectory integration **(X-IVS)** following streamlines resolving more difficult details of the flow with high accuracy. Figure 3 shows some details that may be resolved without a drastically reduction in the time step, as it is normally done by standard Lagrangian integration schemes.
- In general this integration scheme does not suffer for strong time step restrictions caused by the non-linearities present in the flow field (Fig. 3).

As it was mentioned for the unknown field, the temperature, only known for the old time step t^n, an approximation to the future time step should be done using the **X-IVAS** method.

$$T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^{n}, t^{n}) = \int_{t^n}^{t^{n+1}} Q(\mathbf{x_p}^{\tau}, \tau)d\tau + \int_{t^n}^{t^{n+1}} \Big(\nabla \cdot (\alpha \nabla T(\mathbf{x_p}^{\tau}, \tau))\Big)d\tau \tag{22}$$

In the *PFEM* method the last term at the right hand side is approximated in the following form:

$$\begin{aligned} T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^{n}, t^{n}) \approx & \int_{t^n}^{t^{n+1}} Q(\mathbf{x_p}^{\tau}, \tau)d\tau \\ & + \underbrace{(1-\theta) \int_{t^n}^{t^{n+1}} \nabla \cdot (\alpha \nabla T(\mathbf{x_p}^{\tau}, t^{n}))d\tau}_{explicit} \\ & + \underbrace{\theta \nabla \cdot (\alpha \nabla T(\mathbf{x_p}^{n+1}, t^{n+1}))\Delta t}_{implicit} \end{aligned} \tag{23}$$

The last integration is only one possibility to choose among others, the explicit part is solved simultaneously with the particle pathline computation, while the implicit one is solved using the final position of the particles. However other choices may be done in order to improve this computation, that for brevity reasons are not included in this paper.

Comparing (12) with (23) the main difference between both may be written as:

$$\begin{aligned} \Delta_{EUL \to LAG} = & \; T(\mathbf{x_p}^{n+1}, t^{n+1}) - T(\mathbf{x_p}^{n}, t^{n}) \\ & - \Big(T^{n+1}(\mathbf{x}) - T^{n}(\mathbf{x}) + \theta \mathbf{v}(\mathbf{x}, t^{n+1}) \cdot \nabla T(\mathbf{x}, t^{n+1})\Delta t \\ & + (1-\theta)\mathbf{v}(\mathbf{x}, t^{n}) \cdot \nabla T(\mathbf{x}, t^{n})\Delta t\Big) \end{aligned} \tag{24}$$

This difference is due to the error produced by the transformation between both frames, an Eulerian or fixed one and the Lagrangian or mobile one. This difference should tend to zero when the time step goes to zero. However, for large time steps normally needed to speed up the computation, the fact of evaluating the velocity field placed on a fixed position (**x**) for Eulerian formulation in two different time intervals may introduce large errors. Moreover the spatial stabilization needed for advection dominant problems introduce also some extra errors that tends to dissipate the solution much more than the physics, specially at large time steps.

2.5 A Simple Coupled System, Solving the Coupled Flow Field and a Passive Scalar

One of the main purposes of this development is its application to solve coupled problems where the flow and several other fields are solved simultaneously with some sort of interaction. For brevity a simple case is here presented. It deals with the coupling of an incompressible viscous flow with a scalar transport like temperature using the Boussinesq approach.

2.5.1 Physical Equations to Solve

- Navier-Stokes Equation System for Newtonian and incompressible fluids:

$$\begin{aligned} \rho\frac{D\mathbf{v}}{Dt} &= -\nabla p + \nabla\cdot\Big(\mu(\nabla\mathbf{v}^T + \nabla\mathbf{v})\Big) + \mathbf{f} \\ \nabla\cdot\mathbf{v} &= 0 \end{aligned} \tag{25}$$

- Scalar-Transport Equations:

$$\frac{D\phi_j}{Dt} = \nabla\cdot(\alpha_j\ \nabla\phi_j) + Q_j \quad \forall j \in (1 : N_{\mathit{fields}}) \tag{26}$$

- Example of coupling. For $\phi = T \Rightarrow$ Boussinesq approximation:

$$\mathbf{f} = \rho\mathbf{g}\beta(\phi - \phi_c) \tag{27}$$

2.5.2 Discretization

The key of the PFEM algorithm is transporting the information with particles following the streamlines. Although the field $\mathbf{v_p}$ is not stationary, streamlines are taken as stationary on each time step ($\mathbf{v_p}^n$), the particle position follows that field and the particle state is updated by the rate of change determined by the balance equation.

$$\mathbf{x_p}^{n+1} = \mathbf{x_p}^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x_p}^\tau)\, d\tau \tag{28a}$$

$$\mathbf{v_p}^{n+1} = \mathbf{v_p}^n + \int_{t^n}^{t^{n+1}} (\mathbf{a}^n(\mathbf{x_p}^\tau) + \mathbf{f}^\tau)\, d\tau \tag{28b}$$

$$\phi_p^{n+1} = \phi_p^n + \int_{t^n}^{t^{n+1}} (\mathbf{g}^n(\mathbf{x_p}^\tau) + Q^\tau)\, d\tau \tag{28c}$$

where $\mathbf{a}^n = -\nabla p^n + \nabla \cdot \Big(\mu(\nabla^T \mathbf{v}^n + \nabla \mathbf{v}^n)\Big)$ and $\mathbf{g}^n = \nabla \cdot (\alpha\, \nabla \phi^n)$, which are nodal variables.

3 Two Examples to Show the Benefits of the New Ideas

The following two examples serve as the starting point of new ideas behind high accurate and stable convective transport equations.

- Pure advection of a passive scalar field.
- Inviscid transport of a vortex.

The first example was the proof in showing the advantages of using Lagrangian formulations when a pure advective problem is between hands. It is a Gaussian hill profile imposed as an initial condition advected by a pure rotation motion. For this problem the profile shape and its amplitude should be conserved all the way. The second one is one extension of the first example applied to a vector system like Navier-Stokes equations. It consists of an initial vortex that is transported in an inviscid flow. For this problem the intensity of the vortex should be conserved.

3.1 Pure Advection of a Passive Scalar Field

This well known problem normally serves as a benchmark for the spatial stability of Eulerian numerical schemes. The first scope in this benchmark is to show that no spurious oscillations appear and the second one focus on minimizing the numerical diffusion introduced by the stabilization schemes. Also the time integration numerical scheme is responsible for extra numerical dissipation, being the first order explicit ($\theta = 0$) or implicit ($\theta = 1$) schemes not recommended for their high dissipation. Crank-Nicholson ($\theta = \frac{1}{2}$) is preferred in this sense. However looking at the solution it is always present a reduction in the original amplitude that may be improved only reducing the mesh size and the time step.

In [10] several Figs. 9.1–9.4 are shown where it is possible to realize how the amplitude is drastically reduced using large Courant numbers with Eulerian formulations. Even though the spatial stabilization is reduced to a minimum and the time integration is chosen as second order the numerical dissipation is highly noticeable. Only reducing the Courant number with finer meshes it is possible to reduce it but never annihilate.

On the other hand Lagrangian formulations are better positioned for this kind of problems if only the amplitude is observed. This remains exactly constant all the way regardless the Courant number. However with standard first order integration, the problem arises in the definition of the pathlines that are shifted inwards or outwards depending on the explicit or implicit character of the time integration. Only with second order time integration is possible to reduce this pathology but some sort of iteration is needed. See Fig. 9.7 in [10].

Using the X-IVAS integration is possible to fix both errors simultaneously producing a high accurate solution regardless the Courant number. This is also shown in Fig. 9.7 at left in [10].

A final remark about this important result achieved on such a basic example, showing the great capabilities of Lagrangian formulations over Eulerian ones for convective transport, is related to some more accurate Eulerian schemes that are currently being published for transporting signals without causing spurious diffusion. Called as High Resolution Schemes [14, 16], these numerical methods have a robust control to suppress the wiggles with the minimum amount of numerical dissipation. According to the Godunov theorem [16] this is only possible in a nonlinear way. Even though this way circumvents the drawbacks it is important to realize that Lagrangian formulations achieved the same or better results without doing nothing special saving the extra cost normally experienced by such schemes.

3.2 Inviscid Transport of a Vortex

Having found the good benefits of Lagrangian formulations for transporting scalar fields in a stable and accurate way the following step was to extend the same to vector systems. Here the incompresible viscous fluid flow model was taken. The equivalent example in this context is the transport of a vortex in an inviscid flow. It is well known that looking at the hyperbolic part of the whole system, neglecting the diffusion and not considering the role of the pressure, the problem looks like the convection of vorticity waves. If a vortex ring is imposed as initial condition, neglecting the viscosity with slip boundary conditions on the walls, the vortex should conserve its kinetic energy as much as possible.

Figure 4 shows how the Eulerian and Lagrangian formulation transports this vorticity. For both formulations the mesh is kept fixed and the simulation had run with the same time step, with a high enough Courant number in order to highlight the performance and precision comparison. It is possible to conclude that the Lagrangian formulation is more energy conservative than the Eulerian counterparts with the evidence that the vortex may be transported much better. This example confirms the advantages of Lagrangian formulation respect to Eulerian ones not only for advective transport of scalar fields, also for non linear vector fields (Fig. 4).

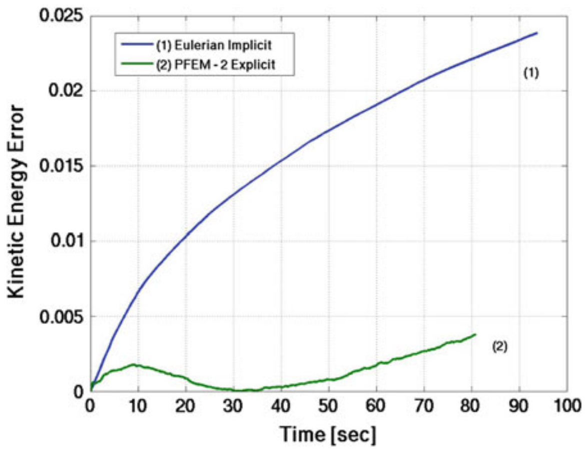


Fig. 4 Inviscid vortex ring dynamics. Conservation of momentum

4 Moving and Fixed Mesh PFEM Versions

The natural evolution of *PFEM* method employed only one mesh built from a cloud of points defined by the moving particle position. There was a one to one relation between mesh nodes and particles. At each time step the original *PFEM* method moved the nodes following the updated particle positions as long as the mesh was not deformed in such a way that an invalid grid appears. Remeshing was only used when the deformation of the mesh was so large that the time step suffered a drastic reduction making the computation too much expensive. At that times, the remeshing was by-passed at extreme for cpu times reasons. Summarizing the stability of the original *PFEM* was mainly affected by:

- critical time step for explicit advective terms ($Co < O(1)$).
- critical time step for explicit diffusive terms ($Fo < O(1)$).
- critical time step for the deforming mesh limited by the inversion of some elements in the mesh (invalid)
- non linearities

The sequence of the problems above defined may be summarized as:

- To solve only the passive scalar transport Eq. (28a) and (28c) are used. If you want to transport N passive scalars, you only have to solve (28c) for each one of the N variables.
- To solve the Navier Stokes equation system, (28a) and (28b) are used, and p must be calculated. A typical Fractional Step Method is used to solve the coupling between the pressure and the velocity (see Eq. 16).
- To solve the thermal and fluid flow coupling (natural convection) (28a), (28c) and (28b) are used and a constitutive law for the buoyancy term should be added (Boussinesq approach)

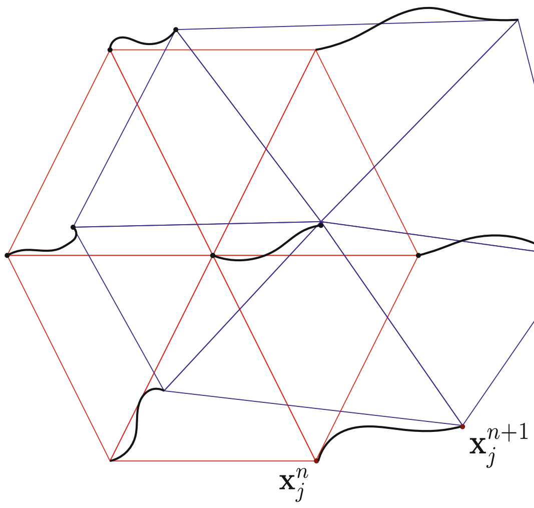


Fig. 5 *PFEM* —moving mesh version

All these steps need a mesh update and again the remeshing returns to the scenario. During the last years a lot of progress was done in terms of more efficient mesh generation and regeneration exploiting the parallelism, making the remeshing affordable. A permanent remeshing circumvents the severe time step restriction produced by the invalid mesh condition. In this sense the *PFEM* had experienced a significant progress increasing the time steps with stable solutions.

- It is necessary to update the mesh states with the particles states. There are two approaches which have generated two versions of the method:
  - Remesh the geometry with new particles positions (particles↔nodes): **PFEM Mobile Mesh**
  - Project states from particles to nodes, preserving the mesh as fixed: **PFEM Fixed Mesh**

The Mobile Mesh version has the following features:

- a 1-1 relation between Particles and Nodes.
- Remeshing at each time step.
- Need permanent assembling, profiling and solving of the algebraic linear system.

Figure 5 shows how the particle motion change the mesh definition at each time step.

The first tests showed very good features in terms of stability and accuracy getting a drastic reduction in the cpu times involved for solving some benchmarks compared with the original version of *PFEM*. This improved performance, added to the possibility of using very large time steps were the first evidences that the permanent

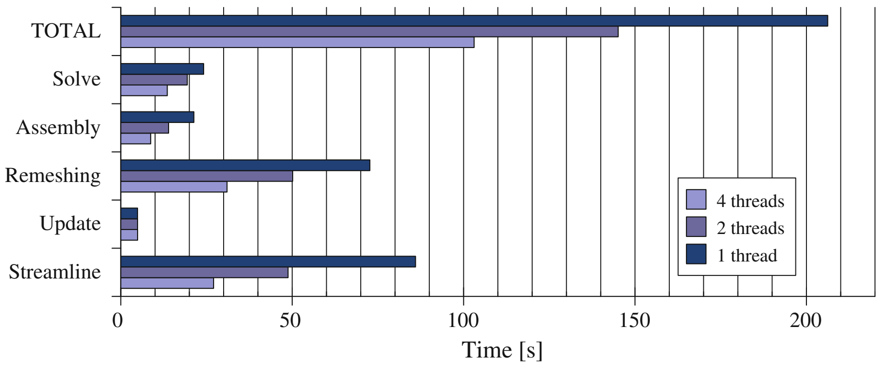


Fig. 6 *PFEM* —moving mesh version—profiling

remeshing using X-IVAS integration were two important numerical ingredients for exploiting the good features evidenced by the Lagrangian formulation.

Even though moving mesh *PFEM* version has several advantages against its Eulerian counterpart, it has some limitations in terms of efficiency. Mainly the permanent remeshing and the assemble/solving of implicit problems limit its scalability The Fig. 6 shows a profiling of the moving mesh version of the *PFEM*.

As it is evident from this figure most of the time is spent in remeshing, assembling and solving the implicit linear systems, with a performance similar to mesh based methods because the particle update only consumes a small part of the whole cost.

In order to reduce the computational cost added by these two stages a novel idea was presented: *the Fixed Mesh Version* of *PFEM* .

This new method combines particles with a background mesh. Particles carry the information along the whole process using the mesh only for secondary computations, those needed to update the particle position and their states. It is normally understood as an hybrid method or dual method where particles act like the master in the computation and the mesh is the slave. The idea does not only avoid the permanent remeshing, using a fixed background mesh it is possible to integrate all the implicit part of the computations with an important and favorable impact on the computational efficiency, the possibility of re-using the matrix profile and for linear diffusion problems also its factorization.

This fact may be exploited only by Lagrangian formulation because the Eulerian counterpart always has the convective term proportional to the changing unknown velocity inside the system matrix to be inverted. Therefore it is not possible to take advantage of it.

The fixed mesh version has the following features :

- Particles cloud over a Fixed Background Mesh.
- No need remeshing.
- It needs Projections and Interpolations between particles and mesh nodes.
- It needs only one LU or Cholesky factorization for implicit calculations.

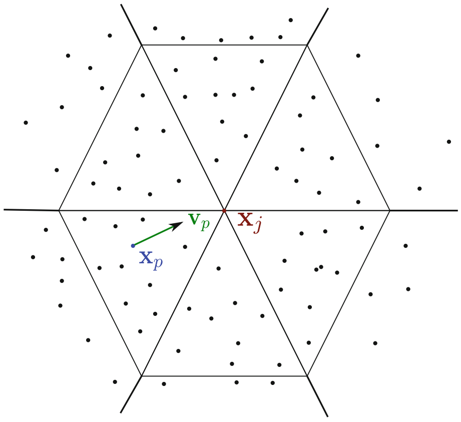


Fig. 7 *PFEM* —fixed mesh version

For constant coefficients problems an initial factorization may be built, reusing it along the whole computation. In this way it is possible to reduce the number of iterations of the preconditioned conjugate gradient used for solving the symmetric and positive definite linear systems involved in the computation (Poisson for the pressure and the heat equation for the implicit part of the diffusive terms ($\theta > 0$)).

However, the fixed mesh version has some cons, specially that concerned with the projection error. This operation allows to reconstruct some fields transported by the particles in a very accurate way on a mesh where the spatial derivatives should be computed. The better the reconstruction the less the error in the computation of the fluxes depending on the spatial derivatives (Fig. 7).

Several numerical experiments have shown that the advection part of the computation is governed by particles and the diffusive part is governed by the mesh. Therefore increasing the number of particles allows to transport very complex initial conditions or represents complex fields produced by its time evolution under complex flow fields. On the other hand refining the mesh allows to improve the diffusive fluxes computation. But this is not the only way to get it, higher order reconstruction is also an alternative to explore in the future.

First order reconstruction may become inconsistent, i.e. refining in the number of particles may produce higher diffusive flux errors. To avoid it one possibility could be to split the elements in subdomains, one around each element node, using only that division to project particles values on the node associated with that subelement.

Instead of using all the particles belonging to the elements connected to a given node, only those particles belonging to the subelements connected to the node are chosen. This subdivision also serve to seed particles when that subelement is void of particles. In a next section some details about it are presented.

In terms of efficiency the Fig. 8 shows as the fixed mesh version has changed its profiling, now with the most important computational cost component concentrated in the particle computation, typical of a Lagrangian formulation.

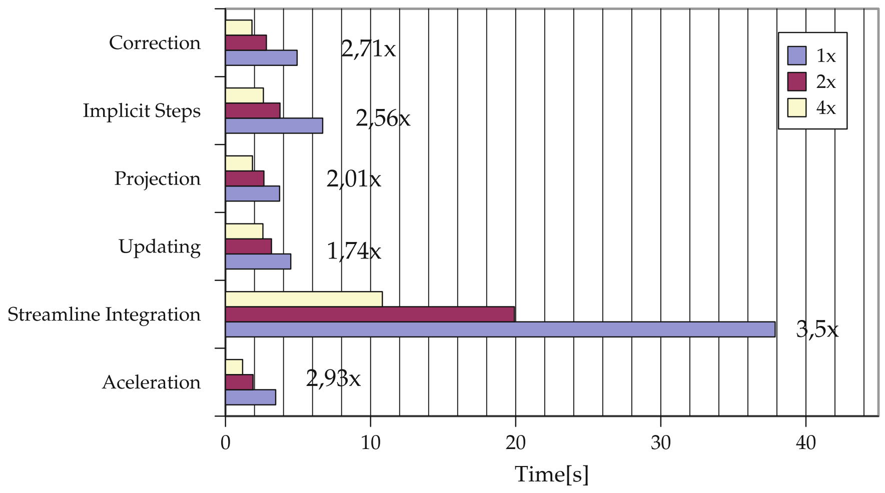


Fig. 8 *PFEM* —fixed mesh version—profiling. *Case* flux around a cylinder 2D—CPU: Intel i7-2600k (4 cores)

Being particle computations cheaper than mesh computations it put the *PFEM* method in a very good condition for high performance computing stuffs, specially for scalability.

Finally this section ends with some review of the two algorithms that were firstly developed in the context of the *PFEM*, one for the scalar transport and the other for the incompressible viscous flow.

Algorithm 1 - Time Step PFEM Scalar Transport

```
1. Calculate scalar change rate on the nodes like a FEM:
   $\int_\Omega \boldsymbol{N}\, \mathbf{g}^n \, d\Omega = -\int_\Omega \nabla \boldsymbol{N}\, \alpha \nabla \phi^n \, d\Omega + \int_\Gamma \boldsymbol{N}\, \nabla \phi^n \cdot \eta \, d\Gamma$
2. Evaluate new particles position and state following the streamlines:
   $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x}_p^\tau)\, d\tau$
   $\phi_p^{n+1} = \phi_p^n + \int_{t^n}^{t^{n+1}} \mathbf{g}^n(\mathbf{x}_p^{n+\tau}) + Q^{n+\tau}\, d\tau$
3. Update particles inventory
4. Project state to the mesh:
   $\phi_j^{n+1} = \boldsymbol{\pi}(\phi_p^{n+1})$
```

Algorithm 2 - Time Step PFEM Incompressible Flow

```
1. Calculate acceleration on the nodes like a FEM:
   $\int_\Omega \boldsymbol{N}\, \nabla \cdot \tau_v \, d\Omega = -\int_\Omega \nabla \boldsymbol{N} \cdot (\mu \nabla \mathbf{v}^n)\, d\Omega + \int_\Gamma \boldsymbol{N}\, \nabla \mathbf{v}^n \cdot \eta \, d\Gamma$
   $\int_\Omega \boldsymbol{N}\, \nabla p^n \, d\Omega = -\int_\Omega \nabla \boldsymbol{N}\, p^n \, d\Omega + \int_\Gamma \boldsymbol{N}\, p^n \cdot \eta \, d\Gamma$
   $\mathbf{a}^n = \nabla \cdot \sigma = -\nabla p^n + \nabla \cdot \tau_v$
2. Evaluate new particles position and state following the streamlines:
   $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x}_p^\tau)\, d\tau$
   $\hat{\mathbf{v}}_p^{n+1} = \mathbf{v}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{a}^n(\mathbf{x}_p^{n+\tau}) + \mathbf{f}^{n+\tau}\, d\tau$
3. Update particles inventory
4. Project state to the mesh:
   $\mathbf{v}_j^{n+1} = \boldsymbol{\pi}(\mathbf{v}_p^{n+1})$
5. Find the pressure value solving the Poisson equation system using FEM:
   $\rho \nabla \cdot \hat{\mathbf{v}}_j^{n+1} = \Delta_t \Delta[\delta p^{n+1}]$
6. Update the velocity value with the new pressure:
   $\rho \mathbf{v}_j^{n+1} = \rho \hat{\mathbf{v}}_j^{n+1} - \Delta_t (\nabla p^{n+1} - \nabla p^n)$
   $\rho \mathbf{v}_p^{n+1} = \rho \hat{\mathbf{v}}_p^{n+1} - \Delta_t \boldsymbol{\pi}^{-1} (\nabla p^{n+1} - \nabla p^n)$
```

It is important to realize that in the fixed mesh version of *PFEM* there is an operation called projection that deserves some attention. In a next section some comments about it will be presented.

5 Particle Inventory Managment

As mentioned before, in the *PFEM* algorithm, after the streamline integration the state variables are placed on the particles. Both, for reasons of incompressibility (pressure) as for the treatment of the diffusion (viscous stress tensor) require that the information should be located on the grid. While for the Mobile Mesh version the mesh is done with the particles themselves, in the Fixed Mesh approach particles and grid are decoupled and a projection $\boldsymbol{\pi}$ from particle states to nodal states should be done.

5.1 Projection Algorithms

Different approaches are available to perform projections, for example SPH or MLS (Moving Least Square) techniques could be used for the interpolation, as well as weights based on the position on the top of the underlying mesh. A brief review of the actual techniques for projection in PFEM are presented (the equations presented are for scalar projection. They are also valid for each component of a vector state variable):

- Mean weighted by Shape Function (P-1):

$$\phi_j = \frac{\sum_{p=1}^{P} \mathbf{N}_j(\mathbf{x}_p)\phi_p}{\sum_{p=1}^{P} \mathbf{N}_j(\mathbf{x}_p)} \tag{29}$$

  where P is the number of particles inside a certain region around the node j.
- Mean weighted by Distance (P-2):

$$\phi_j = \frac{\sum_{p=1}^{P} ||\mathbf{x}_p - \mathbf{x}_j||_2\phi_p}{\sum_{p=1}^{P} ||\mathbf{x}_p - \mathbf{x}_j||_2} \tag{30}$$

  with the same definition of P as just above.
- Weighted Polynomial Least Squares (P-3):

$$\phi_j = h_\theta(\mathbf{x}_j) = \boldsymbol{\theta}^T \mathbf{P}(\mathbf{x}_j) \tag{31}$$

  where:
  $\mathbf{P} = [1\, x\, x^2 \ldots]$ on 1d, $\mathbf{P} = [1\, x\, y\, xy\, x^2\, y^2 \ldots]$ on 2d (truncating at the polynomial order required) and $\boldsymbol{\theta} = (\mathbf{X}^T\ \mathbf{W}\ \mathbf{X})^{-1}(\mathbf{X}^T\ \mathbf{W}\ \mathbf{y})$. It must be noted that to invert the matrix in the calculation of $\boldsymbol{\theta}$, it is required $P >= n$, where n is the number of terms used on P.

5.2 Particle Seeding

For accuracy reasons each one of the presented projection methods require a certain number of particles in certain region near to each node. Some considerations must be taken into account: the *region around the node* must be defined precisely, and is not assured that there were particles inside each region (specially when high Co numbers are used). Then, new particles must be created at these empty regions. In this section several algorithms attending this issue are presented.

The first approach (S-1), used originally in PFEM, consists on setting the states to a new particle interpolating from the nodal states at the previous time step n:

$$\phi_p^{n+1}(\mathbf{x}_p^{n+1}) = \sum_j \lambda_j(\mathbf{x}_p^{n+1})\phi_j^n \tag{32}$$

being λ_j the area coordinates of the particle in the element. Other algorithm (S-2) searches the state following the streamline but in backward direction, thinking in

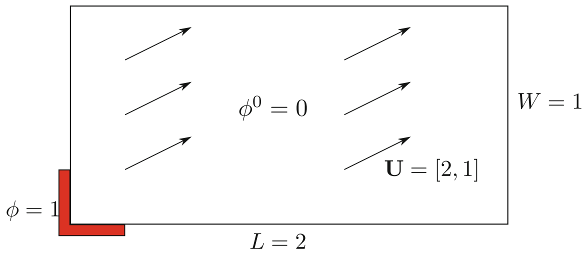


Fig. 9 Pure-convection step problem

finding the particle location that, if it had existed at the beginning of the time step, at the end of it would have arrived at the seed position:

$$\phi_p^{n+1}(\mathbf{x}_p^{n+1}) = \phi_p^n(\mathbf{x}_p^n) + \int_0^{\Delta t} \mathbf{g}^n(\mathbf{x}^{n+\alpha})\, d\alpha \tag{33}$$

The utility of the backward integration to search the state of the new particle is shown in the next example: the pure-convection step problem, which is defined in Fig. 9.

A boundary condition with a sharp discontinuity enters the domain transported with a velocity vector field not aligned with the mesh. Figure 10 shows the results, projected on the mesh, achieved when the particles are created using the criterion defined as (S-1) (left) and also when their states are found using backward integration criterion (S-2) (right).

The results show that S-1 criterion fails for this example putting S-2 criterion as a much better selection for seeding particles when it is necessary.

5.2.1 Particle Removing

On other hand, using a lot of particles increases computing times. During the simulation the seeding is frequent and we need to control the amount of particles inside the domain for computational cost reasons. So, the removal action should be defined following some criteria.

Although it is known that particle which leaves the geometry must be deleted, inside the geometry is not clear when particles should be removed and how to do that. Removing particles decreases computing times of the algorithm but also decreases the quality of the solution because it introduces numerical diffusion in an indirect way.

In the first approach (R-1), the particles are not removed unless that two or more of them are in almost the same position. This approach obtained accurate results solving the pure-advective case of the rotating Gaussian signal [18].

A second approach (R-2), consists on requiring minimum number and a maximum number of particles at each sub-element that must be conserved at each time step.

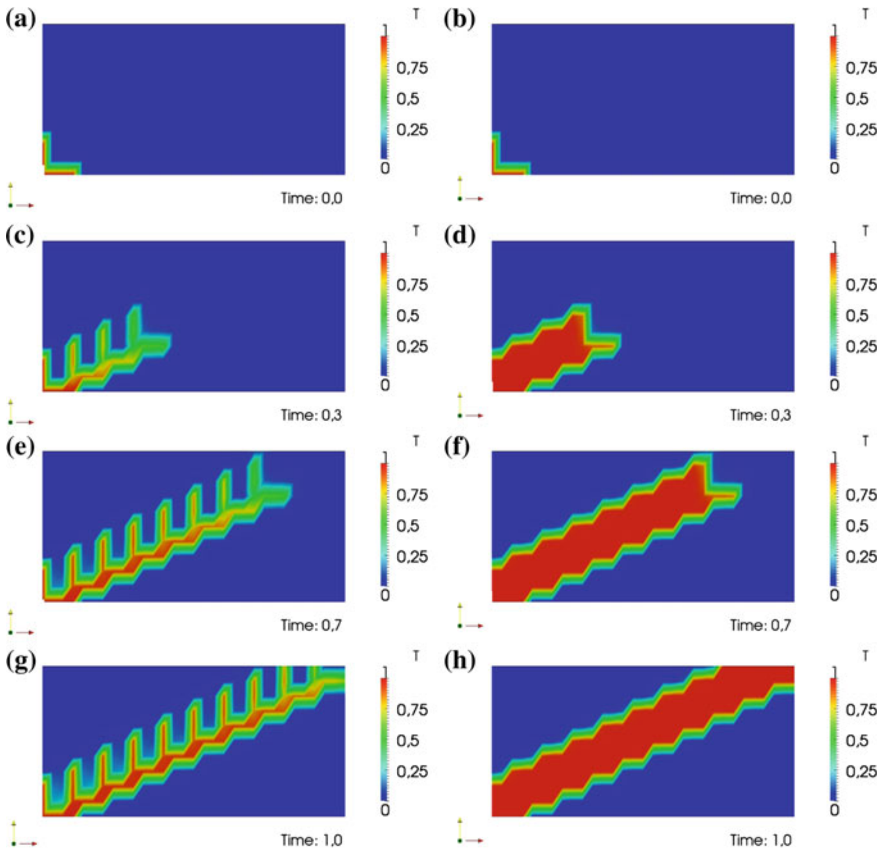


Fig. 10 Pure convection transport. Results for different strategies to new particles states. *Left* S-1. *Right* S-2

The sub-element i is the third parts of the triangle (fourth parts of the tetrahedral in three dimensions) where the area coordinate corresponding to the vertex i is larger than the rest. The idea is to think that if each node has enough number of particles around, the projected state from particles to the node will be accurate. However, as will be demonstrated in the next example, the continuous intrusion to the system could decrease the quality of the solution, specially in scalar problems. This criterion shows good results in solving incompressible flows.

The example of the *rotating Gaussian signal without diffusion* shows that the numerical diffusion is important when a frequent creation and removing of particles is performed following this criterion. The polar mesh (4390 elements) is the same in all cases, the case consists of a Gaussian transported by a rotating flow without diffusion term. Figure 11 compares the value of the maximum through two laps. The best option for this problem is R-1 (in Fig. 1 Tmax old), while R-2 requires a large range ($[min\_subele; max\_subele]$) of particles not to spread: comparing

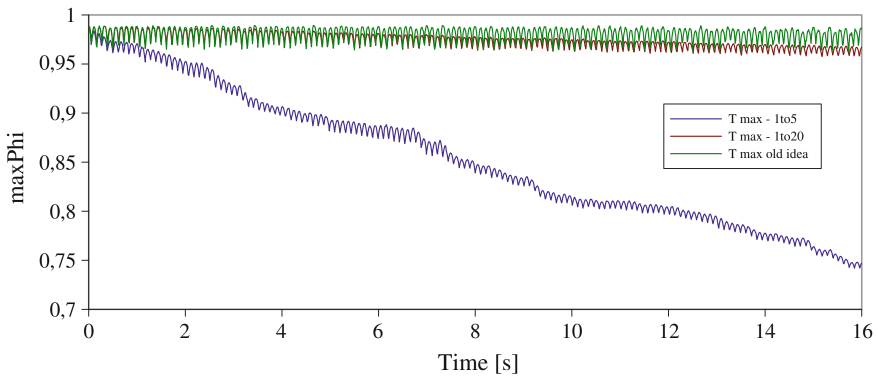


Fig. 11 Rotating Gaussian. Evolution of ϕ_{max} for different update particle techniques

[1; 20] with [1; 5], in the second option the intrusion in the system is greater and the solution decreases its quality.

Regarding the computing times also R-1 is the best, because requires 40 s (mean 40,000 particles), whereas [1; 5] needs 50 s (mean 46,000 particles) and [1; 20] needs 100 s (mean 120,000 particles).

However, R-1 does not work fine for the step's problem (defined in previous subsection), unless it creates new particles using backward integration (criterion S-2). Also, due to the type of projection of the algorithms developed, which searches particles in the sub-elements to send data to nodes, creating new particles in empty elements does not ensure that there will be particles in the region around the node (their sub-elements), so another type of selection of the position of the new particles must be developed.

5.3 Converging into a New Algorithm to Update Particle Inventory

Taking into account the previous discussion, an algorithm to update particle inventory has been implemented, that includes the benefits of the methods discussed previously and follows the principle of not modifying the system unless it be really necessary.

It was mentioned that the condition *empty-element*, except for some particular problems, shows to work not so fine. The main reason is the lack of particles close enough to nodes. So, the new algorithm must not search *empty-elements*, instead of that, it searches *empty-regions*.

The region j is defined as the geometric space where the node j takes particle information to update its state using some projection operator; therefore, if this region is empty, the node does not have enough information to be updated. Once detected an *empty-region*, only one particle is created in exactly the same position of its node and with a state found using backward integration.

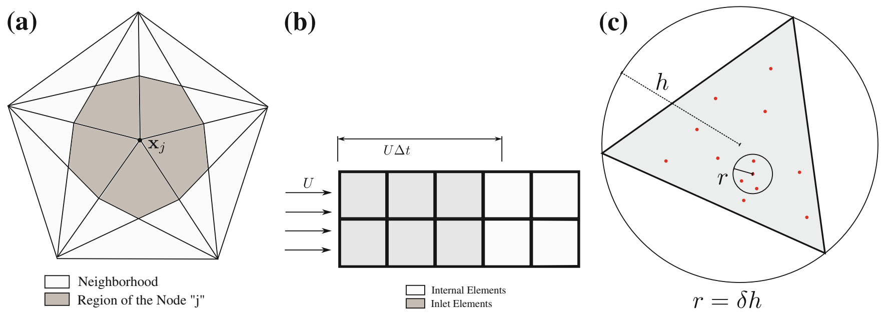


Fig. 12 Graphical scheme to understand the new algorithm proposed

Although to create new particles on the nodes is the least intrusive way to maintain good solution on the nodes, if the problem is not of *confined flow*, the number of particles will decrease while the simulation runs. Typically in the inlet flow boundary the boundary condition is imposed. Then, creating particles in the *Inlet Elements* and doing backward integration to search the states no error is committed, and more than one particles can be created. The position does not have to be on the nodes and its state will not have error. This approach allows to keep approximately the same numbers of particles during the simulation, which preserves the accuracy of the method.

Particle removing is carried out when two or more particles are in a circle (2d) or sphere (3d) with a radius proportional to the size of the element ($r = \delta$h). This approach allows to use different δs over the geometry, being a new tool to control the number of particles. Graphic representation is presented in the Fig. 12c.

Figure 12a shows the neighbor elements and the region belonging to the node j. It must be there at least one particle in the gray zone to have a good projection, else a new particle must be created in the same position of the node and searching its state with backward integration. Figure 12b shows which elements are considered as *inlet elements* and, if they are empty, they must create internal particles.

Finally, this algorithm allows to solve all tests presented in this paper while other approaches have shown to fail: the *Gaussian rigid rotation* and the *step-2d*.

The last example consists on testing this algorithm in the Navier-Stokes solver (*PFEM Fixed Mesh*). The case chosen is the *Flow Around a Cylinder*, because it presents different zones of refinement and patches of inlet and outlet flow. The results are presented in the Fig. 13a and b. Similar accuracy in the amplitude and frequency can be observed, but R-2 obtains better definition of the forces signal, specially for Cd. For more details see [10, 11].

6 Diffusive-Dominant Problems

When the problem is diffusive-dominant, the advantages of the method PFEM are not as clear as in the advective-dominant case. The explicit calculation of the diffusion traditionally used by PFEM is limited by the dimensionaless Fourier number and,

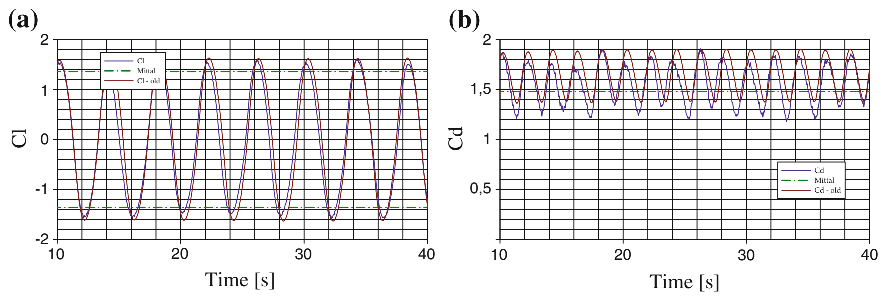


Fig. 13 Lift coefficient and drag coefficient for flow around a cylinder solved using the new updating algorithm and comparing it with R-2 (called *old* in the graphic)

in some particular cases, the temporal change of the transported variables vanishes due to the shape of its own solution. To relax these restrictions, in the Sect. 6.1 a new model to calculate the diffusion is presented. Several tests are presented to confirm improvements in the solution.

6.1 Diffusive Implicit Correction

Simulations solving the diffusive term in an explicit way are restricted by $Fo < 0.5$. This is a strong limitation for the time-step, specially on very refined mesh and in diffusive dominant problems (where $Fo > Co$). Due to explicit PFEM suffers this stability constraint the possibility of enlarging the time step may be lost when the flow locally turns to be diffusive. As we have mentioned normally in the vicinity of bodies some refinement is done to capture boundary layers and flow separations and locally the Fourier number increases. Also, in some particular cases, the temporal change of the transported variables vanishes due to the shape of the own solution: when the time-step is chosen such that the integral of the curvature of the function ϕ_j^n vanishes, the method will not apply diffusion on ϕ, so that the solution will be wrong. This case may be present for traveling waves with diffusion.

A new approach to solve the diffusive term is based on the *theta method* which consists on discretizing the non-stationary variable using a weighted mixture between an explicit prediction and an implicit correction.

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = \theta \mathbf{g}^{n+1} + (1 - \theta)\mathbf{g}^n \tag{34}$$

Doing a first step in explicit way

$$\frac{\hat{\phi}^{n+1} - \phi^n}{\Delta t} = (1 - \theta)\mathbf{g}^n \tag{35}$$

and doing the correction in an implicit way, this is subtracting (35) from (34), it follows

$$\frac{\phi^{n+1} - \hat{\phi}^{n+1}}{\Delta t} = \theta \mathbf{g}^{n+1} \tag{36}$$

Algorithm 3 - Time Step PFEM Scalar Transport Explicit Diffusion - Implicit Correction

```
1. Calculate scalar change rate on the nodes like a FEM:
   $\int_\Omega \boldsymbol{N}\, \mathbf{g}^n \, d\Omega = -\int_\Omega \nabla \boldsymbol{N}\, \alpha \nabla \phi^n \, d\Omega + \int_\Gamma \boldsymbol{N}\, \nabla \phi^n \cdot \eta \, d\Gamma$
2. Evaluate new particles position and state following the streamlines:
   $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x}_p^\tau)\, d\tau$
   $\hat{\phi}_p^{n+1} = \phi_p^n + \int_{t^n}^{t^{n+1}} \mathbf{g}^n(\mathbf{x}_p^\tau) + Q^{n+\tau}\, d\tau$
3. Update particles inventory
4. Project state to the mesh:
   $\hat{\phi}_j^{n+1} = \boldsymbol{\pi}(\hat{\phi}_p^{n+1})$
5. Implicit correction:
   $\phi_j^{n+1} = \hat{\phi}_j^{n+1} + \Delta t\, \theta \mathbf{g}_j^{n+1}.$
6. Interpolate state to particles:
   $\phi_p^{n+1} = \hat{\phi}_p^{n+1} + \boldsymbol{\pi}^{-1}(\delta \phi_j^{n+1}).$
```

An standard FEM formulation is used to compute the implicit correction. This problem may be solved either with the approaches *absolute* (37) or *incremental* (38).

$$[\mathbf{M} + \theta\ \Delta t\ \mathbf{K}]\ \phi^{n+1} = \mathbf{M}\ \hat{\phi}^{n+1} + \Delta t\ \mathbf{F}^{n+\frac{1}{2}} \tag{37}$$

$$[\mathbf{M} + \theta\ \Delta t\ \mathbf{K}]\ \delta\phi^{n+1} = -\theta\ \Delta t\ \mathbf{K}\ \hat{\phi}^{n+1} + \Delta t\ \mathbf{F}^{n+\frac{1}{2}} \tag{38}$$

where $\mathbf{M}$ is a mass matrix, $\mathbf{K}$ is the stiffness matrix and $\mathbf{F}$ is the load vector of a standard FEM discretization. It must be noted that the matrix $[\mathbf{M} + \theta\ \Delta t\ \mathbf{K}]$ for $\mathbf{K} \neq \mathbf{K}(t)$ and $\Delta t = cte$ does not depend on the time, then it can be factorized at the beginning of the computation and used as a preconditioner afterward with a significant cpu-time reduction.

6.2 Analytic Diffusion 1D: Sinusoidal Signal

A diffusive-dominant problem with analytic solution is presented. It is solved using explicit, implicit and semi-implicit schemes for diffusion and using different Fo values.

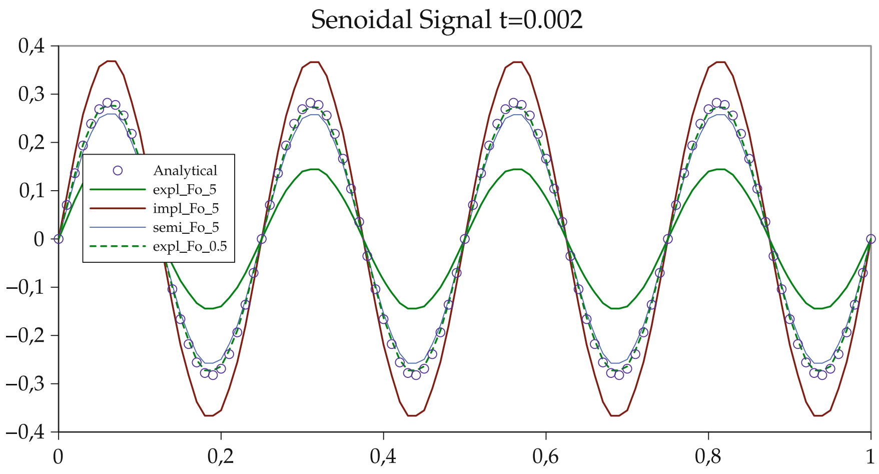


Fig. 14 Comparison between Explicit ($\theta = 0$), Implicit ($\theta = 1$) and Semi-Implicit ($\theta = 0.5$) schemes for diffusion in PFEM with the analytic solution at $t = 0.002$

The problem is:

$$\frac{\partial \phi}{\partial t} = \alpha \frac{\partial^2 \phi}{\partial^2 x} \quad \forall x \in (0, 1) \tag{39}$$

$$\phi(x = 0, t) = \phi(x = L, t) = 0; \quad t > 0 \tag{40}$$

$$\phi(x, t = 0) = \sin(kx); \quad t = 0 \tag{41}$$

with its analytic solution as:

$$\phi(x, t) = \sin(kx)e^{-(2\pi kx)^2 t} \tag{42}$$

where $\alpha = 1$ is chosen, the wave number of the problem is $k = \frac{2\pi}{\lambda} = 4$ and the mesh size is $\Delta x = 0.02$.

Figure 14 shows that using $Fo = 5$ more accurate results are obtained choosing a semi-implicit scheme. While explicit simulations are not accurate with $Fo = 5$, the solution with $Fo = 0.5$ is useful. These results show that temporal integration error in PFEM behaves as usual, i.e. semi-implicit or Crank Nicholson schemes are more accurate than first order explicit or fully implicit schemes. However for semi-implicit solvers we need to give up the idea of an explicit solver with severe consequences on the efficiency. Do not forget that one of the main goals in PFEM development is having a robust (stable) solver that allows to switch between accuracy and efficiency with greater freedom. But, due to the matrix for the implicit part of the computation of the diffusion can be factorized once at the beginning, it is possible to run simulations using big time-steps without loosing accuracy and efficiency. This idea of combining explicit schemes with efficient implicit schemes gives Fixed Mesh

PFEM its stronghold. Efficient implicit schemes means solving linear systems in an iterative way with good preconditioners.

6.3 A Pathological Case: Sinusoidal Signal Travelling

In this section, a pathological case is presented. The explicit calculation of the diffusion updates the state variable with the integral of the second derivative of the variable itself, i.e. the integral of the curvature. When certain conditions are accomplished, that integral vanishes and the explicit diffusion is null generating wrong new states. However, an implicit calculation of the diffusion solves that problem.

The problem consists on a sinusoidal wave transported by a field **v** with a non negligible diffusive term. The idea is to force numerical and physical parameters searching that the integral of the curvature of the function vanishes at each time-step.

If the length traveled by a particle is multiple of the length wave of the signal ($U\Delta t = m\lambda$), then $x_p^{n+1} = x_p^n + m\lambda$, hence, the rate of change of the variable (its curvature κ) will be null because

$$\kappa = \frac{d^2\phi}{dx^2} = \frac{d^2}{dx^2}[\sin(\frac{2\pi}{\lambda}x)] = C\ \sin(\frac{2\pi}{\lambda}x) = \mathbf{g}$$

and $\int_{x^n}^{x^{n+1}} \mathbf{g}\, dx = 0$.

This pathological situation has a very low probability and only is present in Lagrangian formulations where advection and diffusion are weakly coupled.

The problem to solve consists on:

$$\frac{\partial\phi}{\partial t} + U\frac{\partial\phi}{\partial x} = \alpha\frac{\partial^2\phi}{\partial^2 x} \quad \forall x \in (0,\infty) \tag{43}$$

$$\phi(x=0,t) = \sin(\omega t) \quad t > 0 \tag{44}$$

$$\phi(x,t=0) = \sin(\frac{2\pi}{\lambda}x) \quad t = 0 \tag{45}$$

where the advection and the diffusion can be analytically solved in an uncoupled way, allowing to determinate the decay of the signal.

$$\phi(x,t) = \sin(\frac{2\pi}{\lambda}\ [x - (x_0 + Ut)])\ e^{-(\frac{2\pi}{\lambda})^2 t} \tag{46}$$

Using the parameters:

- $U = 5{,}000$
- $\alpha = 1$
- $L_x = 10$

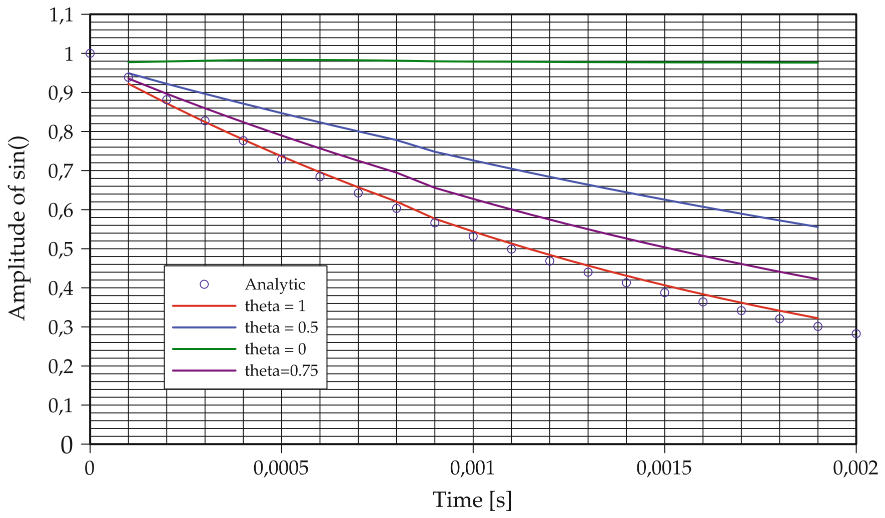


Fig. 15 Temporal evolution of sinusoidal amplitude

- $\lambda = 0.25$
- $\Delta x = 0.025$
- $\Delta t = 0.0001$ ($Fo = 0.16$).

In Fig. 15 results obtained with different values of θ are presented comparing with analytic decay. The most accurate simulation is using $\theta = 1$, using $\theta = 0$ decay is not observed and with other values for intermediate θ solutions are obtained. Finally, a corrective step of an erroneous explicit prediction does not ensure accurate results due to the bad performance of explicit schemes for this very special case.

It must be emphasized that the presented case rarely appears in non-academic problems, but it allows to demonstrate another reason to choose an implicit calculation of the diffusion instead of an explicit.

6.4 Implicit Calculation of the Viscous Diffusion

The theta method can also be adopted to calculate the viscous effects on incompressible flow problems. Again, this strategy allows to extend the maximum time-step without the limitation of the Fourier number. The expressions are similar to the scalar case presented in Eqs. (34)–(36), but replacing ϕ with $\mathbf{v}$ and $\mathbf{g}$ with $\nabla \cdot \tau_{\mathbf{v}}$ where $\tau_{\mathbf{v}} = \mu(\nabla \mathbf{v} + \nabla \mathbf{v}^T)$.

Finally, the algorithm for the implicit correction of the viscous stress tensor is presented in the following algorithm:

Algorithm 4 - Time Step PFEM Incompressible Flow with Implicit Correction of the viscous diffusion.

```
1. Calculate acceleration on the nodes like a FEM:
   $\int_\Omega \boldsymbol{N}\ \nabla \cdot \tau_v\ d\Omega = -\int_\Omega \nabla \boldsymbol{N} \cdot (\mu \nabla \mathbf{v}^n)\ d\Omega + \int_\Gamma \boldsymbol{N}\ \nabla \mathbf{v}^n \cdot \eta\ d\Gamma$
   $\int_\Omega \boldsymbol{N}\ \nabla p^n\ d\Omega = -\int_\Omega \nabla \boldsymbol{N}\ p^n\ d\Omega + \int_\Gamma \boldsymbol{N}\ p^n \cdot \eta\ d\Gamma$
   $\mathbf{a}^n = \nabla \cdot \sigma = -\nabla p^n + \nabla \cdot \tau_v$
2. Evaluate new particles position and state following the streamlines:
   $\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{v}^n(\mathbf{x}_p^\tau)\ d\tau$
   $\hat{\hat{\mathbf{v}}}_p^{n+1} = \mathbf{v}_p^n + \int_{t^n}^{t^{n+1}} \mathbf{a}^n(\mathbf{x}_p^\tau) + \mathbf{f}^{n+\tau}\ d\tau$
3. Update particles inventory
4. Project state to the mesh:
   $\hat{\hat{\mathbf{v}}}_j^{n+1} = \boldsymbol{\pi}(\hat{\hat{\mathbf{v}}}_p^{n+1})$
5. Implicit correction:
   $\hat{\mathbf{v}}_j^{n+1} = \hat{\hat{\mathbf{v}}}_j^{n+1} + \Delta t\ \theta \nabla \cdot \tau_j^{n+1}.$
6. Find the pressure value solving the Poisson equation system using FEM:
   $\rho \nabla \cdot \hat{\mathbf{v}}_j^{n+1} = \Delta_t \Delta[\delta p^{n+1}]$
7. Update the velocity value with the new pressure:
   $\rho \mathbf{v}_j^{n+1} = \rho \hat{\mathbf{v}}_j^{n+1} - \Delta_t(\nabla p^{n+1} - \nabla p^n)$
   $\rho \mathbf{v}_p^{n+1} = \rho \hat{\mathbf{v}}_p^{n+1} - \Delta_t \boldsymbol{\pi}^{-1}(\nabla p^{n+1} - \nabla p^n)$
```

The equation system for the implicit correction of the viscous diffusion can be solved using the same strategy as presented in (37) or (38). Also, for $\mu \neq \mu(t)$ and $\Delta t = cte$ the matrix does not depend on the time, then it can be factorized only once.

6.5 A Simple Test for a Diffusive Dominant Problem. The Mesh Size and the Time Step Dependency

6.5.1 Advective-Diffusive Transport of a Gaussian Hill

The transport of a Gaussian Hill problem was used to demonstrate the goodness of PFEM method to solve a scalar transport problem [18]. This case also made evident the pathology that explicit Eulerian approaches suffer in solving a pure advective transport problem with $CFL > 1$. The problem consists of a Gaussian hill signal used as initial condition transported with physical diffusion. The velocity field is a flow rotating around the center of a square domain. The Gaussian signal is displaced from the center of the domain at a certain radius and its shape makes the transported signal have a non-zero value in a limited region of the domain initially. The signal should be transported following circular path lines. Figure 16 shows the problem definition.

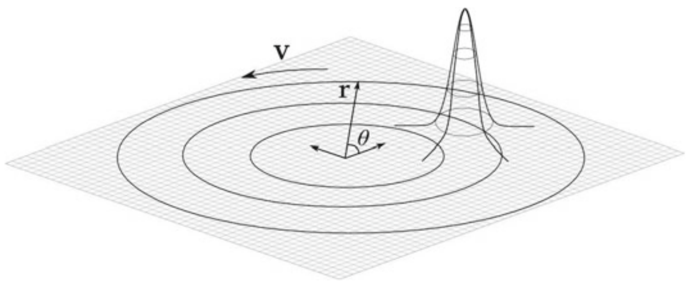


Fig. 16 Initial temperature distribution for the advective-diffusive transport problem

6.5.2 Problem Parameter Definition

This problem is taken from Donea and Huerta [4]. The initial condition is:

$$\phi(\mathbf{x},0) = \begin{cases} \frac{1}{4}(1+cos(\pi X))(1+cos(\pi Y)) & if\ X^2+Y^2 \le 1 \\ 0 & otherwise \end{cases} \tag{47}$$

where $\mathbf{X} = (\mathbf{x} - \mathbf{x}_0)/\sigma$ with a boundary condition $\phi = 0$ for all the nodes lying on the boundary. The initial position of the center of the signal and its radius are $\mathbf{x}_0$ and σ respectively. In this example $\mathbf{x}_0 = (\frac{1}{6}, \frac{1}{6})$ and $\sigma = 0.2$ are taken. The velocity field corresponds to a rigid rotation with angular velocity $\omega = 2$, therefore $\mathbf{v}(\mathbf{x}) = (-\omega y, \omega x)$. The diffusivity chosen is $\alpha = 0.0001$.

Three different meshes were employed, all defined over a unit square $[-\frac{1}{2}, -\frac{1}{2}] \times [\frac{1}{2}, \frac{1}{2}]$. The coarse mesh called $M1$, with 30×30 quadrangular elements split in triangles. Another finer called $M2$, with 100×100 and finally the finest mesh called ($M3$), with 500×500, in order to define a reference solution for comparison playing the role of an almost exact solution.

6.5.3 FEM and PFEM Simulations

Next the results are compared. They were obtained using:

- an Eulerian FEM+SUPG code using different time integration θ schemes, where $\theta = 1$ is the first order implicit Backward-Euler and $\theta = 0.5$ is the second order Crank-Nicholson. They are labeled as `FEM-1order` and `FEM-2order` respectively.
- a Lagrangian code called PFEM using different number of particles per element seeded at the initial time step, (3,9,15,24), labeled as `PFEM-3p`, `PFEM-9p`, `PFEM-15p` and `PFEM-24p` respectively. For the seeding and the removing at least one particle for each subelement was fixed as the minimum limit and 20 as the maximum. The strategy for the diffusion treatment was fully implicit, i.e. X-IVS method was employed.

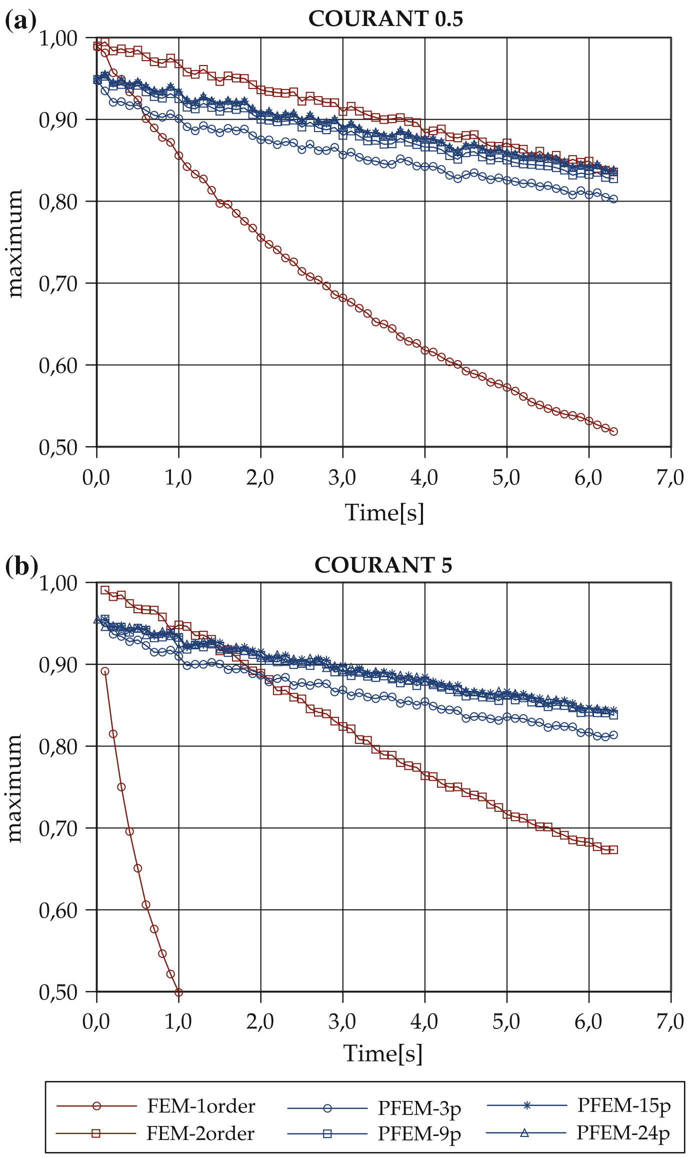


Fig. 17 Amplitude evolution on mesh M1 for $Co = 0.5$ (**a**) with $Co = 5$ (**b**). PFEM results are shown projected on the mesh

Figure 17 presents the evolution of the amplitude of the signal for the different simulations. It is confirmed again the fact that Eulerian simulations introduce a lot of numerical diffusion mainly due to the temporal scheme and the spatial stabilization.

It is noted that *PFEM* simulations start with an amplitude $\phi_{\max} \neq 1$, it is due to the projection of the maximum value on particles over the grid nodes. It is not an error because the particle information does not suffer for any type of numerical dissipation when is transported. The only error source is when the information is projected on the mesh for secondary computations. To confirm this, the signal amplitude over the particles may be viewed in Fig. 18. Here, another important result arises: the maximum value over the particles is independent of the number of particles initially seeded. This fact depends on the particle removing limits chosen during the computation and also on the projection operator design.

6.5.4 Simulation on a Finer Mesh $M2$

Figure 19 presents the evolution of the signal amplitude for different simulations. In this case the maximum on the mesh are almost the same as the maximum over the particles. It is due to the finer mesh involved, making the projection operation less diffusive.

7 Some Applications for More Complex Problems

This section finally end the paper showing some brief details about new promising and challenging applications of PFEM method. In some sense all theses applications present some sort of coupling problems. The first is a typical natural convection heat transfer problem in both, laminar and turbulent regime. The following example is the well known benchmark of turbulence modeling proposed by Rodi and Ferziger, a cube mounted on a channel floor and finally one example of multifluids flow, going towards the multiphase flow problems a very demanding need of the industry.

7.1 Thermal and Fluid Dynamics Coupled Problems

7.1.1 Natural Convection in a Square Cavity

The problem presented deals with the two dimensional flow with a Prandtl number $Pr = 0.71$ in a square cavity of side $H = 1[\mathrm{m}]$. The boundary conditions for the momentum equation are non slip at all boundaries. Horizontal walls are isolated, and the vertical sides are at different temperatures $T_c < T < T_h$ ($\phi = T$ for natural convection problems). Figure 20 exhibits the geometry of the cavity. Simulations were carried out using a mesh of triangular elements with 100×100 nodes and refinement towards the walls. The wide range of Ra numbers (48) was obtained by a constant temperature difference of $\Delta T = 1\,K$ adjusting the thermal expansion coefficient β to supply the desired Ra.

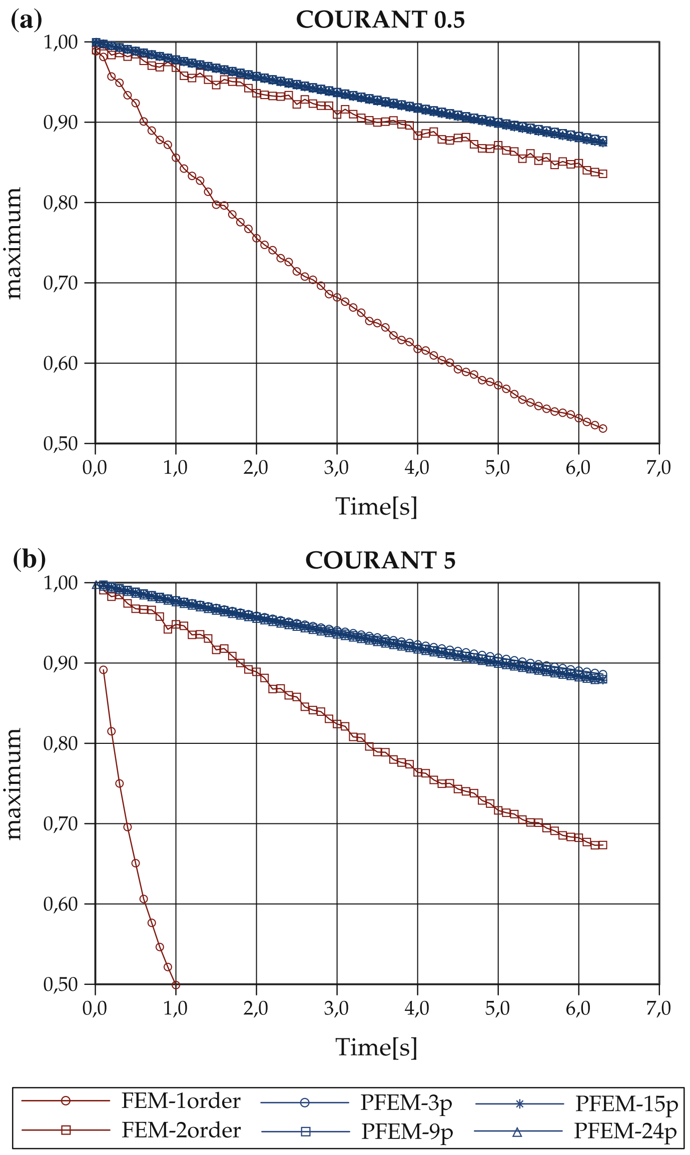


Fig. 18 Amplitude evolution on mesh M1 for $Co = 0.5$ (**a**) and for $Co = 5$ (**b**). PFEM results are shown on the particles, not projected on the mesh

$$Ra = \frac{g\beta H^3(\phi_h - \phi_c)}{\alpha\nu} \tag{48}$$

where α is the thermal diffusivity corresponding to air with the above mentioned Pr in standard temperature and pressure conditions.

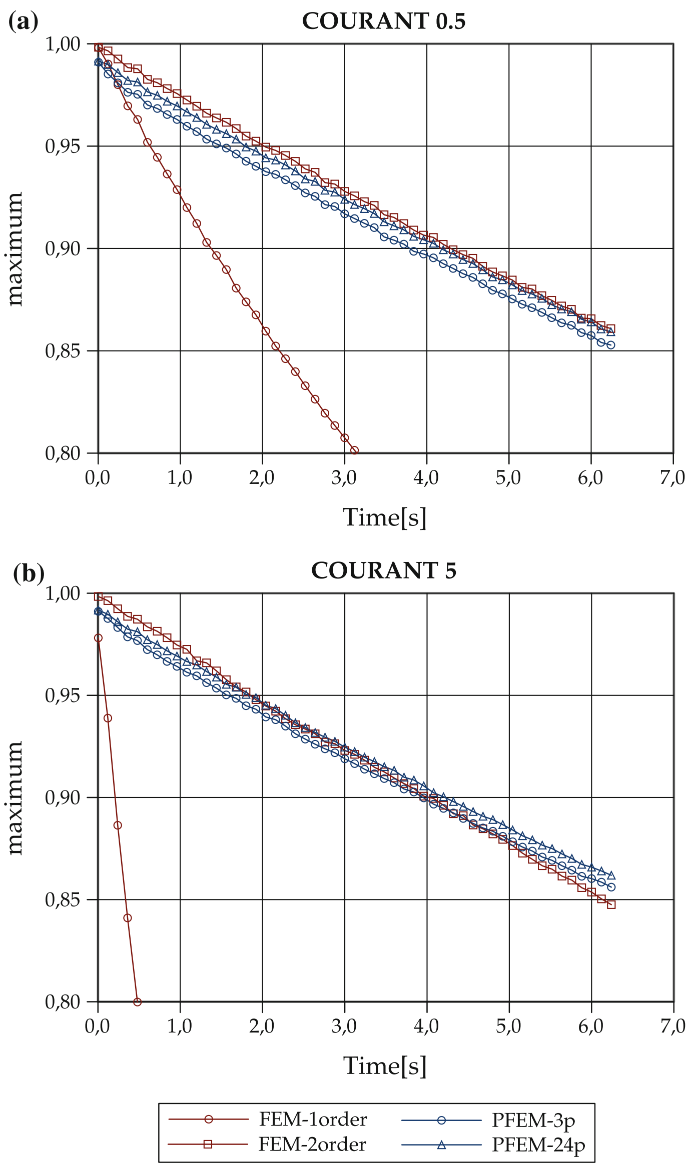


Fig. 19 Amplitude evolution on mesh $M2$ for $Co = 0.5$ (**a**) and for $Co = 5$ (**b**). PFEM results are shown on the mesh

7.1.2 Results and Discussion

This section provides a set of solutions at low Ra number. The quantities under study are the following:

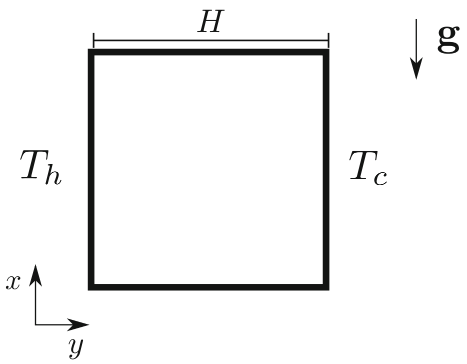


Fig. 20 Detail of cavity simulated, *left* wall at T_h, *right* wall at T_c, *top* and *bottom* walls are insulated

Table 1 Numerical solution for thermal square cavity with PFEM comparing with reference data

Ra	Data	PFEM2	Corzo [1]	Davis [2]
10^3	$u_{\max}$ (x = 0.5)	3.605	3.640	3.634
10^3	$y_{\max}$ (x = 0.5)	0.814	0.812	0.813
10^3	$v_{\max}$ (y = 0.5)	3.650	3.700	3.679
10^3	$x_{\max}$ (y = 0.5)	0.183	0.177	0.179
10^4	$u_{\max}$ (x = 0.5)	15.982	16.281	16.182
10^4	$y_{\max}$ (x = 0.5)	0.824	0.822	0.823
10^4	$v_{\max}$ (y = 0.5)	19.378	19.547	19.509
10^4	$x_{\max}$ (y = 0.5)	0.116	0.123	0.120
10^6	$u_{\max}$ (x = 0.5)	64.483	64.558	65.330
10^6	$y_{\max}$ (x = 0.5)	0.845	0.851	0.851
10^6	$v_{\max}$ (y = 0.5)	218.054	221.572	216.750
10^6	$v_{\max}$ (y = 0.5)	0.037	0.067	0.039

$u_{\max}(\frac{1}{2})$: The maximum horizontal velocity on the vertical mid-plane of the cavity (together with its location).

$v_{\max}(\frac{1}{2})$: The maximum vertical velocity on the horizontal mid-plane of the cavity (together with its location).

Table 1 shows PFEM results for $Ra = 10^3$, 10^4 and 10^6 compared with the [1, 2] solutions. Excellent agreement to experimental data in both results for momentum and energy equations prove the accuracy of this approach for this low Ra number range. The horizontal velocity component in the vertical mid-plane is shown in Fig. 21. Here is worthy to note that when Ra number increases the boundary layer becomes thinner and the maximum values in the velocity get closer to the walls. Finally Fig. 22 presents the temperature profiles for the three cases.

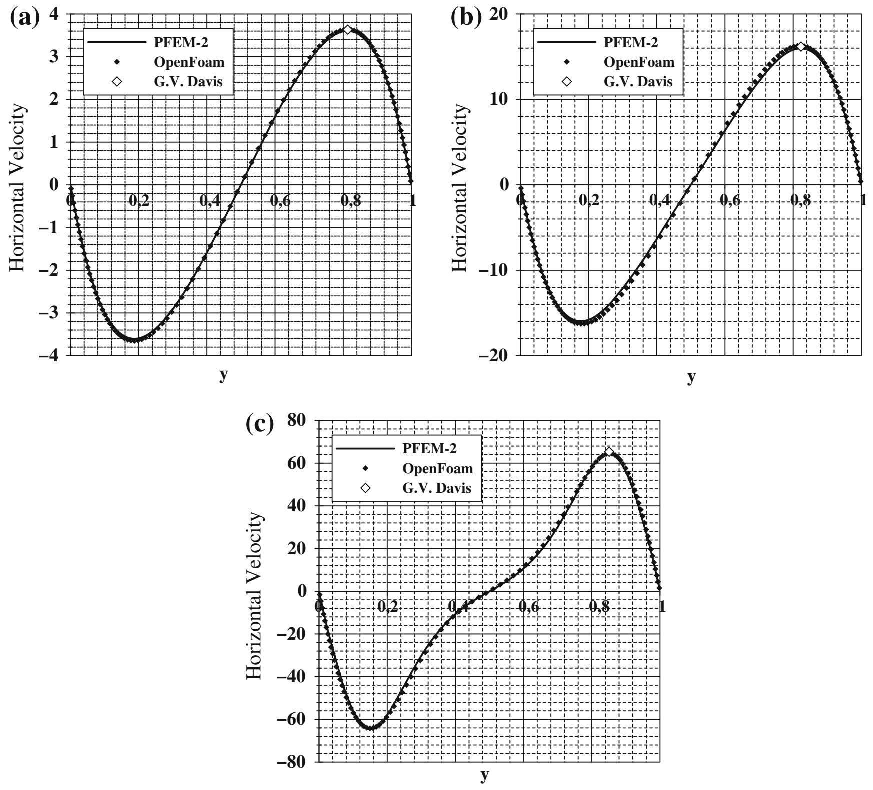


Fig. 21 Horizontal velocity profiles at x mid-plane to **a** $Ra = 10^3$, **b** $Ra = 10^4$ and **c** $Ra = 10^6$

7.1.3 Natural Convection in a Cubic Cavity

The schematic model for the problem is shown in Fig. 23. The cubic cavity is one meter length with an aspect ratio of unity and is filled with air as working fluid. The Prandtl number is fixed at $Pr = 0.71$. All surrounding walls are rigid and impermeable. The vertical walls located at $x = 0$ and $x = 1$ are retained to be isothermal but at different temperatures of T_h and T_c, respectively. The buoyancy force due to gravity works downwards (i.e., in negative z-direction).

7.1.4 Results and Discussion

For the present range of Ra numbers, solutions were obtained on a mesh with 81,000 tetrahedral elements and around of eighteen thousand nodes, and with refinement towards the walls. The following characteristic quantities are presented:

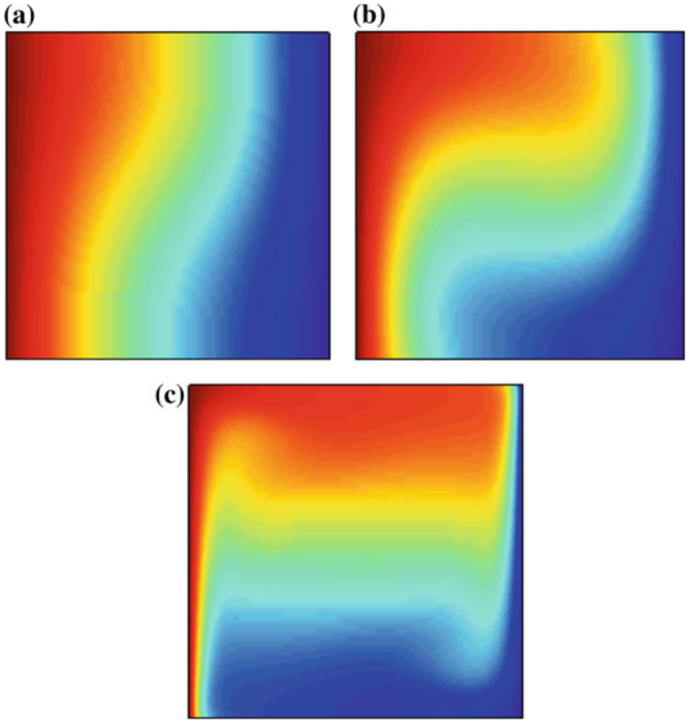
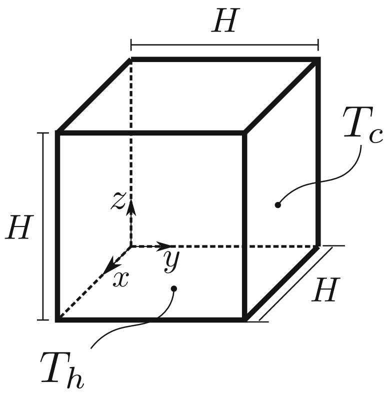


Fig. 22 Temperature field ϕ to **a** $Ra = 10^3$, **b** $Ra = 10^4$ and **c** $Ra = 10^6$

Fig. 23 Schematic model for the natural convection in a cubical cavity



$u_{\max}(\frac{1}{2})$: The maximum horizontal velocity for x-direction on center line (x = 0.5, y = 0.5) of the cavity and its location.
$w_{\max}(\frac{1}{2})$: The maximum vertical velocity for z-direction on center line (y = 0.5, z = 0.5) of the cavity and its location.

Table 2 Numerical solution for thermal cubic cavity with PFEM comparing with reference data

Ra	Data	PFEM	Wakashima [24]	Fusegi [5]
10^4	$u_{\max}(x = y = 0.5)$	0.1978	0.1989	0.2013
10^4	$z_{\max}(x = y = 0.5)$	0.8460	0.8250	0.8167
10^4	$w_{\max}(y = z = 0.5)$	0.2190	0.2211	0.2252
10^4	$x_{\max}(y = z = 0.5)$	0.1260	0.1253	0.1167
10^5	$u_{\max}(x = y = 0.5)$	0.1409	0.1423	0.1468
10^5	$z_{\max}(x = y = 0.5)$	0.8460	0.8500	0.8547
10^5	$w_{\max}(y = z = 0.5)$	0.2359	0.2407	0.2471
10^5	$x_{\max}(y = z = 0.5)$	0.0680	0.0751	0.0647
10^6	$u_{\max}(x = y = 0.5)$	0.0766	0.0813	0.0842
10^6	$z_{\max}(x = y = 0.5)$	0.8570	0.8500	0.8557
10^6	$w_{\max}(y = z = 0.5)$	0.2897	0.2382	0.2588
10^6	$x_{\max}(y = z = 0.5)$	0.0280	0.0500	0.0331

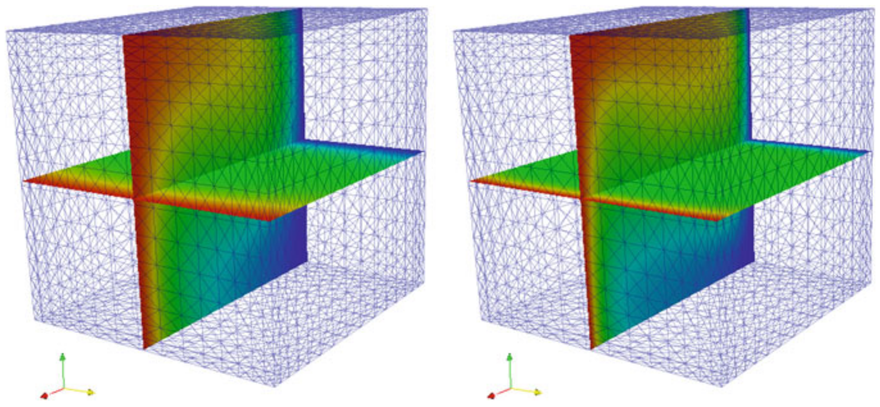


Fig. 24 Mesh with slices of section at mid-planes $y = 0.5$ and $z = 0.5$

Table 2 shows PFEM results for $Ra = 10^4$, 10^5 and 10^6 compared with the [5, 24] solutions. Finally Fig. 24 presents a wireframe of the mesh used with slices of section at mid-planes $y = 0.5$ and $z = 0.5$ respectively.

7.2 Turbulent Flows

7.2.1 Wall Mounted Cube Simulation

Turbulent flows around three-dimensional obstacles are common in nature and occur in many applications including flow around tall buildings, vehicles and computer chips. Understanding and predicting the properties of these flows are necessary for

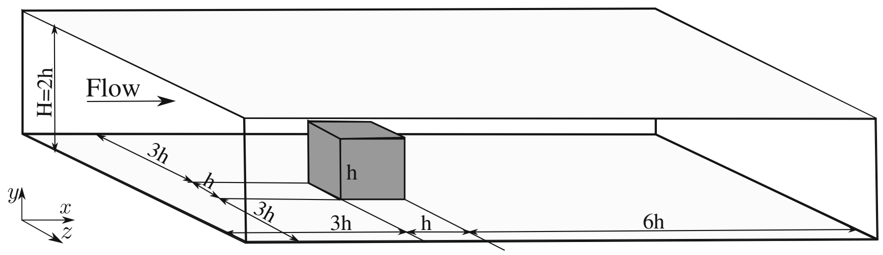


Fig. 25 Geometry for the flow around a cube obstacle

safe, effective and economical engineering designs. Experimental techniques are expensive and often provide data that is not sufficiently detailed. With the advent of supercomputers it has become possible to investigate these flows using numerical simulations.

In this paper the simulation of the turbulent flow around a cube obstacle is presented. This test is known as flow over a wall mounted cube, and it was analyzed experimentally by Martinuzzi and Tropea [17] and numerically by Sha and Ferziger [21], Lakehal and Rodi [15], and Rodi et al. [20] among others. Flow around a cube exhibits characteristics as three dimensionality of the mean flow, separation and large-scale unsteadiness. Quantitative results of this flow are scarce, then flows patters are exhaustively analyzed and compared.

The geometry of the problem is presented in the Fig. 25.

Computational Modeling
In this work, the numerical method used is the Particle Finite Element Method (PFEM) with Large Eddy Simulation (LES) for turbulence modeling. The Sub-Grid Scale (SGS) model used is the Static Smagorinsky model. Regarding to the computational domain, the problem was solved using two grids: the first one is a relatively coarse grid of one million of tetrahedral elements (refined towards the cube and behind it), with a mesh-size of $\delta_h = h/25$ over the cube. On the other hand, the second grid has the same kind of refinement but it has around four million of tetrahedral elements, with $\delta_h = h/40$.

The spanwise boundary condition is slip, the spanwise width is $7\,h$, assuring that blockage effects are small. In the streamwise direction, inflow-outflow boundary conditions are used. A parabolic flow with some perturbations is used at the inlet and fixed pressure condition is applied at the exit. The streamwise length of the domain is $10\,h$.

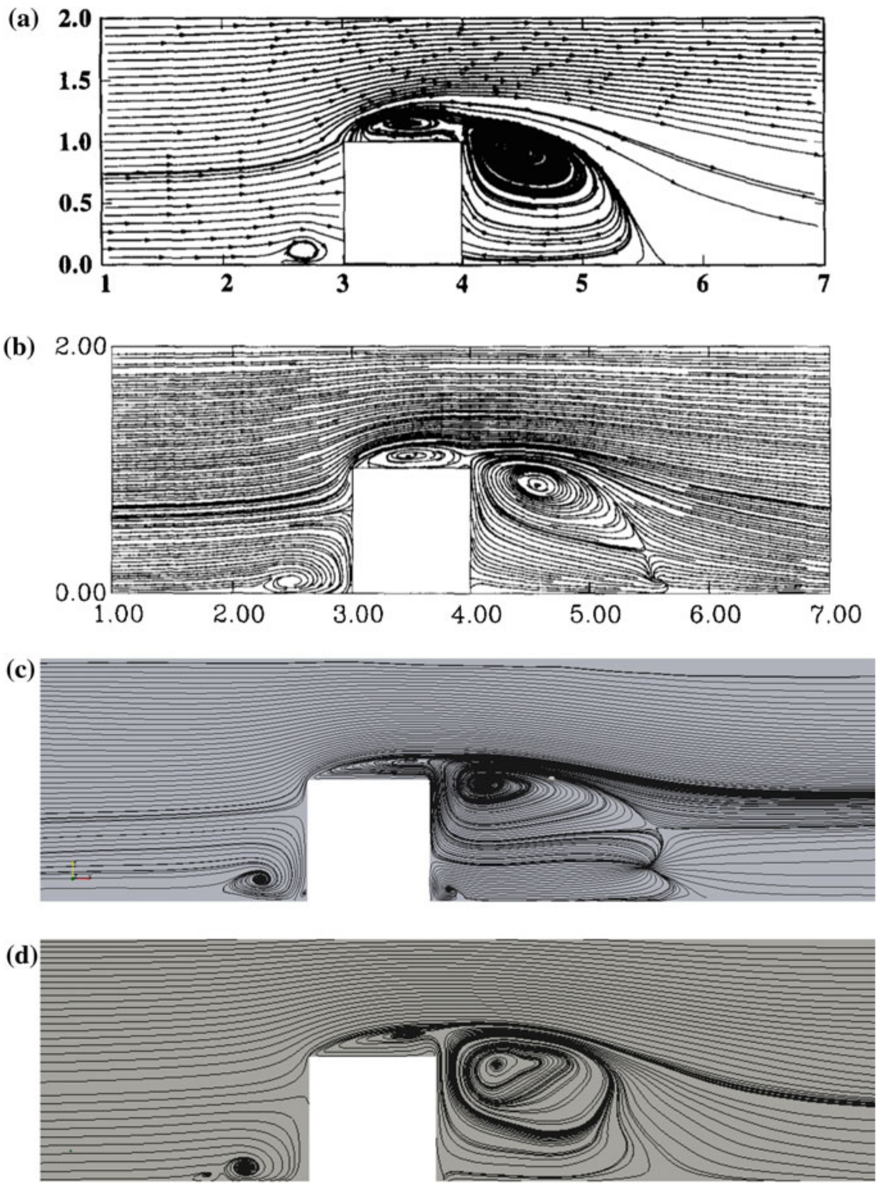


Fig. 26 The streamlines on the symmetry plane at $Re = 40{,}000$. **a** shows the experimental result of Martinuzzi and Tropea [17], **b** result from LES simulation on [21], and **c** and **d** presents the results of PFEM using a coarse and finer mesh respectively

Summary of Results

Large eddy simulations were performed at $Re = 40{,}000$. Figure 26 shows a comparison of time-averaged streamlines on the symmetry plane. The overall prediction of the separation region on the roof and behind the obstacle is quite good even using coarse grids. Shah and Ferziger commented that in its simulations the stagnation point was located high on the front face, and in this work could be arrived the same conclusion. Fluid striking the body above it goes over the obstacle and using the finer mesh we can find a solution where it reattach on the roof, something that Shah could not. Using the finer mesh, the rear recirculation region is not closed, but streamlines originating upstream of the obstacle do not enter this region; fluid enters the rear recirculation region from sides. Near the top of the recirculation region we find the head of the arch vortex. Results using coarse mesh are not accurate, mainly behind the obstacle.

Figure presents the time-averaged streamlines on the floor of the channel. The streamline patterns are consistent with those observed by Martinuzzi and Tropea [17]. These streamlines, which may be viewed as skin friction lines, show the complexity of this 3-D flow. On the reference [21], the primary separation occurs at a saddle point located about one obstacle height (1.05 h) ahead of the obstacle (experimental value = 1.026), whereas PFEM simulation reach approximately (0.89 h) with coarse grid and (0.92 h) with the finer grid. The separation region wraps around the obstacle and forms a strong horseshoe vortex. The converging and diverging streamlines that mark the extent of this vortex are regions of strong upwash and downwash. This horseshoe is better represented by PFEM using the finer mesh, whereas with the coarse mesh the streamlines are too much closed behind the obstacle. Instantaneous pictures (not presented here) of the flow show that the horseshoe vortex is, in fact, highly intermittent; an intact structure is almost never found in these snapshots. The mean flow on the side faces is entirely reversed. In Shah and Ferziger, the primary reattachment length of $1.65\,h$ agrees well with the experimental value of $1.61\,h$, however PFEM reattachment is found in $1.8\,h$.

In the work of Shah and Ferziger [21], it is said that both the primary separation point ahead of the obstacle and the rear reattachment points are singular points (zero skin friction) where the so-called separation lines begin and end. Also, they comment that the owl-face shaped streamlines in the rear recirculation zone of the obstacle correspond to the base of the arch vortex. The arch vortex is formed by quasi-periodic vortex shedding from the upstream vertical corners that resembles a von Karman street. This intact arch vortex exists only in the mean flow and is an artifact of averaging and PFEM can reproduce only approximately this behavior, and strangely with a coarse mesh the result are more accurate. Must be noticed that both grids are not good enough near the floor of the channel, then a better refinement is required to reach the same quality of results as Shah and Ferziger.

Efficiency

In this section the scalability of the current implementation of PFEM is presented. The above mentioned test, using the finer grid, was carried out over a Infiniband

Table 3 CPU-times comparison in seconds between different PFEM2 algorithms and OpenFOAM for one, two and four cores

Cores	$1x$ (s)	$2x$ (s)	$4x$ (s)
OpenFOAM	754	402	286
PFEM2 moving mesh (CIMNE)	484	371	326
PFEM2 fixed mesh (CIMNE)	284	179	138
PFEM2 fixed mesh (CIMEC)	330	176	99

interconnected cluster, which has dual socket nodes with Intel Xeon E5-2600 CPUs and 64 Gb RAM. The interconnection is with IB-QDR 40 Gbps (Fig. 27).

Figure 28 presents the scalability of each PFEM stage and of the entire simulation using an Eulerian weighting strategy, obtaining approximately the same number of degrees of freedom in each partition. Could be noted that the efficiency of the Infiniband cluster is good enough also running with 32 cores, reaching a global $S_{32} \approx 26x$. Using more cores the efficiency decays because there is not enough work for each process to overweight the communication time.

7.3 Multifluids

7.3.1 Sloshing Test

In this section a comparison with the results of the sloshing test is presented. For the experiment, the same mesh and configuration than that presented in Idelsohn et al. [9] have been used (Figs. 29 and 30).

Table 3 shows the computational time necessary to simulate 1 sec. in an Intel(R) Core(TM) i7-3820 CPU 3.60 GHz with OpenFOAM and PFEM2 versions of the International Center for Numerical Methods in Engineering (CIMNE).

On the other hand, the test of the implementation presented in this paper was executed in an Intel(R) Core(TM) i5-3230M CPU 2.60 GHz. To match the heterogeneous platforms a benchmarking factor $\frac{4{,}007}{9{,}010}$ (extracted for the web-page http://cpubenchmark.net/high_end_cpus.html) is used, and the final values are presented in the table.

The reported values evidence that, for the settings described, *PFEM* with fixed mesh is more than $2\times$ faster than OpenFOAM.

7.3.2 Dam-Break Test

In this section a comparison with the results of a dam-break test is presented. For the experiment, the same mesh and configuration which is presented in Idelsohn et al. [9] have been used.

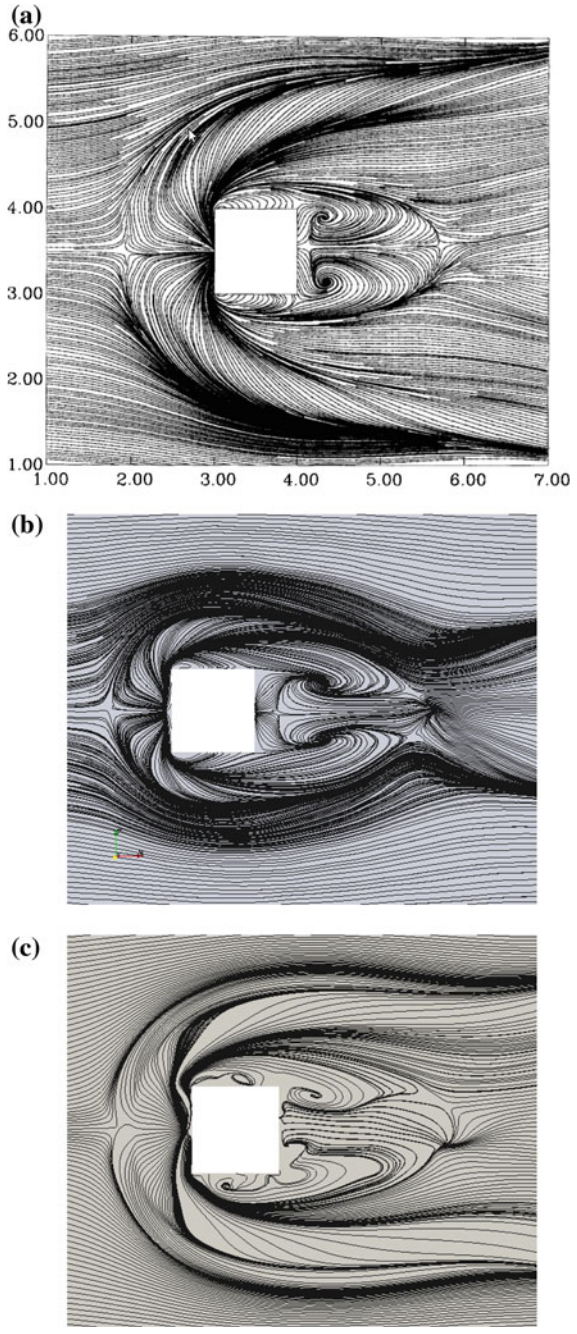


Fig. 27 The streamlines in a plane near to the floor at $Re = 40{,}000$. **a** shows the numerical result of Shah and Ferziger [21], and **b** and **c** present the results of PFEM using a finer mesh respectively

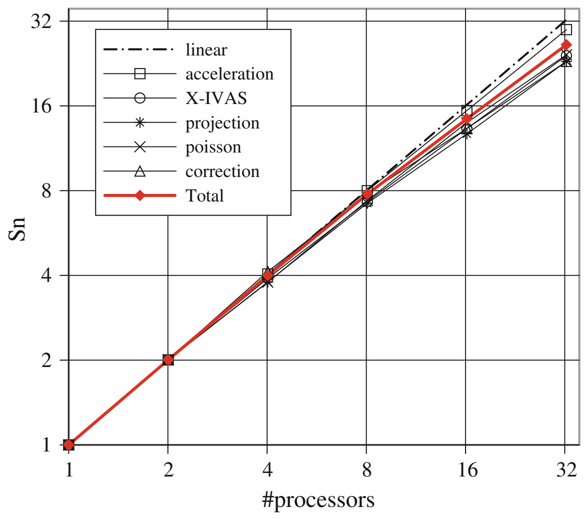


Fig. 28 Speed-up over an Infiniband cluster. Case: flow around a mounted cube in 3d

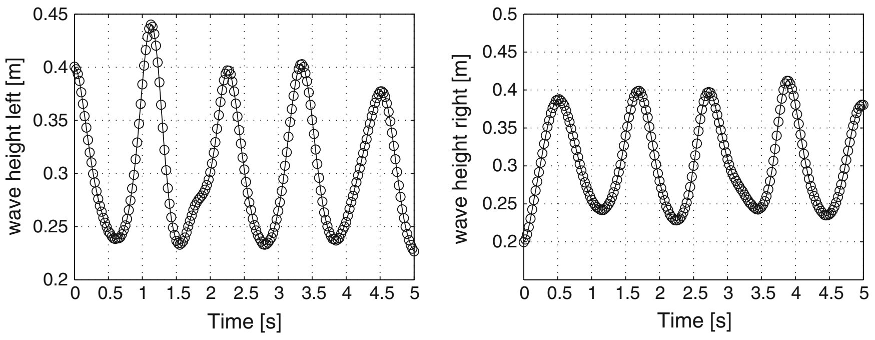


Fig. 29 Interface relative height at the vertical walls (*left side* and *right side*) for *PFEM* fixed mesh

Figure 31 presents snapshots of the simulation. Comparing with the results obtained in [9], good agreement both in the shape of the free surface and in the time evolution with experimental and OpenFOAM results can be observed. The current version is using Courant number larger than 10 (maximum 15), without present any difficulty for this large time-step.

The CPU-Time time needed to simulate 1 s of real time is 274.75 s (running in serial mode). Idelsohn et al. reports 278 s for the CIMNE pfem2 fixed mesh version and 473 s with OpenFOAM.

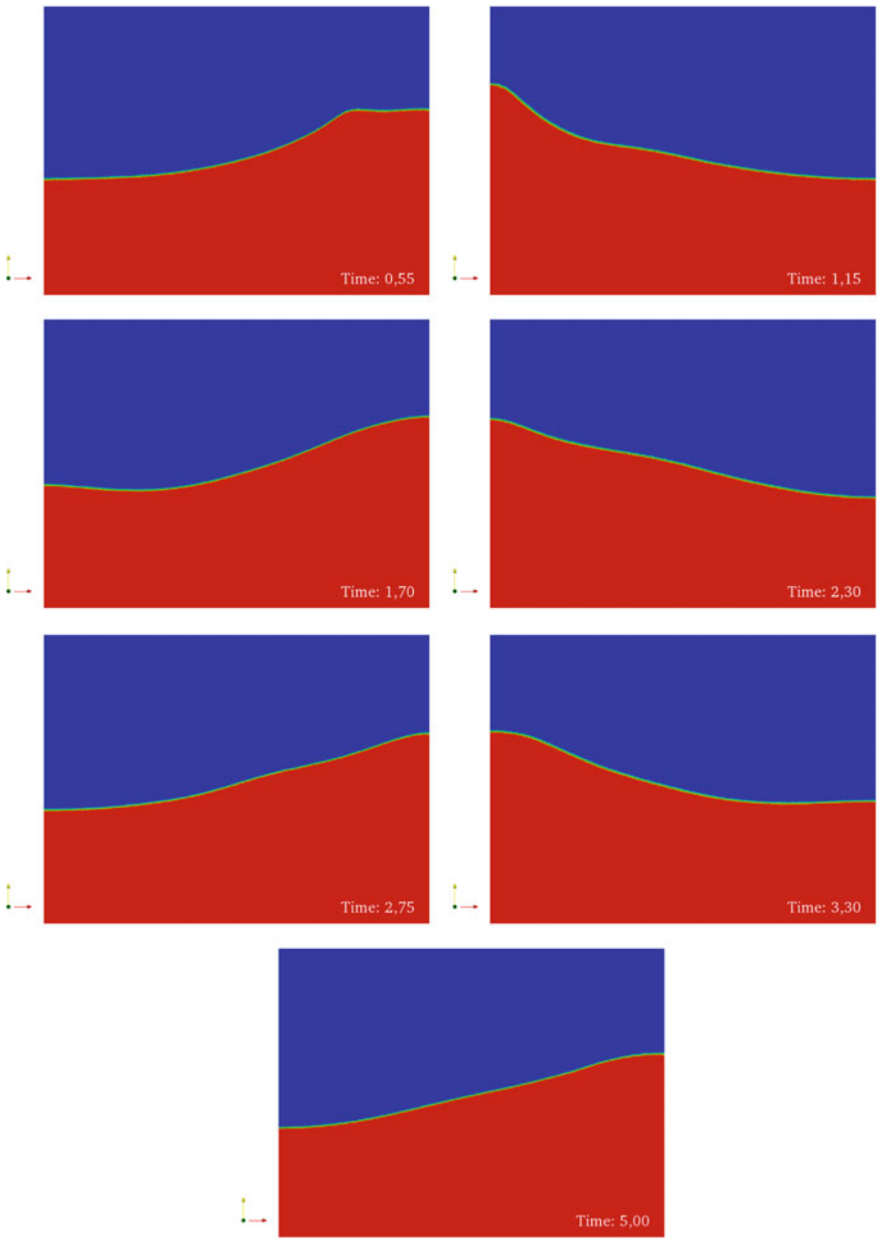


Fig. 30 From *left* to *right* and *top* to *bottom*: sloshing of two immiscible fluids with a large jump in the density: snapshots at different time steps (t = 0.55, 1.15, 1.7, 2.3, 2.75, 3.35 and 5 s.)

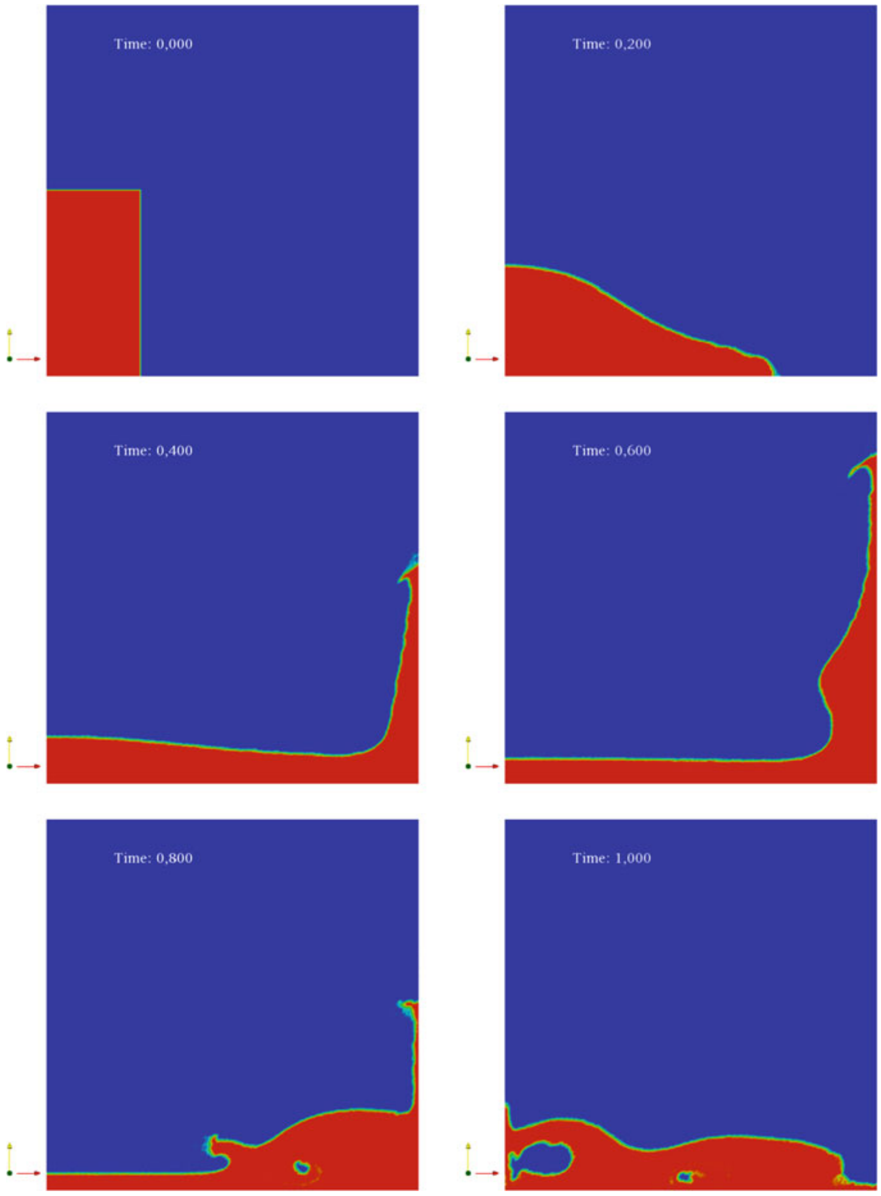


Fig. 31 From *left* to *right* and *top* to *bottom*: snapshots of the dam break without obstacle at t = 0, 0.2, 0.4, 0.6, 0.8 and 1 s

8 Conclusions

In this paper a review and the present developments of PFEM are presented. In recent years much effort has been devoted to improve the performance of this method in order to make it competitive with the rest of the solvers mostly used in computational mechanics. Not only that, but with recent findings that have emerged is thought to be on the gates of a paradigm shift in the way of performing the simulations, especially considering that the community is demanding of methods that are commensurate with the needs of engineering design.

While this paper does not delve into the numerical analysis it establishes the basis to do so in the next few years with the target to demonstrate mathematically the goodness of the Lagrangian methods of this type in front of the very commonly used Eulerian methods.

Finally the last goal has been to show that in addition to the well-known virtues that owns the method to resolve problems with heterogeneous flows, it is also possible to implement complex homogeneous flows, as in the case of turbulence and cases with thermal coupling.

Acknowledgments This work was partially supported by the European Research Council under the Advanced Grant: ERC-2009-AdG Real Time Computational Mechanics Techniques for Multi-Fluid Problems. Norberto Nigro and Juan Gimenez want to thanks to CONICET,Universidad Nacional del Litoral and ANPCyT for their financial support (grants PICT 1645 BID (2008), CAI+D 65-333 (2009)). Also thanks to Santiago Marquez Damian for their invaluable assistance in show the goodness of PFEM vis–vis other solvers available, of whom Santiago is an expert user. To Eugenio Oñate and CIMNE for their unconditional support and his teachings throughout his scientific life. To Pedro Morin, Marta Bergallo for interesting mathematics discussions and to Nestor Calvo and Pablo Novara for sharing some discussions about mesh generation and computational geometry.

References

1. Corzo S, Marquez Damian S, Nigro N (2011) Numerical simulation of natural convection phenomena. Mecánica Computacional XXX:277–296
2. De Vahl Davis G (1983) Natural convection of air in a square cavity: a benchmark numerical solution. Int J Num Meth Fluids 3:249–264
3. Del Pin F (2003) The meshless finite element method applied to a lagrangian particle formulation of fluid flows. Ph.D. Thesis, Facultad de Ingeniería y Ciencias Hídricas (FICH) Instituto de Desarrollo Tecnológico para la Industria Química (INTEC) Universidad Nacional del Litoral
4. Donea J, Huerta A (1983) Finite element method for flow problems. Wiley, Chichester
5. Fusegi T, Hyun J, Kuwahara K, Farouk B (1991) A numerical study of three-dimensional natural convection ina differentially heated cubical enclosure. Int J Heat Mass Transfer 34:1543–1551
6. Gimenez JM, Nigro NM (2011) Parallel implementation of the particle finite element method. Mecánica Computacional XXX:3021–3032
7. Gimenez JM, Nigro NM, Idelsohn SR (2012) Improvements to solve diffusion-dominant problems with pfem-2. Mecánica Computacional XXXI:137–155
8. Hryb D, Cardozo M, Ferro S, Goldschmidt M (2009) Particle transport in turbulent flow using both lagrangian and eulerian formulations. Int Cummun Heat Mass Transfer 36:451–457

9. Idelsohn S, Marti JM, Becker P, Oñate E (2014) Analysis of multi-fluid flows with large time-steps using the particle finite element method. Int J Num Meth in Fluids (in press)
10. Idelsohn S, Nigro NM, Limache A, Oñate E (2012) Large time-step explicit integration method for solving problems with dominant convection. Comput Methods Appl Mech Eng 217–220:168–185
11. Idelsohn SR, Nigro NM, Gimenez JM, Rossi R, Marti J (2013) A fast and accurate method to solve the incompressible navier-stokes equations. Eng Comput 30(2):197–222
12. Idelsohn SR, Oñate E, Calvo N, Del Pin F (2003) The meshless finite element method. Int J Num Meth Eng 58(6):893–912
13. Idelsohn SR, Oñate E, Del Pin F (2004) The particle finite element method a powerful tool to solve incompressible flows with free-surfaces and breaking waves. Int J Numer Meth 61:964–989
14. Jasak H (1996) Error analysis and estimation for the finite volume method with applications to fluid flows. Ph.D. Thesis, London
15. Lakehal D, Rodi W (1997) Calculation of the flow past a surface-mounted cube with two-layer turbulence models. J Wind Eng Ind Aerodyn 67:65–78
16. Leveque R (2002) Finite volume methods for hyperbolic problems, 1st edn. Cambridge University Press, Cambridge
17. Martinuzzi R, Tropea C (1993) The flow around surface-mounted, prismatic obstacles placed in a fully developed channel flow. J Fluids Eng 115:85–92
18. Nigro N, Gimenez J, Limache A, Idelsohn S, Oñate E, Calvo N, Novara P, Morin P (2011) A new approach to solve incompressible navier-stokes equation using a particle method. Mecánica Computacional XXX
19. Oñate E, Idelsohn SR, Del Pin F, Aubry R (2004) The particle finite element method, an overview. Int J Comput Meth 1:267–307
20. Rodi W, Ferziger J, Breuer M, Pourquie M (1997) Status of large eddy simulation: results of a workshop. Trans ASME J Fluid Eng 119:248–262
21. Shah KB, Ferziger JH (1997) A fluid mechanicians view of wind engineering: large eddy simulation of flow past a cubic obstacle. J Wind Eng Ind Aerodyn 67&68:211–224
22. Sklar DM, Gimenez JM, Nigro NM, Idelsohn SR (2012) Thermal coupling in particle finite element method - second generation. Mecánica Computacional XXXI:4143–4152
23. Stam J (1999) Stable fluids. In: SIGGRAPH 99 Conference Proceedings, Annual Conference Series, pp 121–128
24. Wakashima S, Saitoh T (2004) Benchmark solutions for natural convection in a cubic cavity using the high-order time-space method. Int J Heat Mass Transfer 47:853–864

Part VI
Fluid-Structure Interactions Problems

Computational Engineering Analysis and Design with ALE-VMS and ST Methods

Kenji Takizawa, Yuri Bazilevs, Tayfun E. Tezduyar, Ming-Chen Hsu, Ole Øiseth, Kjell M. Mathisen, Nikolay Kostov and Spenser McIntyre

Abstract Flows with moving interfaces include fluid–structure interaction (FSI) and quite a few other classes of problems, have an important place in engineering analysis and design, and pose significant computational challenges. Bringing solution and analysis to them motivated the Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) method and also the variational multiscale version of the Arbitrary Lagrangian–Eulerian method (ALE-VMS). These two methods and their improved versions have been applied to a diverse set of challenging problems with a common core computational technology need. The classes of problems solved include free-surface and two-fluid flows, fluid–object and fluid–particle interaction, FSI, and flows with solid surfaces in fast, linear or rotational relative motion. Some of the most challenging FSI problems, including parachute FSI, wind-turbine FSI and arterial FSI, are being solved and analyzed with the DSD/SST and ALE-VMS methods as core technologies. Better accuracy and improved turbulence modeling were brought with the recently-introduced VMS version of the DSD/SST method, which is called DSD/SST-VMST (also ST-VMS). In specific classes of problems, such as parachute

K. Takizawa (✉)
Department of Modern Mechanical Engineering and Waseda Institute for Advanced Study, Waseda University, 1-6-1 Nishi-Waseda, Shinjuku-ku, Tokyo 169-8050, Japan
e-mail: Kenji.Takizawa@tafsm.org

Y. Bazilevs
Structural Engineering, University of California, San Diego, 9500 Gilman Drive, La Jolla, CA 92093, USA

T. E. Tezduyar · N. Kostov · S. McIntyre
Mechanical Engineering, Rice University, 6100 Main Street, Houston, TX 77005, USA

M.-C. Hsu
Department of Mechanical Engineering, Iowa State University, 2025 Black Engineering, Ames, IA 50011, USA

O. Øiseth · K. M. Mathisen
Department of Structural Engineering, Norwegian University of Science and Technology, 7491 Trondheim, Norway

FSI, arterial FSI, ship hydrodynamics, fluid–object interaction, aerodynamics of flapping wings, and wind-turbine aerodynamics and FSI, the scope and accuracy of the modeling were increased with the special ALE-VMS and ST techniques targeting each of those classes of problems. This article provides an overview of how the core and special ALE-VMS and ST techniques are used in computational engineering analysis and design. The article includes an overview of three of the special ALE-VMS and ST techniques, which are just a few examples of the many special techniques that complement the core methods. The impact of the ALE-VMS and ST methods in engineering analysis and design are shown with examples of challenging problems solved and analyzed in parachute FSI, arterial FSI, ship hydrodynamics, aerodynamics of flapping wings, wind-turbine aerodynamics, and bridge-deck aerodynamics and vortex-induced vibrations.

1 Introduction

Flows with moving interfaces include fluid–structure interaction (FSI), fluid–object interaction (FOI), fluid–particle interaction (FPI), free-surface and multi-fluid flows, and flows with solid surfaces in fast, linear or rotational relative motion. These problems are frequently encountered in engineering analysis and design, pose some of the most formidable computational challenges, and have a common core computational technology need. That crucial need motivated the development of the Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) method [1–5], which is a general-purpose interface-tracking (moving-mesh) technique, as a core computational technology. The DSD/SST method is an alternative to the Arbitrary Lagrangian–Eulerian (ALE) finite element formulation [6], which is the most widely used moving-mesh technique, with increased emphasis on FSI in recent years (see, for example, [7–26]). Though less widely used than the ALE formulation, over the past 20 years the DSD/SST method has been applied to some of the most challenging moving-interface problems. The classes of problems solved with the DSD/SST method since its inception include the free-surface and multi-fluid flows [1, 27–30], FOI [1, 28, 30], aerodynamics of flapping wings [31–34], flows with solid surfaces in fast, linear or rotational relative motion [15, 28, 29, 35–37], compressible flows [28], shallow-water flows [29, 38], FPI [28, 29], and FSI [3–5, 31, 39–57]. Very recently, a new version of the DSD/SST method that can address the computational challenges involved in topology changes, such as contact between solid surfaces, was introduced in [58] with the name "ST-TC."

In the DSD/SST formulation, the ST computations are carried out one ST "slab" at a time, where the "slab" is the slice of the ST domain between the time levels n and $n+1$. The basis functions are continuous within a ST slab, but discontinuous from one ST slab to another. The original DSD/SST method [1] is based on the SUPG/PSPG stabilization, where "SUPG" and "PSPG" stand for the Streamline-Upwind/Petrov-Galerkin [59] and Pressure-Stabilizing/Petrov-Galerkin [1] methods. Starting in its very early years, the DSD/SST method also included the "LSIC" (least-squares on incompressibility constraint) stabilization. New versions of the DSD/SST method have been introduced since its inception, including those in [3], which have been serving as the core numerical technology in the majority of the ST FSI computations carried out in recent years. The most recent DSD/SST method is the ST version [4, 5] of the residual-based variational multistage (RBVMS) method [60–63]. It was named "DSD/SST-VMST" (i.e. the version with the VMS turbulence model) in [4], which was also called "ST-VMS" in [5]. The original DSD/SST method was named "DSD/SST-SUPS" in [4] (i.e. the version with the SUPG/PSPG stabilization), which was also called "ST-SUPS" in [22].

The ALE-VMS formulation [15, 54] is a moving-domain extension of the RBVMS formulation, originally proposed in [62] and successfully applied to simulation of turbulent flows and FSI in [9–11, 23, 63–68]. An important additional feature of the ALE-VMS methodology is weak enforcement of essential boundary conditions. Weakly enforced essential boundary conditions were introduced in [69]. Weak boundary conditions produce significantly more accurate solutions than strongly enforced boundary conditions on meshes with insufficient boundary-layer resolution [63, 70, 71], which is almost always the case in practice. The ALE-VMS method with weakly enforced boundary conditions is the main computational technology behind the ALE-VMS computations presented in this chapter.

The Mixed Interface-Tracking/Interface-Capturing Technique (MITICT) [29] was introduced primarily for FOI with multiple fluids (see, for example, [72, 73]). The MITICT was successfully tested in [30], where the interface-tracking technique was an ST formulation, and the interface-capturing method was the Edge-Tracked Interface Locator Technique (ETILT) [29]. It was also tested in [74] by using a moving Lagrangian interface technique [75] for interface tracking and the ETILT.

In this article, the ALE-VMS formulation is used in the context of the MITICT technique, in which the air-water interface is captured using the level set approach [76]. The level set function, which is convected with the flow, is used for separating the air and water subdomains. The Navier–Stokes equations of incompressible flows are employed in both air and water subdomains. The Navier–Stokes and level set equations are written in an ALE frame [6]. The rigid object is described using balance equations of linear and angular momentum. The ALE technique is employed to track the interface between the moving fluid domain (consisting of air and water subdomains) and the rigid object. Application of the ALE-VMS formulation to free-surface flow and FOI may be found in [12, 16, 77, 78].

Moving-mesh methods require mesh update methods. Mesh update consists of moving the mesh for as long as possible and remeshing as needed. With the key objectives being to maintain the element quality near solid surfaces and to minimize frequency of remeshing, a number of advanced mesh update methods [3, 27, 29, 79, 80] were developed to be used with the DSD/SST method, including those that minimize the deformation of the small elements placed near solid surfaces.

An ST method will naturally involve more computational cost per time step than an ALE method, but it gives us the option of using higher-order basis functions in time, including the NURBS basis functions, which have been used very effectively as spatial basis functions (see [9, 64, 81, 82]). This of course increases the order of accuracy in the computations [4, 5, 48], and the desired accuracy can be attained with larger time steps, but there are positive consequences beyond that. The ST context provides us better accuracy and efficiency in temporal representation of the motion and deformation of the moving interfaces and volume meshes, and better efficiency in remeshing. This has been demonstrated in a number of 3D computations, specifically, flapping-wing aerodynamics [32–34, 83], separation aerodynamics of spacecraft [56], and wind-turbine aerodynamics [37].

There are some advantages in using a discontinuous temporal representation in ST computations. For a given order of temporal representation, we can reach a higher order accuracy than one would reach with a continuous representation of the same order. When we need to change the spatial discretization (i.e. remesh) between two ST slabs, the temporal discontinuity between the slabs provides a natural framework for that change. There are advantages also in continuous temporal representation. We obtain a smooth solution, NURBS-based when needed. We also can deal with the computed data in a more efficient way, because we can represent the data with fewer temporal control points, and that reduces the computer storage cost. These advantages motivated the development of the ST computation techniques with continuous temporal representation (ST-C) [84].

The core and special ALE-VMS and ST FSI methods mentioned above were motivated by the need for the solution and analysis of specific classes of challenging problems, such as parachute FSI, arterial FSI, aerodynamics of flapping wings, ship hydrodynamics and FOI, and wind-turbine aerodynamics and FSI. This can be seen from the ALE-VMS and ST articles cited in the first paragraph, especially the articles since 2008, and will also be seen from the examples we will present in this chapter. In the case of the parachute FSI, the special methods were motivated also by the need for supporting the design process for the NASA spacecraft parachutes.

For the governing equations and core methods, including the ALE-VMS and DSD/SST methods, and for much of the special techniques, we refer the interested reader to [15, 22, 33, 50, 54]. An overview of three of the special techniques is provided in Sect. 2. Examples of the challenging problems solved are presented in Sect. 3, and the concluding remarks are given in Sect. 4.

2 Special Methods

A certain class of FSI problems might involve some specific computational challenges beyond those encountered in a typical FSI problem. That requires development of special FSI methods targeting those challenges. A good number of special methods were developed in conjunction with the core ST FSI method to address the specific computational challenges involved in parachute FSI [53], patient-specific arterial FSI [50], aerodynamics of flapping wings [33, 34], and wind-turbine aerodynamics [37]. The details on these special methods can be found in the references cited above. Here we give three examples.

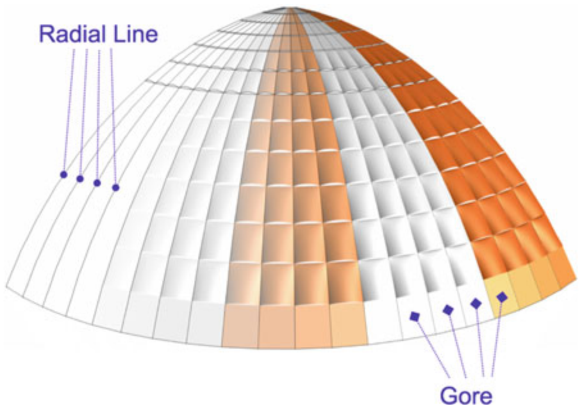


Fig. 1 Parachute radial lines and gores

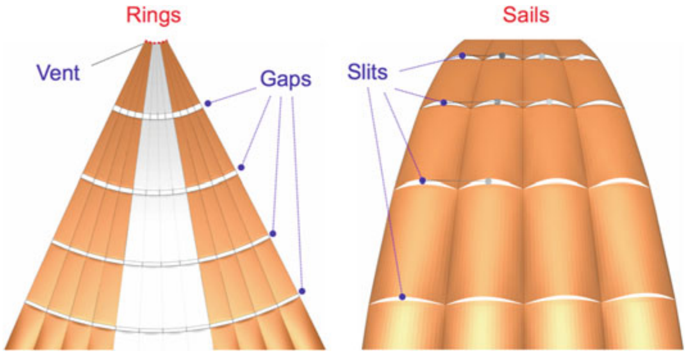


Fig. 2 Rings, sails, ring gaps, and sail slits

2.1 Homogenization Model for Ringsail Parachutes

Parachute FSI involves all the computational challenges of a typical FSI problem. Spacecraft parachutes are often very large ringsail parachutes, made of a large number of gores, where a gore is the slice of the canopy between two radial reinforcement cables running from the parachute vent to the skirt (see Fig. 1). Ringsail parachute gores are constructed from rings and sails, resulting in a parachute canopy with hundreds of ring gaps and sail slits (see Fig. 2). The complexity created by this geometric porosity makes FSI modeling inherently challenging.

The Homogenized Modeling of Geometric Porosity (HMGP) [3] and its new version, "HMGP-FG" [53], were introduced to help us bypass the intractable complexities of the geometric porosity by approximating it with an equivalent, locally varying homogenized porosity. In HMGP-FG, the normal velocity crossing the parachute canopy under a pressure differential Δp is modeled as

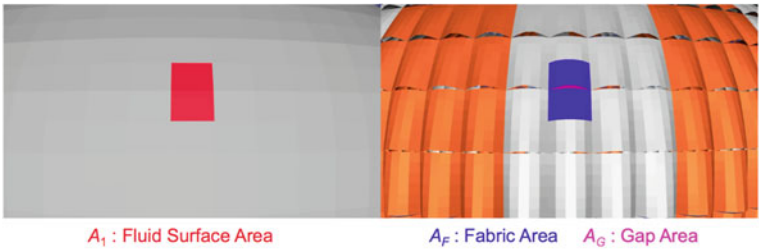


Fig. 3 Areas used in HMGP-FG

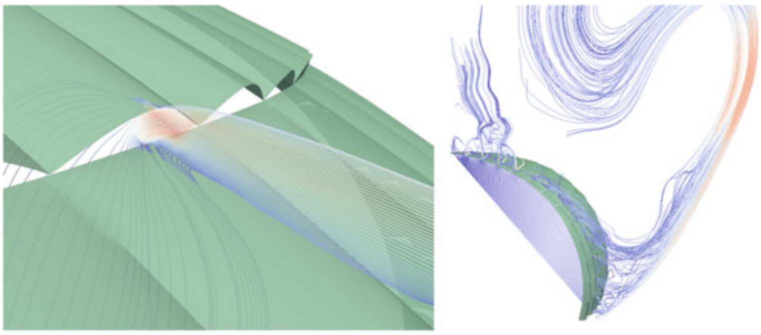


Fig. 4 The two porosity coefficients for each patch are calculated in a one-time fluid mechanics computation with an n-gore slice of the parachute canopy, where the flow through all the gaps and slits is resolved. Expect for the first and last patches, each patch contains a gap or a slit. See [3, 53] for details

$$u_n = -(k_F)_J \frac{A_F}{A_1} \Delta p - (k_G)_J \frac{A_G}{A_1} \operatorname{sgn}(\Delta p) \sqrt{\frac{|\Delta p|}{\rho}}, \tag{1}$$

where A_1, A_F and A_G are defined in Fig. 3, and $(k_F)_J$ and $(k_G)_J$ are the homogenized porosity coefficients for each patch J, calculated in a one-time fluid mechanics computation with an n-gore slice of the parachute canopy (see Fig. 4). Even in a fully open configuration, the parachute canopy goes through a periodic breathing motion where the diameter varies between its minimum and maximum values. The shapes and areas of the gaps and slits vary significantly during this breathing motion (see Fig. 5). The porosity coefficients have very good invariance properties with respect to these shape and area changes, and this can be seen in Fig. 6.

2.2 *Flapping-Wing Motion Representation with Higher-Order Temporal Functions*

Computer modeling of the aerodynamics of flapping wings requires an accurate temporal representation of the motion and deformation of the wings. It also requires

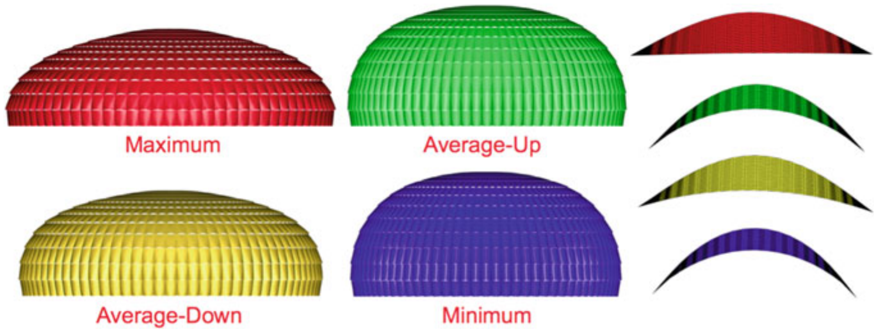


Fig. 5 The shapes and the areas of the slits vary significantly during the canopy breathing motion

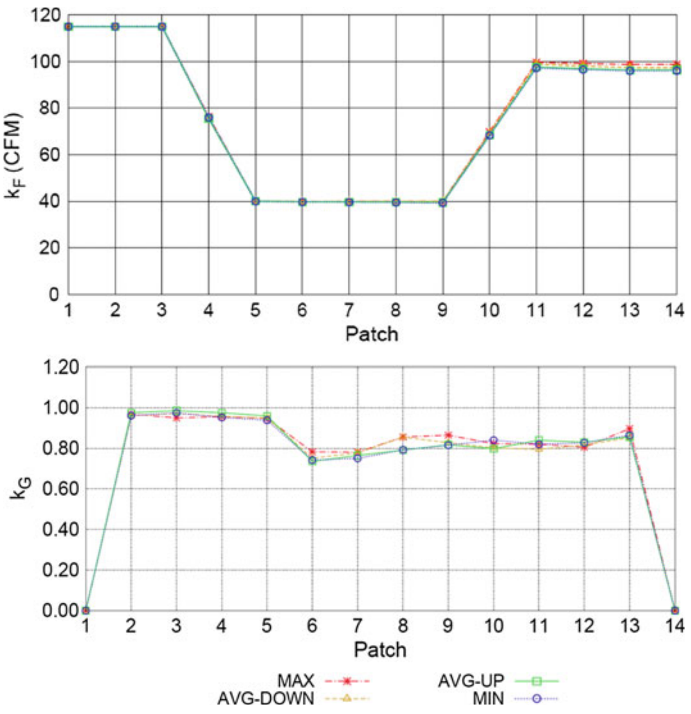


Fig. 6 The porosity coefficients $(k_F)_J$ and $(k_G)_J$ for each patch J, at different canopy shapes during the breathing motion. The plots show good invariance for these coefficients with respect to the shape changes

robust and efficient ways of moving the mesh and remeshing as needed. Special techniques to be used in conjunction with the DSD/SST method have been developed (see [32–34]) based on using higher-order functions (specifically NURBS basis functions) in time in representing the wing motion and deformation, mesh motion,

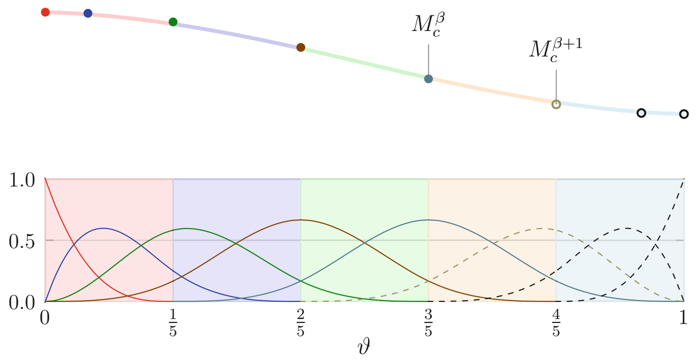


Fig. 7 Mesh motion is represented by using NURBS basis functions in time. The temporal-control meshes are the coefficients of the NURBS basis functions

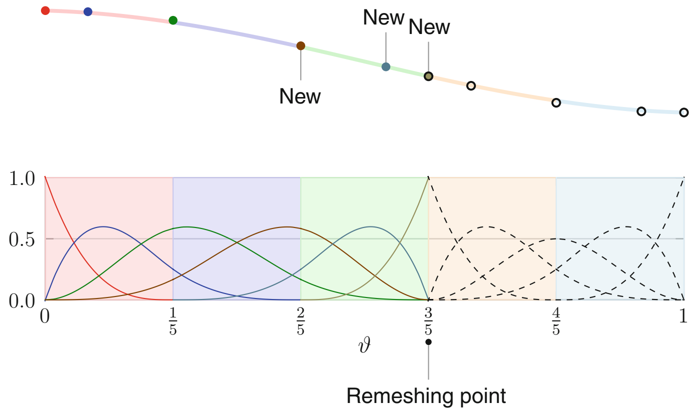


Fig. 8 Remeshing is handled by multiple knot insertion where we want to remesh. That point in time becomes a patch boundary

and remeshing. Using cubic NURBS basis functions in temporal representation of the wing position gives us a continuous representation of the acceleration, which in turn gives us a continuous representation of the aerodynamic forces. Using NURBS basis functions in temporal representation of the mesh motion (see Fig. 7) gives us a very effective way of dealing with moving meshes. This allows us to do mesh computations with longer time in between, but get the mesh-related information, such as the coordinates and their time derivatives, from the temporal representation whenever we need it. Figure 8 illustrates how remeshing is handled in this approach. We perform multiple knot insertions where we want to remesh, and that point in time becomes a patch boundary. More details on how temporal NURBS basis functions are used in mesh motion and remeshing can be found in [32, 33].

2.3 *Redistancing and Mass Conservation for the Level-Set Formulation*

When the ALE-VMS method is used in the context of the MITICT with the level-set formulation, additional computational technology is employed to enhance the accuracy and robustness of the free-surface flow formulation. The use of a regularized Heaviside function in the definition of the fluid density and viscosity necessitates the level set to satisfy the signed-distance function property near the air-water interface. To maintain the signed-distance property of the level set function, a *redistancing* procedure based on the Eikonal partial differential equation is employed. The details of the numerical formulation may be found in [12, 16, 77, 78].

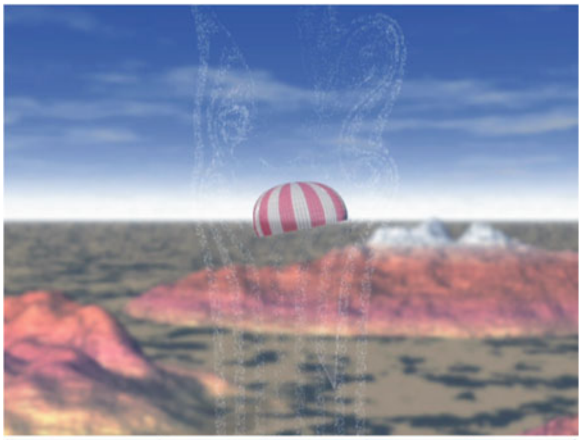
Furthermore, both convection and redistancing of the level set do not inherently conserve mass. Convergence to a mass-conserving solution occurs only with mesh refinement. Coarse (and not-so-coarse) mesh simulations may suffer form significant water mass loss. (This depends on the problem setup and boundary conditions. In the case of liquids sloshing in closed containers, mass loss may be significant. In problems with inflow and outflow boundaries the effect may not be as pronounced.) This effect is amplified when the equations are integrated for a long time period, when seemingly small mass errors for a given time step compound into a large mass error toward the end of the computation. As a result, an explicit mass correction procedure is necessary. To ensure mass balance at every time step, after redistancing of the level set, we modify the level set function by a *global constant*, such that the following equations holds:

$$\begin{aligned}&\int_{\Omega_{n+1}} \rho_{n+1}\, d\Omega - \int_{\Omega_n} \rho_n \,\mathrm{d}\Omega \\ &+ \Delta t_{n+1} \int_{\Gamma_{n+1/2}} \rho_{n+1/2} \left(\mathbf{u}^h_{n+1/2} - \mathbf{v}^h_{n+1/2}\right) \cdot n_{n+1/2}\, \mathrm{d}\Gamma = 0, \end{aligned} \tag{2}$$

where $\mathbf{v}^h$ is the mesh velocity. In Eq. (2), the quantities are subscripted with a temporal index and Δt_{n+1} is the time step size. This is the simplest technique that restores mass balance in the simulations. Other versions of mass correction are also possible: in [75, 85, 86] the authors proposed a total-domain based mass conservation technique, validated it experimentally in [87], and developed it in the context of MITICT (with mass conservation for fluid–solid interfaces) in [74]. A "chunk"-based (subdomain-based) version of mass conservation was developed in [29, 30].

3 Examples

Examples in Sects. 3.1–3.4 were computed with the DSD/SST methods, and the examples in Sects. 3.5–3.8 with the ALE-VMS methods.



	V_D (ft/s)	V_{RH} (ft/s)	T_B (s)	T_S (s)
Test Data	<10% Diff	Comparable	<10% Diff	<10% Diff
Computation	21.4	4 to 13	6.7	16.4

Fig. 9 Parachute shape and flow field at an instant during the computation and comparison with the test data. Here V_D, V_{RH}, T_B, and T_S are the descent speed, horizontal speed, breathing period, and swinging period

3.1 FSI Analysis of Spacecraft Parachutes

The first example, a parachute computation, serves the purpose of comparing our computed results to data from drop tests with a base parachute design and gaining confidence in our parachute FSI model. Figure 9 shows the parachute shape and flow field at an instant during the computation and the comparison with the test data. With that confidence, we can do simulation-based design studies [53], such as evaluating the aerodynamic performance of the parachute as a function of the suspension line length (see Fig. 10).

Spacecraft parachutes are typically used in clusters of two or three parachutes. The contact between the canopies of the parachute cluster is a computational challenge that we have addressed recently (see [53]). Figure 11 shows a cluster of three parachutes at three different instants during the FSI computation, with contact between two of the parachutes.

Spacecraft parachutes are also typically used in multiple stages, starting with a "reefed" stage where a cable along the parachute skirt constrains the diameter to be less than the diameter in the subsequent stage. After a certain period of time during the descent, the cable is cut and the parachute "disreefs" (i.e. expands) to the next stage. Computing the parachute shape at the reefed stage and FSI modeling during the disreefing involve additional computational challenges created by the increased geometric complexities and by the rapid changes in the parachute geometry. Figure 12 shows such a disreefing (see [55]).

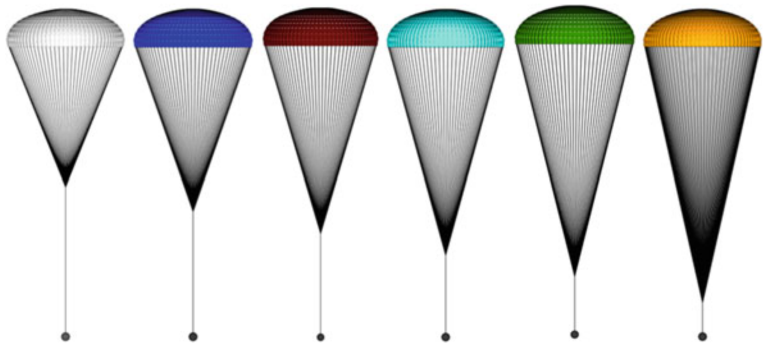


Fig. 10 A simulation-based parachute design study, where the objective is to evaluate the aerodynamic performance of the parachute as a function of the suspension line length. See [53] for details of the study

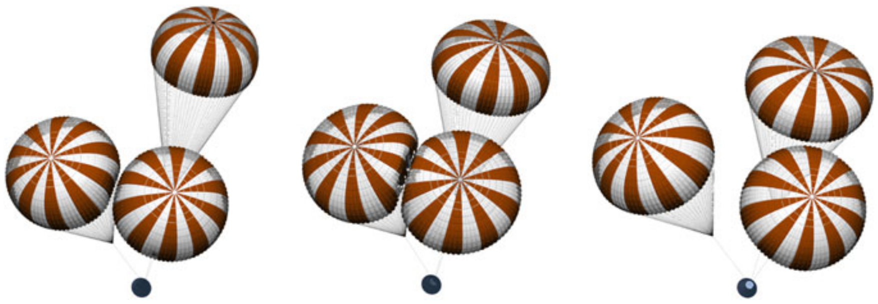


Fig. 11 A cluster of three parachutes at three instants during the FSI computation, with contact between two of the parachutes

As an additional computational challenge, the ringsail parachute canopy might, by design, have some of its panels and sails removed. The purpose is to increase the aerodynamic performance of the parachute. In FSI computation of parachutes with such "modified geometric porosity," the flow through the "windows" created by the removal of the panels and the wider gaps created by the removal of the sails cannot be accurately modeled with the HMGP and needs to be actually resolved. This challenge was successfully addressed in the computations reported in [57]. Figure 13 shows a cluster of three parachutes with modified geometric porosity, at an instant during the FSI computation.

3.2 Aerodynamic Analysis of Wind Turbines

Computer modeling of wind-turbine aerodynamics is challenging because correct aerodynamic torque calculation requires correct separation-point calculation, which requires an accurate flow field, which in turn requires good mesh resolution and

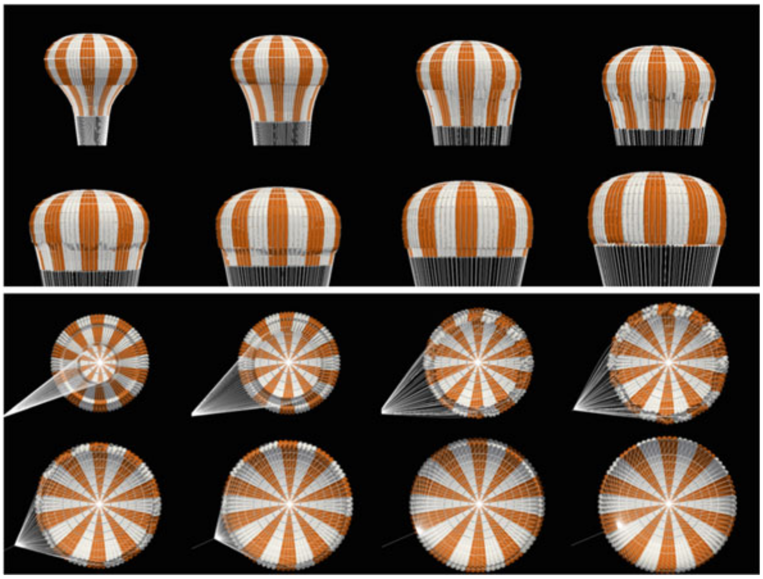
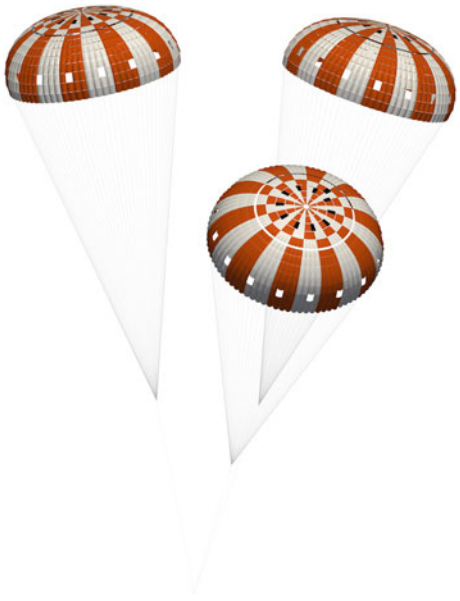


Fig. 12 Parachute disreefing from [55]: *side* and *bottom* views

Fig. 13 A cluster of three parachutes with modified geometric porosity, at an instant during the FSI computation reported in [57]



turbulence model. We describe from [35] computation of the aerodynamics of an actual wind-turbine rotor with the DSD/SST-SUPS and DSD/SST-VMST methods. Figure 14 shows time history of the aerodynamic torque generated by a single blade, as computed with the DST/SST-SUPS, DSD/SST-VMST, and ALE methods.

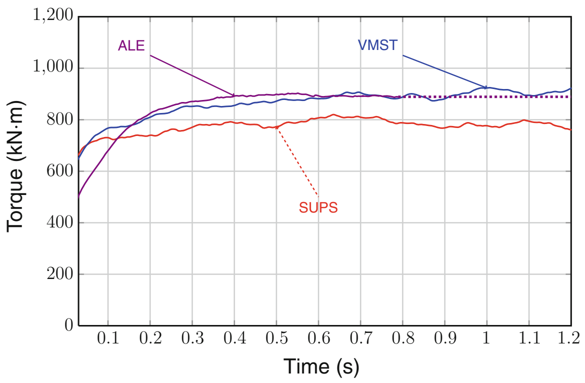


Fig. 14 Time history of the aerodynamic torque generated by a single blade. Computed with the DST/SST-SUPS ("SUPS"), DST/SST-VMST ("VMST"), and ALE methods

Including the tower in the model increases the computational challenge because of the fast, rotational relative motion between the rotor and tower. We address this additional challenge in [37] by using NURBS basis functions for the temporal representation of the rotor motion, mesh motion and also in remeshing. This is essentially the same computational technology described in Sect. 2.2 for modeling the aerodynamics of flapping wings. We named this "ST/NURBS Mesh Update Method (STNMUM)" in [37]. Figure 15 shows, from [37], the vorticity magnitude, computed with the DST/SST-VMST method and the STNMUM. In that figure, the color range from blue to red corresponds to a vorticity range from low to high, and lighter and darker shades of a color correspond to lower and higher values.

3.3 Patient-Specific FSI Analysis of Cerebral Arteries with Aneurysm

Patient-specific arterial FSI modeling has many challenges. They include calculating an estimated zero-pressure arterial geometry, specifying the velocity profile at an inflow with non-circular shape, using variable wall thickness, building layers of refined fluid mesh near the walls, proper calculation of the wall shear stress (WSS) and oscillatory shear index (OSI), and properly scaling the flow rate at the inflow. Special techniques developed to address these challenges can be found in [50]. Here we present some computations from [50] for cerebral arteries with aneurysm. Figure 16 shows the lumen obtained from voxel data for three arterial models: Model 1, Model 2, and Model 3. Figure 17 shows the fluid mechanics mesh for Model 3. Figure 18 shows the streamlines at the maximum flow rate.

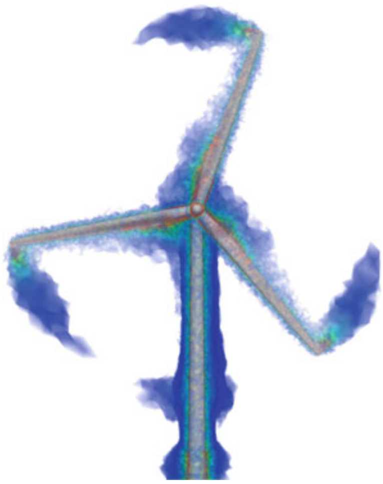


Fig. 15 Vorticity, computed with the DST/SST-VMST method and the STNMUM (see [37])

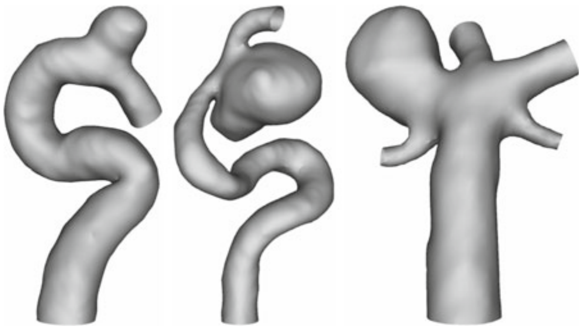


Fig. 16 Arterial lumen geometry obtained from voxel data for Model 1, Model 2, and Model 3

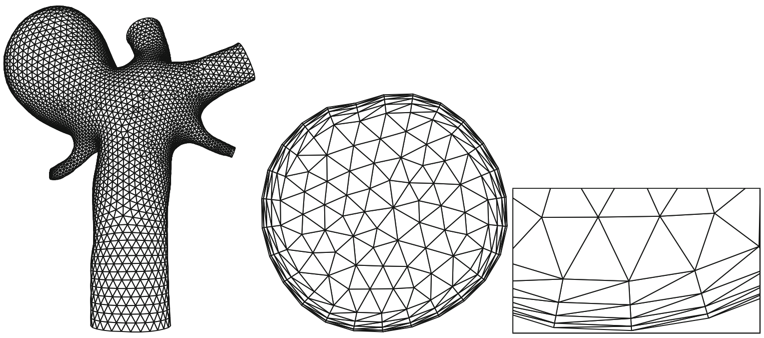


Fig. 17 Fluid mechanics mesh for Model 3. Mesh at the fluid–structure interface and inflow plane

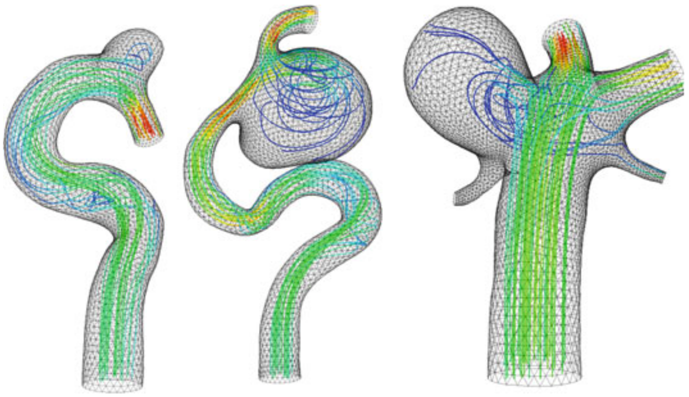


Fig. 18 Streamlines for the three models when the volumetric flow rate is maximum

3.4 Aerodynamic Analysis of Flapping Wings of an Actual Locust and an MAV

As a last set of examples from analyses with the ST methods, we present from [33, 34] computational aerodynamics modeling of flapping wings of an actual locust and an MAV. The motion and deformation data for the wings is extracted from the high-speed, multi-camera video recordings of a locust in a wind tunnel at Baylor College of Medicine (BCM), Houston. The video recording is accomplished by using a set of tracking points marked on the forewings (FW) and hintings (HW) of the locust. The tracking points are seen in Fig. 19. How the wing motion and deformation data is extracted from the video data and represented using NURBS basis functions in space and time is described in detail in [33]. Figures 20 and 21 show the wind tunnel photographs and the computational model at eight points in time. Figure 22 shows how the body and wings compare for the locust and MAV models, and Fig. 23 shows the length scales involved in the computations with those models. Figure 24 shows the streamlines for the locust. Figures 25 and 26 show for the locust the vorticity magnitude during the second flapping cycle. Figures 27 and 28 show for the MAV the vorticity magnitude during the third flapping cycle. In Figs. 25, 26, 27 and 28, the color range from blue to red corresponds to a vorticity range from low to high, and lighter and darker shades of a color correspond to lower and higher values. Figure 29 shows the lift and thrust for the locust and MAV.

3.5 The MARIN Dam Break Problem

The setup of the dam break problem, initially proposed by the Maritime Research Institute Netherlands (MARIN) [88], is depicted in Fig. 30, and is taken from [12]. The problem consists of a column of water, initially at rest, that collapses under the

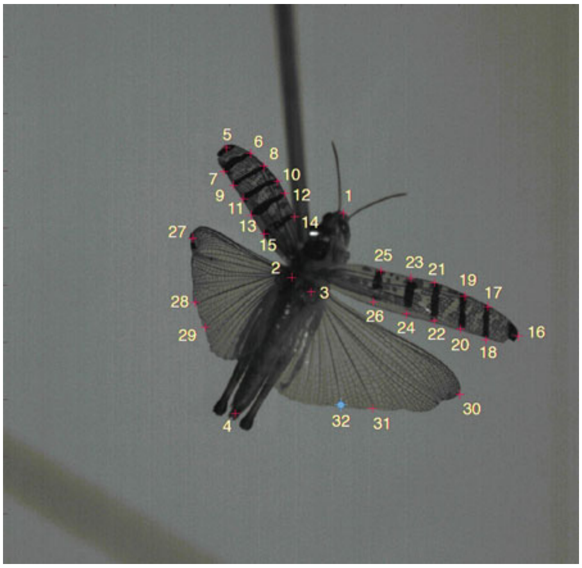


Fig. 19 Tracking points in the data set from the BCM wind tunnel

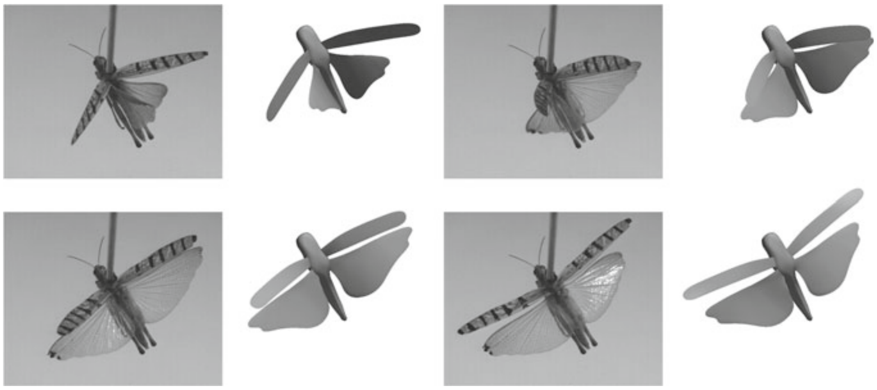


Fig. 20 Comparison of computational model and wind tunnel photographs at first four points in time. Viewing angles are matched approximately. Wind tunnel photographs are from BCM

action of gravity and impacts a fixed rectangular container. We compute the problem using two types of the spatial discretization: linear tetrahedral finite elements and NURBS. The quadratic NURBS mesh is significantly more coarse than the linear tetrahedral mesh. Free-slip and no-penetration boundary conditions are applied on all surfaces, including the top of the tank. The problem is run until $T = 6$ s. Snapshots comparing the solutions coming from tetrahedral FEM and NURBS computations are given in Fig. 31. Large-scale features of the solution are very similar in the two simulations, however the details of the small-scale features are better represented on

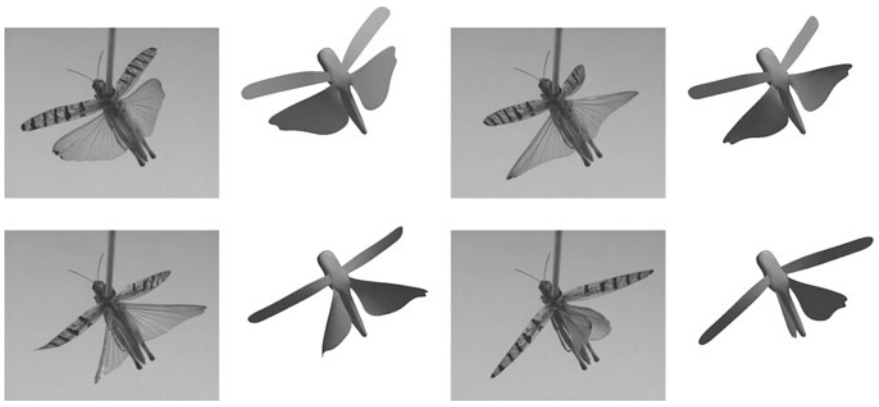


Fig. 21 Comparison of computational model and wind tunnel photographs at last four points in time. Viewing angles are matched approximately. Wind tunnel photographs are from BCM

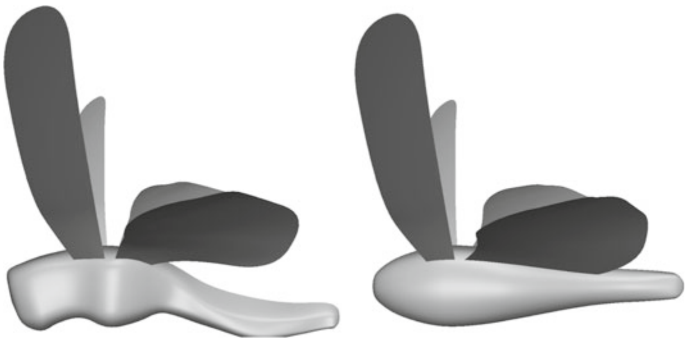


Fig. 22 Locust body and wings (*left*) and MAV body and wings (*right*)

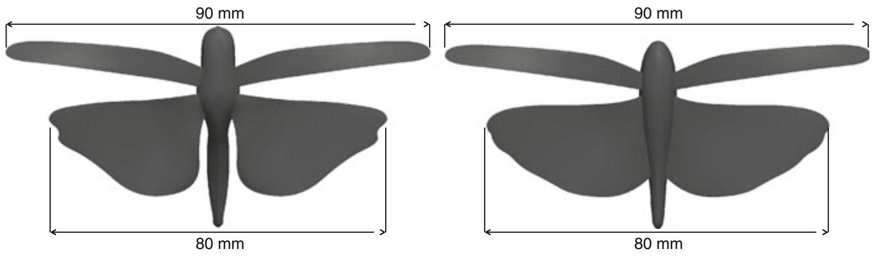


Fig. 23 Length scales in the computations with the locust (*left*) and MAV (*right*) models

a much finer tetrahedral grid, as expected. Time series of the pressure at different locations on the obstacle are shown in Fig. 32. The first wave hits the block at approximately $t = 0.5$ s, and the second, much smaller wave arrives at the block

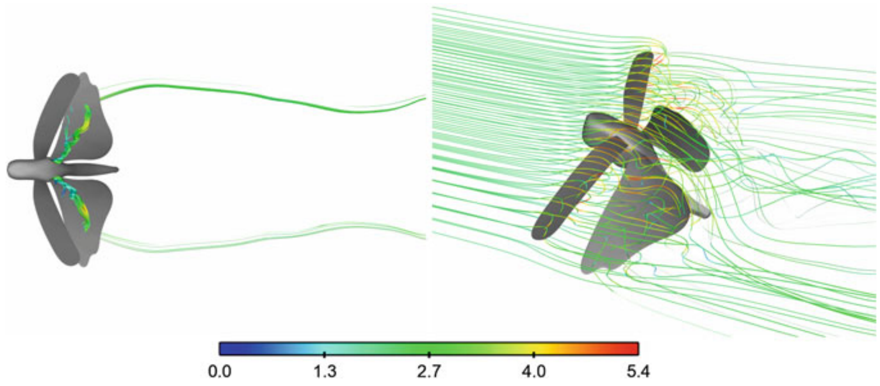


Fig. 24 Locust. Streamlines colored by velocity magnitude in m/s at approximately 25 % (*left*) and 50 % (*right*) of the second flapping cycle

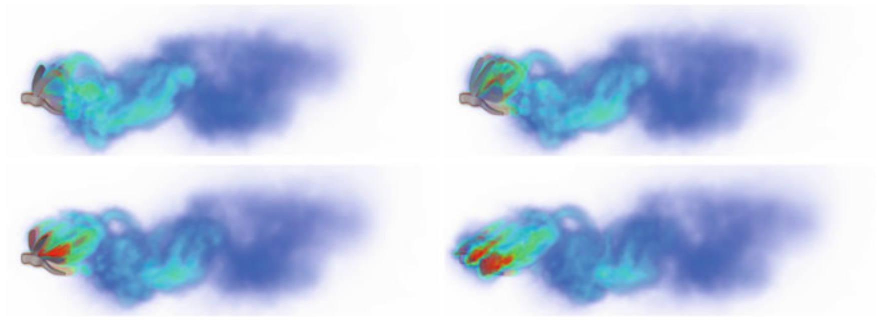


Fig. 25 Locust. Vorticity for the first 4 of 8 equally-spaced points during the 2nd flapping cycle

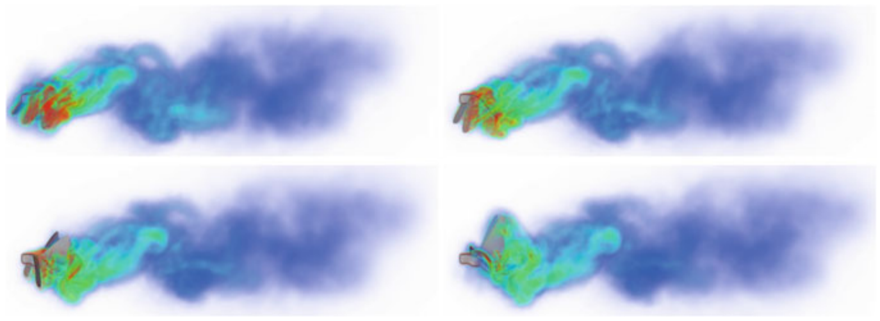


Fig. 26 Locust. Vorticity for the last 4 of 8 equally-spaced points during the 2nd flapping cycle

at about $t = 5$ s. The wave impact times and pressure peaks are predicted very well with both linear elements and quadratic NURBS. Given that the NURBS mesh has about half of the degrees-of-freedom of the linear FEM mesh in each Cartesian

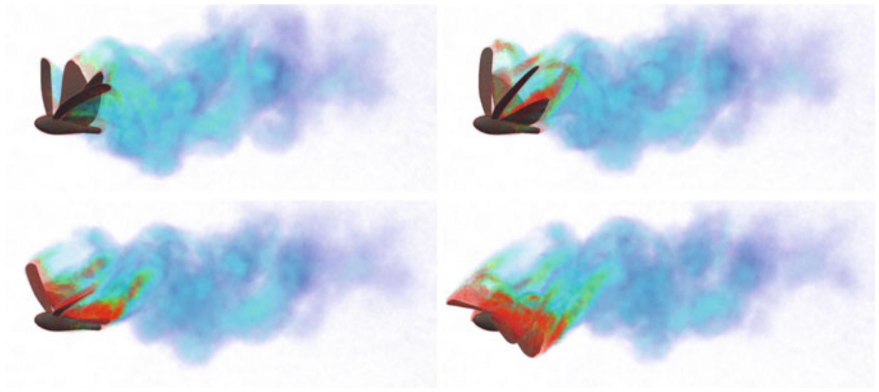


Fig. 27 MAV. Vorticity for the first 4 of 8 equally-spaced points during the 2nd flapping cycle

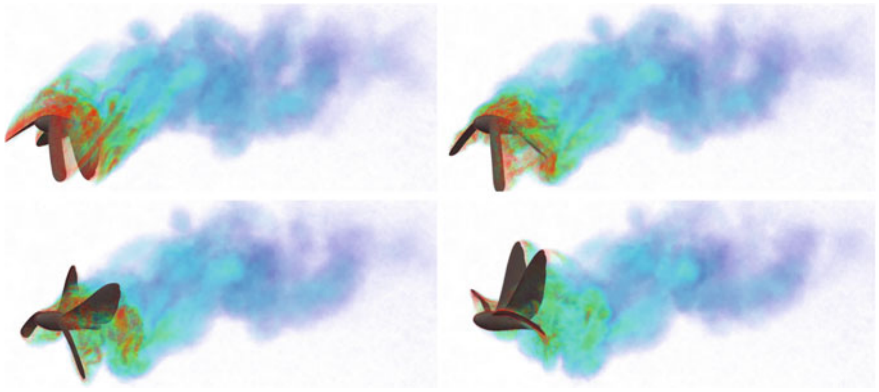


Fig. 28 MAV. Vorticity for the last 4 of 8 equally-spaced points during the 2nd flapping cycle

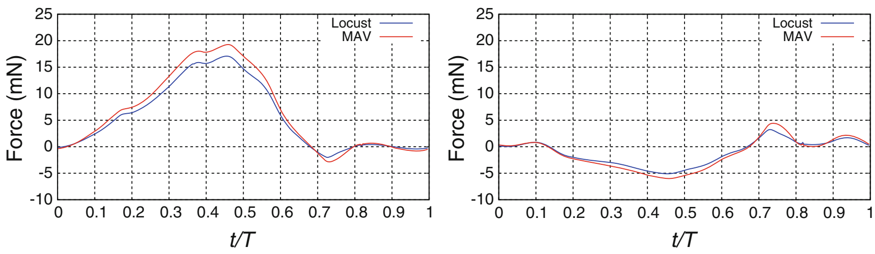


Fig. 29 Total lift (*left*) and thrust (*right*) generated over one cycle

direction, the accuracy of NURBS results is remarkable; linear FEM is not capable of attaining such accuracy at this level of resolution (see [12]), and requires a finer mesh for comparable accuracy.

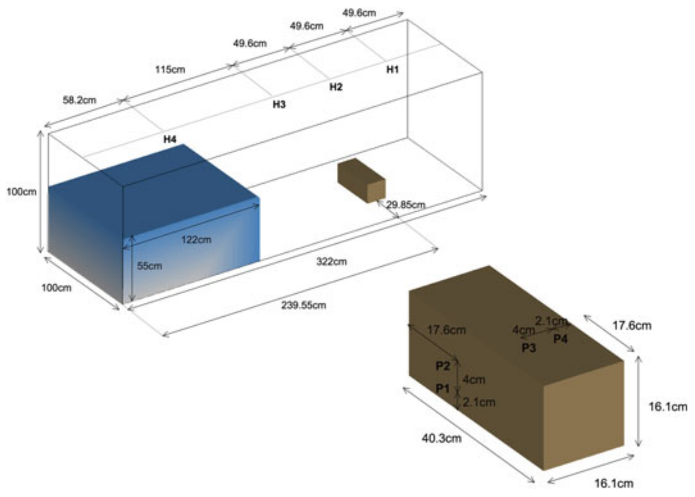


Fig. 30 The MARIN dam break problem. Geometry definition. The computational domain is a rectangular box with dimensions 3.22 m × 1 m × 1 m. The object has dimensions 0.2 m × 0.2 m × 0.4 m and is placed at the back end of the tank. The water column, initially at rest, has dimensions 1 m × 1 m × 0.55 m. The locations where pressure and water height are sampled are also depicted

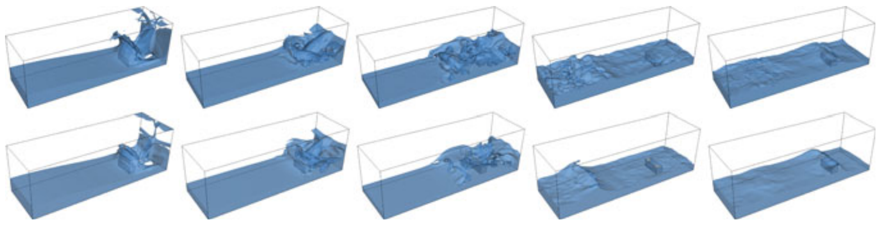


Fig. 31 The MARIN dam break problem. Snapshots of the free surface solution on the tetrahedral (*top*) and NURBS (*bottom*) meshes at $t = 1.0, 1.5, 2.0, 4.0,$ and 5.0 s

3.6 Fridsma Planing Hull

We present results for the Fridsma planing hull [89]. We give a detailed definition of the hull geometry, present a mesh refinement study, and assess the effect of hull speed on the drag force and trim angle. Only flat-water (i.e., no waves), constant hull speed cases are considered. The computational results presented are from [16]. The Fridsma hull geometry definition is given in Fig. 33. The hull is comprised of idealized shapes: a bow consisting of four ruled surfaces followed by a wedge-shaped straight section with an constant deadrise angle of 20°. Analytical expressions for the bounding curves for the ruled surfaces are provided in the figure. The relevant global geometry parameters are, Length (L): 114.3 cm, Beam (b): 22.86 cm, Height: 14.2875 cm, and Deadrise: 20°. The hull mass, center of gravity, and moment of inertia are, Mass (m): 7.257 kg, x_{cg}: 80.01 cm, z_{cg}: 6.721 cm, Gyradius (r): 25 % L,

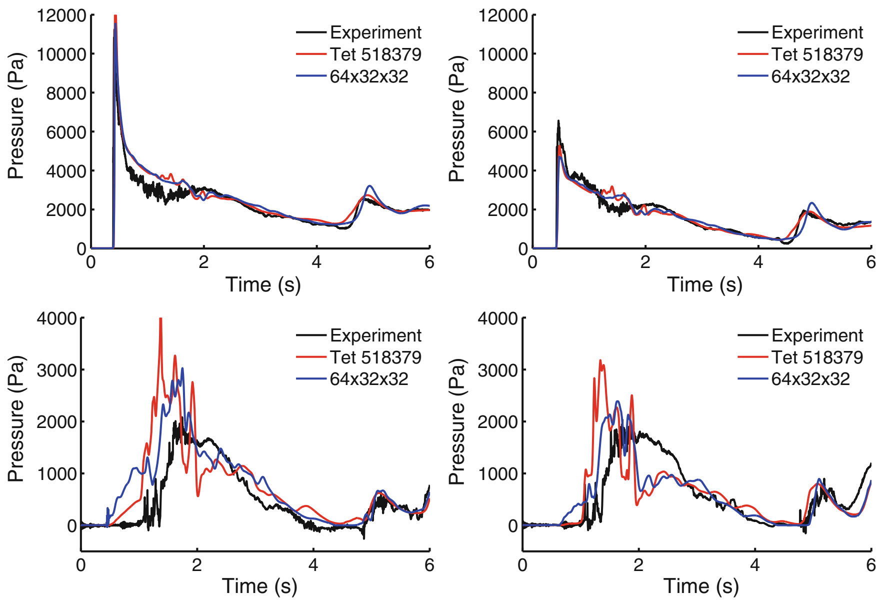


Fig. 32 The MARIN dam break problem. Time history of the pressure at four locations on the obstacle. Experimental data is from [88]

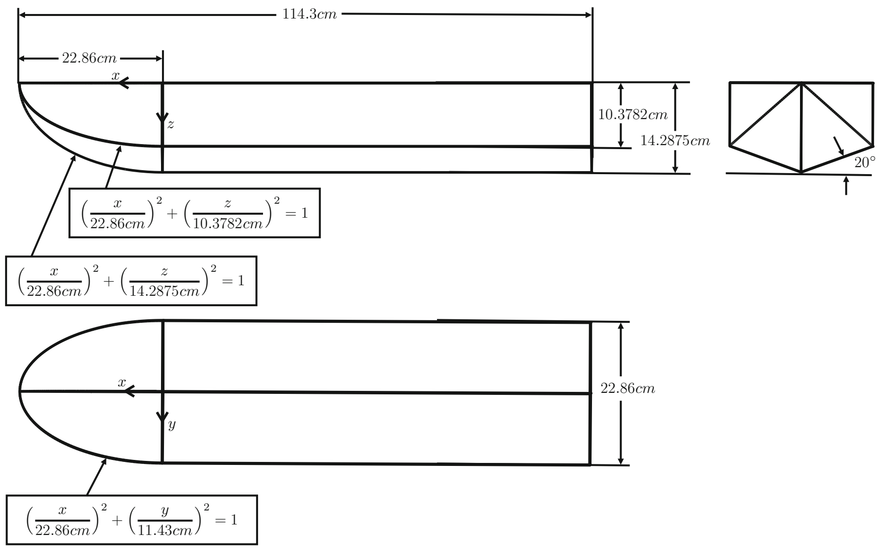

114.3cm
22.86cm
x
z
10.3782cm
14.2875cm
20°
$\left(\frac{x}{22.86cm}\right)^2 + \left(\frac{z}{10.3782cm}\right)^2 = 1$
$\left(\frac{x}{22.86cm}\right)^2 + \left(\frac{z}{14.2875cm}\right)^2 = 1$
y
22.86cm
$\left(\frac{x}{22.86cm}\right)^2 + \left(\frac{y}{11.43cm}\right)^2 = 1$

Fig. 33 Fridsma hull. Geometry definition

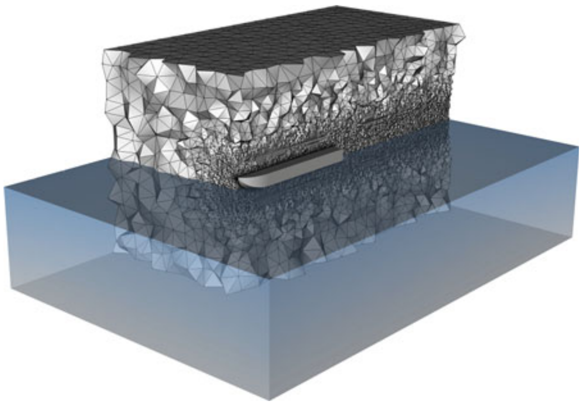


Fig. 34 Fridsma hull. Coarsest mesh with water and air domains shown

and $I_{yy} = mr^2$: 0.6165 kg m^2. This data pertains to the center of gravity located at 70 % of the hull length, measured from the aft of the hull [89].

We perform a mesh convergence study at Froude number $Fr = 0.8950$.[1] The Froude number is defined as $Fr = \frac{u}{\sqrt{gL}}$, where u is the hull speed, g is the magnitude of gravitational acceleration, L is the hull length. At this chosen Froude number, according to [89], the trim angle reaches its maximum. A convergence study was performed on a sequence of four meshes. The coarsest mesh is shown in Fig. 34. The figure also shows the water and air subdominant in the undisturbed configuration. The mesh is dense near the hull surface and in the wake. The hull is fixed in the direction of travel, and the corresponding velocity is set at the inflow of the computational domain together with the level set function. The hull is allowed to pitch, and displace in the vertical direction. At the outflow a hydrostatic pressure profile is imposed as a traction boundary condition. On the side, bottom, and top boundaries of the computational domain free-slip boundary conditions are imposed. Figure 35 shows the deformed free surface colored by the flow speed relative to the hull speed. The hull rises up and develops a trim angle such that the bow is higher than the aft. Note the presence of the "rooster tail" feature, which is typical for planing hulls. Also note that the rooster tail feature goes all the way to the outflow boundary, which suggests that a longer-domain simulation may be needed in the future. Figure 36 shows convergence of the drag force and trim angle. The drag force is non-dimensionalized by the gravitational force. From the results we see that the drag force converges quickly to the experimental value. On the other hand, the trim angle is underestimated by 12 % with respect to the experimental data, and does not improve with mesh refinement. Possible causes may be the choice downstream, lateral, and bottom boundary locations. Error in the experimental data is also possible.

We also examine the effect of the hull speed on the drag force and trim angle is studied. In addition to the $Fr = 0.8950$ case, we consider $Fr = 0$, $Fr = 0.5925$,

[1] In Fridsma [89] the results are reported in terms of the Speed-Length Ratio (SLR), $u/\sqrt{L}$, which is a dimensional quantity. Here we chose to report the results in terms of the Froude number.

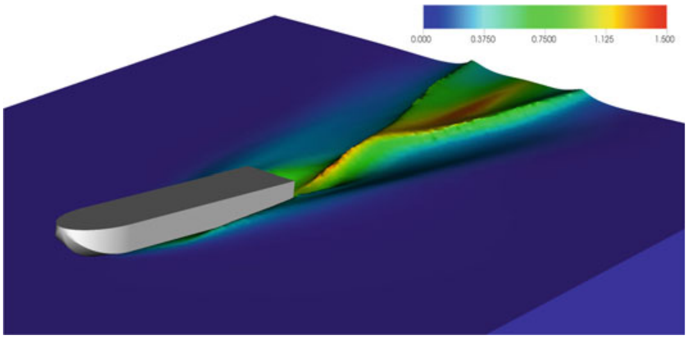


Fig. 35 Fridsma hull. Free surface colored by the flow speed relative to the hull speed in m/s

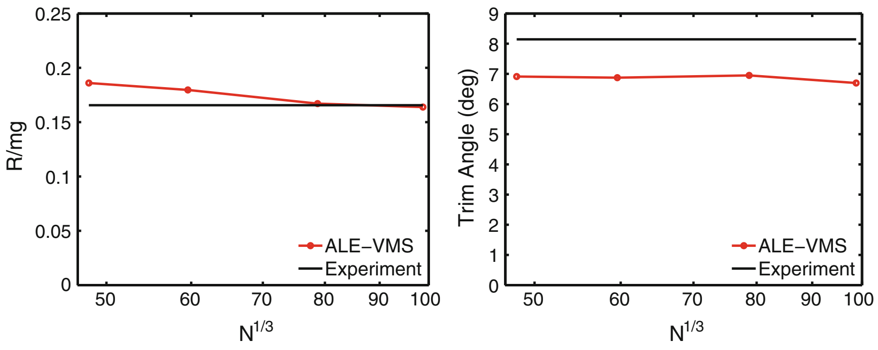


Fig. 36 Fridsma hull. Convergence of the drag force (*left*) and trim angle (*right*) with mesh refinement and comparison with experimental results

and $Fr = 1.190$ cases. The simulations are started impulsively in the configuration depicted in Fig. 34. In the case of $Fr = 0$, although the hull speed is zero, a non-zero trim angle develops such that the hull is in equilibrium with the hydrostatic forces. In all other cases, there is a rapid transient followed by a largely steady-state response. The steady-state drag force and trim angle are plotted as a function of Froude number, and compared to the experimental results in Fig. 37. Accurate prediction of the drag force is attained in all cases. The trim angle is predicted very well for the first two Froude number cases, and a deviation from the experiment by 10–12 % is seen in the remaining two cases.

3.7 DTMB 5415 Navy Combatant in Head Waves

Here we present the simulation of the DTMB 5415 Navy combatant at lab scale from [78]. This ship has been investigated by other researchers, both experimentally and computationally (see, e.g., [90–92]). The length of the ship hull is 5.72 m. The ship mass, center of gravity and inertia tensor are computed by meshing the ship

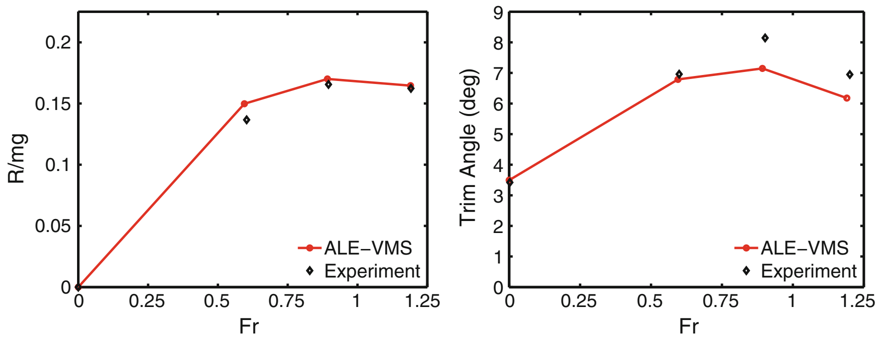


Fig. 37 Fridsma hull. Steady-state drag force (*left*) and trim angle (*right*) as a function of Froude number. Comparison with experimental results

interior and performing a direct computation. The total ship volume is 1,366 m^3. The ship mass is equal to 532.3 kg. It is obtained by multiplying the volume of the ship below the water line by the constant water density. The center of gravity and the inertia tensor are computed assuming the ship's effective density (i.e., the ship mass divided by its total volume), which results in $\mathbf{X}_0 = (2.761, 0, 0.280)\,\mathrm{m}$ and

$$\mathbf{J}_0 = \begin{bmatrix} 7.256\text{E-}2 & 2.69\text{E-}7 & 5.35\text{E-}2 \\ 2.69\text{E-}7 & 2.89 & -2.44\text{E-}8 \\ 5.35\text{E-}2 & -2.44\text{E-}8 & 2.91 \end{bmatrix} \text{kg m}^2. \tag{3}$$

We compute the ship in head waves, meaning the waves that travel in the direction opposite to that of the ship. We assume that the ship speed is $U_{in} = 1.873\,\mathrm{m/s}$, which gives $Fr = 0.25$ based on the ship length. The ship was allowed to move vertically, to pitch and to roll, while the rest of the rigid body degrees-of-freedom were constrained. We make use of the linear Airy waves [93] to prescribe inlet boundary conditions. The Airy waves may be derived using potential theory, and are specified as follows: Given, the wave amplitude, wave length and water depth, $A_w = 0.2\,\mathrm{m}$, $L_w = 5.72\,\mathrm{m}$ and $h = 3.49$ m, respectively, we compute $k = 2\pi/L_w$, the angular wavenumber, $\omega = \sqrt{gk\tanh(kh)}$, the wave phase speed, and $A_v = \frac{\omega A_w}{\sinh(kh)}$, the velocity amplitude. With these definitions, the Airy waves are given by

$$u = A_v \cosh(kz)\cos(kx - \omega t) + U_{in} \tag{4}$$

$$v = 0 \tag{5}$$

$$w = A_v \sinh(kz)\sin(kx - \omega t) \tag{6}$$

$$\phi = A_w \cos(kx - \omega t) + h - z, \tag{7}$$

where $(u, v, w)^T$ is the fluid velocity vector and the air-water interface in the hydrostatic configuration is assumed to be located at $z = 0$.

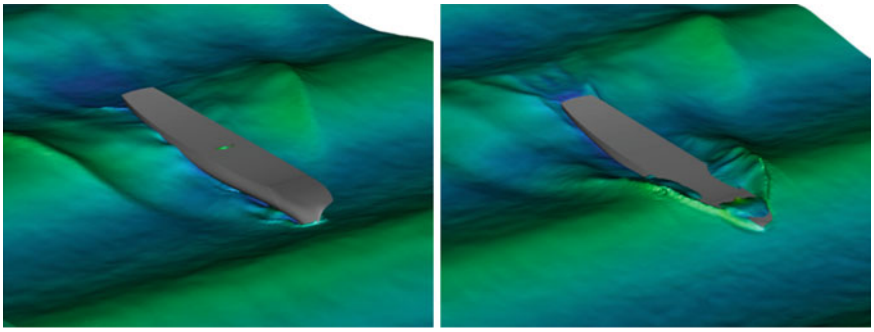


Fig. 38 DTMB 5415 in head waves at t = 9.0 and 9.5 s. Water surface colored by the fluid speed

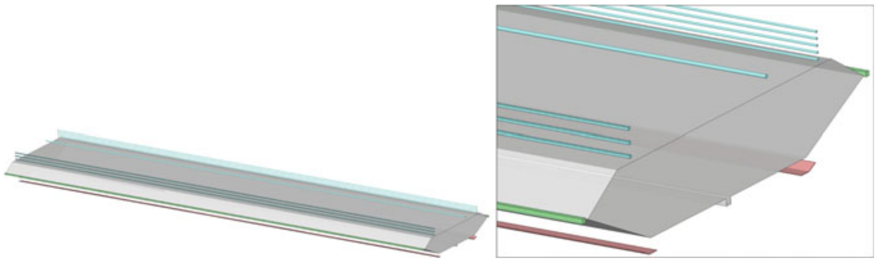


Fig. 39 Geometric model of the scaled Hardanger bridge deck section and zoom on the geometric details of the bridge deck section model. The guide-vane-like vortex mitigation devices are located on the underside of the deck and are shown in *light red* color

Figure 38 shows the ship negotiating high-amplitude waves. The right part of Fig. 38 shows the ship partially submerged in water, which is a result of the oncoming wave hitting the bow of the ship. In this case, near the bow, the free surface experiences topological changes, which necessitates the use of an interface-capturing method to handle the air-water interface for this class of problems.

3.8 Vortex-Induced Vibrations of a Bridge Deck

Here we present an example of a fluid–object interaction simulation using a scaled model of the Hardanger bridge deck section [94]. The bridge deck geometric model is shown in Fig. 39. This study was initiated to examine the effect of the guide-vane-like vortex mitigation devices (VMDs) installed on the underside of the bridge deck (see Fig. 39 for a zoom on the guide vanes) on the resulting wind aerodynamics and structural response of the bridge. The height h, width b, and length l of the deck scaled model are 0.0666, 0.366 and 1.7 m, respectively. The bridge deck section computational domain is shown in Fig. 40. The locations of the top, bottom, and lateral walls are coincident with those of the wind tunnel where the experiments took

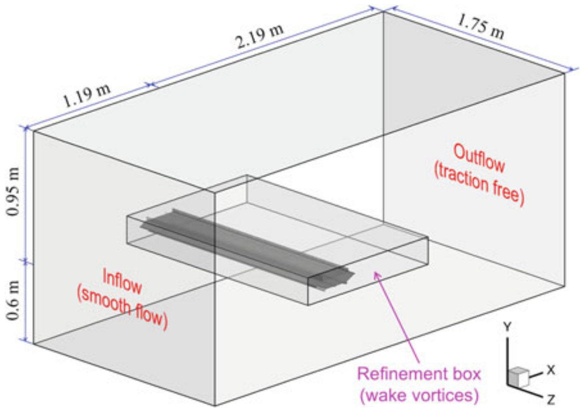


Fig. 40 Bridge deck section computational domain and boundary conditions

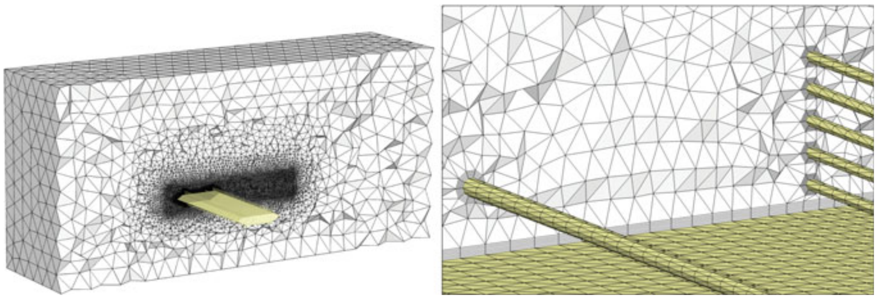


Fig. 41 Aerodynamics mesh of the bridge deck section and zoom on the boundary layer mesh of the *top* deck and rails

place. Note that there is only a 2.5 cm gap between the tunnel wall and the side of the bridge deck section. All the geometric details of the model-scale bridge deck are modeled in the computations, including the hand and bicycle rails on the top of the deck, and the maintenance rails in the front and rear of the deck. Computations are performed for 2.6 and 6.0 m/s wind speed, with and without the VMDs. Figure 41 shows the mesh resolution used in this study. Boundary-layer prismatic elements are used near all solid surfaces, and tetrahedral elements are used elsewhere in the computational domain. The mesh is refined near the deck and downstream of it to better capture the wake turbulence. The uniform wind speed is prescribed at the inflow boundary, the traction vector is set to zero at the outflow boundary, and the slip condition is set on the top, bottom, and lateral boundaries of the computational domain (see Fig. 40). The no-slip boundary condition on the bridge deck surface is enforced weakly. The bridge deck is modeled as a rigid object. For the bridge deck mass, moment of inertia tensor, and stiffness and damping matrices the readers are

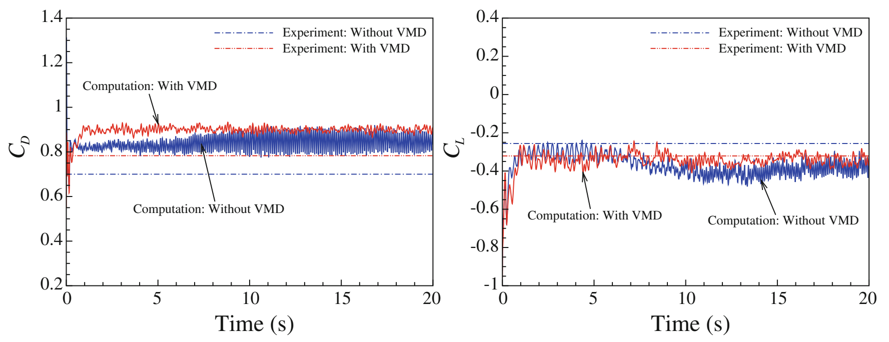


Fig. 42 Time history of the drag and lift coefficients for cases with and without VMDs for 2.6 m/s wind speed. Time-averaged experimental measurements from [94] are plotted for comparison

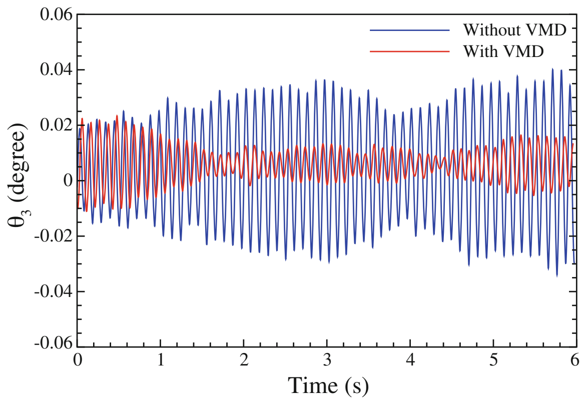


Fig. 43 Time history of the pitching angle with and without VMDs for 6.0 m/s wind speed

referred to [94]. The deck is allowed to displace vertically, and undergo pitching and rolling motions.

Figure 42 shows drag and lift coefficients for cases with and without the VMDs. The drag and lift coefficients are defined as $C_D = \frac{F_D}{\frac{1}{2}\rho U^2 hl}$ and $C_L = \frac{F_L}{\frac{1}{2}\rho U^2 bl}$. Results are compared with the experimental measurements from [94] and reasonable agreement is achieved. Figure 43 shows the time history of angular displacement of the bridge deck corresponding to the pitching motion. The figure clearly shows that with the added VMDs the bridge deck experiences smaller rotational motions then without, which was also observed in the wind tunnel tests. To better understand the underlying mechanics, the differences in the air flow with and without VMDs are shown on a planar cut of the bridge deck in Fig. 44. The guide vanes keep the flow attached to the underside of the deck, which delays flow separation and precludes formation of large-scale vortical structures that drive the bridge deck response. Figure 45 shows the 3D view of the deck with guide vanes, where air speed contours at an instant are

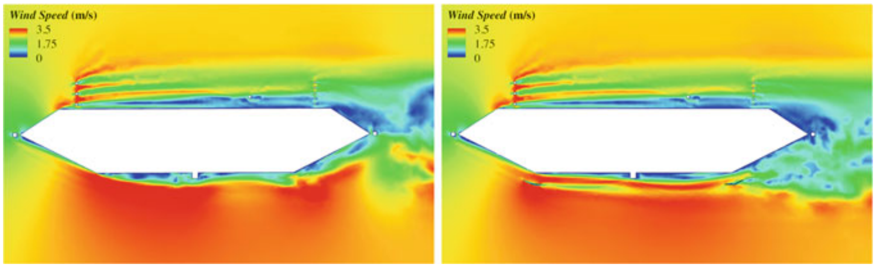


Fig. 44 Instantaneous air speed contours on a planar cut near the bridge deck for 2.6 m/s wind speed. *Left* Case without VMDs. *Right* Case with VMDs

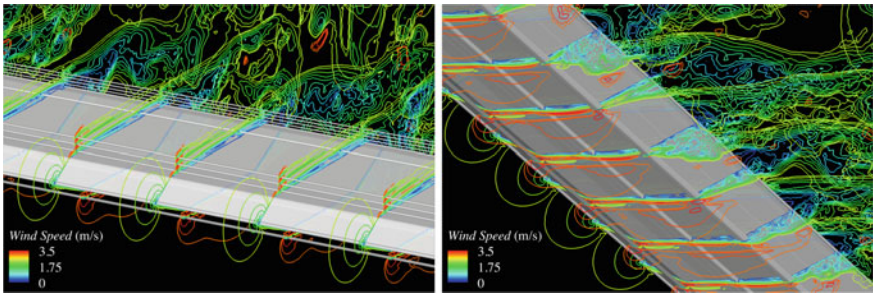


Fig. 45 Instantaneous air speed contours on a set of cuts along the deck length for 2.6 m/s wind speed. Top and bottom deck views are shown

plotted on a set of cuts along the deck length. Top and bottom views are shown. The flow is turbulent and 3D, which underscores the importance of 3D aerodynamics modeling for this class of problems.

4 Concluding Remarks

Bringing solution and analysis to specific classes of moving-interface problems with a common computational technology need motivated the development of our core ALE-VMS and ST methods, their recent versions, and the special ALE-VMS and ST techniques targeting specific classes of problems, such as parachute FSI, aerodynamics of flapping wings, wind-turbine aerodynamics and FSI, and free-surface flow and FOI for ship hydrodynamics. We presented an overview of how the core and special ALE-VMS and ST techniques are used in computational engineering analysis and design. We included an overview of three of the special ALE-VMS and ST techniques, just as examples of the many special techniques that complement the core methods. We presented examples of different classes of challenging problems solved: spacecraft parachute FSI, ship hydrodynamics, wind-turbine aerodynamics, patient-specific arterial FSI, aerodynamics of flapping wings of an actual locust and an MAV, and vortex-induced vibrations of a bridge deck section. In some of the problems, we included a comparison with the experimental data, and the comparison was always favorable. The examples show that in a diverse set of engineering applications, with the scope and power afforded by the core and special ALE-VMS and ST techniques, we can provide reliable analysis and support the design process.

Acknowledgments This work was supported in part by NASA JSC Grant NNX13AD87G. Method development and evaluation components of the work on aerodynamics of flapping wings and wind-turbine aerodynamics were supported in part by ARO Grant W911NF-12-1-0162 (TT) and Rice–Waseda research agreement (KT). The development and application of FOI techniques for bridge aerodynamics was supported by the program for preferred research areas at the Faculty of Engineering Science and Technology, the Norwegian University of Science and Technology. The research work on free-surface FOI was supported by the ARO Grant W911NF-11-1-0083 (YB). We wish to thank the Texas Advanced Computing Center (TACC) at the University of Texas at Austin, the San Diego Supercomputer Center (SDSC) at the University of California, San Diego, and the Norwegian Metacenter for Computational Science (Notur) for providing some of the HPC resources used. We thank Professor Fabrizio Gabbiani and Dr. Raymond Chan (Baylor College of Medicine) for providing us the digital data extracted from the wind-tunnel videos of the locust.

References

1. Tezduyar TE (1992) Stabilized finite element formulations for incompressible flow computations. Adv Appl Mech 28:1–44. doi:10.1016/S0065-2156(08)70153-4
2. Tezduyar TE (2003) Computation of moving boundaries and interfaces and stabilization parameters. Int J Numer Methods Fluids 43:555–575. doi:10.1002/fld.505
3. Tezduyar TE, Sathe S (2007) Modeling of fluid–structure interactions with the space–time finite elements: solution techniques. Int J Numer Methods Fluids 54:855–900. doi:10.1002/fld.1430
4. Takizawa K, Tezduyar TE (2011) Multiscale space–time fluid–structure interaction techniques. Comput Mech 48:247–267. doi:10.1007/s00466-011-0571-z
5. Takizawa K, Tezduyar TE (2012) Space–time fluid–structure interaction methods. Math Models Methods Appl Sci 22:1230001. doi:10.1142/S0218202512300013
6. Hughes TJR, Liu WK, Zimmermann TK (1981) Lagrangian-Eulerian finite element formulation for incompressible viscous flows. Comput Methods Appl Mech Eng 29:329–349
7. Ohayon R (2001) Reduced symmetric models for modal analysis of internal structural-acoustic and hydroelastic-sloshing systems. Comput Methods Appl Mech Eng 190:3009–3019
8. van Brummelen EH, de Borst R (2005) On the nonnormality of subiteration for a fluid–structure interaction problem. SIAM J Sci Comput 27:599–621
9. Bazilevs Y, Calo VM, Hughes TJR, Zhang Y (2008) Isogeometric fluid–structure interaction: theory, algorithms, and computations. Comput Mech 43:3–37
10. Bazilevs Y, Hsu M-C, Akkerman I, Wright S, Takizawa K, Henicke B, Spielman T, Tezduyar TE (2011) 3D simulation of wind turbine rotors at full scale. Part I: geometry modeling and aerodynamics. Int J Numer Methods Fluids 65:207–235. doi:10.1002/fld.2400
11. Bazilevs Y, Hsu M-C, Kiendl J, Wüchner R, Bletzinger K-U (2011) 3D simulation of wind turbine rotors at full scale. Part II: fluid–structure interaction modeling with composite blades. Int J Numer Methods Fluids 65:236–253
12. Akkerman I, Bazilevs Y, Kees CE, Farthing MW (2011) Isogeometric analysis of free-surface flow. J Comput Phys 230:4137–4152

13. Hsu M-C, Bazilevs Y (2011) Blood vessel tissue prestress modeling for vascular fluid–structure interaction simulations. Finite Elem Anal Des 47:593–599
14. Nagaoka S, Nakabayashi Y, Yagawa G, Kim YJ (2011) Accurate fluid–structure interaction computations using elements without mid-side nodes. Comput Mech 48:269–276. doi:10.1007/s00466-011-0620-7
15. Bazilevs Y, Hsu M-C, Takizawa K, Tezduyar TE (2012) ALE-VMS and ST-VMS methods for computer modeling of wind-turbine rotor aerodynamics and fluid–structure interaction. Math Models Methods Appl Sci 22:1230002. doi:10.1142/S0218202512300025
16. Akkerman I, Dunaway J, Kvandal J, Spinks J, Bazilevs Y (2012) Toward free-surface modeling of planing vessels: simulation of the fridsma hull using ALE-VMS. Comput Mech 50:719–727
17. Minami S, Kawai H, Yoshimura S (2012) Parallel BDD-based monolithic approach for acoustic fluid–structure interaction. Comput Mech 50:707–718
18. Miras T, Schotte J-S, Ohayon R (2012) Energy approach for static and linearized dynamic studies of elastic structures containing incompressible liquids with capillarity: a theoretical formulation. Comput Mech 50:729–741
19. van Opstal TM, van Brummelen EH, de Borst R, Lewis MR (2012) A finite-element/boundary-element method for large-displacement fluid–structure interaction. Comput Mech 50:779–788
20. Yao JY, Liu GR, Narmoneva DA, Hinton RB, Zhang Z-Q (2012) Immersed smoothed finite element method for fluid–structure interaction simulation of aortic valves. Comput Mech 50:789–804
21. Larese A, Rossi R, Onate E, Idelsohn SR (2012) A coupled PFEM–Eulerian approach for the solution of porous fsi problems. Comput Mech 50:805–819
22. Bazilevs Y, Takizawa K, Tezduyar TE (2013) *Computational fluid–structure interaction: methods and applications*. Wiley, Chichester
23. Korobenko A, Hsu M-C, Akkerman I, Tippmann J, Bazilevs Y (2013) Structural mechanics modeling and FSI simulation of wind turbines. Math Models Methods Appl Sci 23:249–272
24. Yao JY, Liu GR, Qian D, Chen CL, Xu GX (2013) A moving-mesh gradient smoothing method for compressible CFD problems. Math Models Methods Appl Sci 23:273–305
25. Kamran K, Rossi R, Onate E, Idelsohn SR (2013) A compressible lagrangian framework for modeling the fluid–structure interaction in the underwater implosion of an aluminum cylinder. Math Models Methods Appl Sci 23:339–367
26. Hsu M-C, Akkerman I, Bazilevs Y (2013) Finite element simulation of wind turbine aerodynamics: validation study using NREL phase VI experiment. Wind Energy. doi:10.1002/we.1599
27. Tezduyar T, Aliabadi S, Behr M, Johnson A, Mittal S (1993) Parallel finite-element computation of 3d flows. Computer 26:27–36. doi:10.1109/2.237441
28. Tezduyar T, Aliabadi S, Behr M, Johnson A, Kalro V, Litke M (1996) Flow simulation and high performance computing. Comput Mech 18:397–412. doi:10.1007/BF00350249
29. Tezduyar TE (2001) Finite element methods for flow problems with moving boundaries and interfaces. Arch Comput Methods Eng 8:83–130. doi:10.1007/BF02897870
30. Akin JE, Tezduyar TE, Ungor M (2007) Computation of flow problems with the mixed interface-tracking/interface-capturing technique (MITICT). Comput Fluids 36:2–11. doi:10.1016/j.compfluid.2005.07.008
31. Mittal S, Tezduyar TE (1995) Parallel finite element simulation of 3D incompressible flows—fluid–structure interactions. Int J Numer Methods Fluids 21:933–953. doi:10.1002/fld.1650211011
32. Takizawa K, Henicke B, Puntel A, Spielman T, Tezduyar TE (2012) Space–time computational techniques for the aerodynamics of flapping wings. J Appl Mech 79:010903. doi:10.1115/1.4005073
33. Takizawa K, Henicke B, Puntel A, Kostov N, Tezduyar TE (2012) Space–time techniques for computational aerodynamics modeling of flapping wings of an actual locust. Comput Mech 50:743–760. doi:10.1007/s00466-012-0759-x
34. Takizawa K, Kostov N, Puntel A, Henicke B, Tezduyar TE (2012) Space–time computational analysis of bio-inspired flapping-wing aerodynamics of a micro aerial vehicle. Comput Mech 50:761–778. doi:10.1007/s00466-012-0758-y

35. Takizawa K, Henicke B, Tezduyar TE, Hsu M-C, Bazilevs Y (2011) Stabilized space–time computation of wind-turbine rotor aerodynamics. Comput Mech 48:333–344. doi:10.1007/s00466-011-0589-2
36. Takizawa K, Henicke B, Montes D, Tezduyar TE, Hsu M-C, Bazilevs Y (2011) Numerical-performance studies for the stabilized space–time computation of wind-turbine rotor aerodynamics. Comput Mech 48:647–657. doi:10.1007/s00466-011-0614-5
37. Takizawa K, Tezduyar TE, McIntyre S, Kostov N, Kolesar R, Habluetzel C (2014) Space–time VMS computation of wind-turbine rotor and tower aerodynamics. Comput Mech 53:1–15. doi:10.1007/s00466-013-0888-x
38. Takase S, Kashiyama K, Tanaka S, Tezduyar TE (2011) Space–time supg finite element computation of shallow-water flows with moving shorelines. Comput Mech 48:293–306. doi:10.1007/s00466-011-0618-1
39. Kalro V, Tezduyar TE (2000) A parallel 3d computational method for fluid–structure interactions in parachute systems. Comput Methods Appl Mech Eng 190:321–332. doi:10.1016/S0045-7825(00)00204-8
40. Tezduyar TE, Sathe S, Keedy R, Stein K (2006) Space–time finite element techniques for computation of fluid–structure interactions. Comput Methods Appl Mech Eng 195:2002–2027. doi:10.1016/j.cma.2004.09.014
41. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2006) Computer modeling of cardiovascular fluid–structure interactions with the Deforming-Spatial-Domain/Stabilized Space–Time formulation. Comput Methods Appl Mech Eng 195:1885–1895. doi:10.1016/j.cma.2005.05.050
42. Tezduyar TE, Sathe S, Cragin T, Nanna B, Conklin BS, Pausewang J, Schwaab M (2007) Modeling of fluid–structure interactions with the space–time finite elements: arterial fluid mechanics. Int J Numer Methods Fluids 54:901–922. doi:10.1002/fld.1443
43. Tezduyar TE, Sathe S, Pausewang J, Schwaab M, Christopher J, Crabtree J (2008) Interface projection techniques for fluid–structure interaction modeling with moving-mesh methods. Comput Mech 43:39–49. doi:10.1007/s00466-008-0261-7
44. Tezduyar TE, Sathe S, Schwaab M, Pausewang J, Christopher J, Crabtree J (2008) Fluid–structure interaction modeling of ringsail parachutes. Comput Mech 43:133–142. doi:10.1007/s00466-008-0260-8
45. Takizawa K, Christopher J, Tezduyar TE, Sathe S (2010) Space–time finite element computation of arterial fluid–structure interactions with patient-specific data. Int J Numer Methods Biomed Eng 26:101–116. doi:10.1002/cnm.1241
46. Takizawa K, Moorman C, Wright S, Christopher J, Tezduyar TE (2010) Wall shear stress calculations in space–time finite element computation of arterial fluid–structure interactions. Comput Mech 46:31–41. doi:10.1007/s00466-009-0425-0
47. Takizawa K, Moorman C, Wright S, Spielman T, Tezduyar TE (2011) Fluid–structure interaction modeling and performance analysis of the orion spacecraft parachutes. Int J Numer Methods Fluids 65:271–285. doi:10.1002/fld.2348
48. Takizawa K, Wright S, Moorman C, Tezduyar TE (2011) Fluid-structure interaction modeling of parachute clusters. Int J Numer Methods Fluids 65:286–307. doi:10.1002/fld.2359
49. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2011) Influencing factors in image-based fluid–structure interaction computation of cerebral aneurysms. Int J Numer Methods Fluids 65:324–340. doi:10.1002/fld.2448
50. Tezduyar TE, Takizawa K, Brummer T, Chen PR (2011) Space–time fluid–structure interaction modeling of patient-specific cerebral aneurysms. Int J Numer Methods Biomed Eng 27:1665–1710. doi:10.1002/cnm.1433
51. Takizawa K, Spielman T, Tezduyar TE (2011) Space–time fsi modeling and dynamical analysis of spacecraft parachutes and parachute clusters. Comput Mech 48:345–364. doi:10.1007/s00466-011-0590-9
52. Manguoglu M, Takizawa K, Sameh AH, Tezduyar TE (2011) A parallel sparse algorithm targeting arterial fluid mechanics computations. Comput Mech 48:377–384. doi:10.1007/s00466-011-0619-0

53. Takizawa K, Tezduyar TE (2012) Computational methods for parachute fluid–structure interactions. Arch Comput Methods Eng 19:125–169. doi:10.1007/s11831-012-9070-4
54. Takizawa K, Bazilevs Y, Tezduyar TE (2012) Space–time and ALE-VMS techniques for patient-specific cardiovascular fluid–structure interaction modeling. Arch Comput Methods Eng 19:171–225. doi:10.1007/s11831-012-9071-3
55. Takizawa K, Fritze M, Montes D, Spielman T, Tezduyar TE (2012) Fluid–structure interaction modeling of ringsail parachutes with disreefing and modified geometric porosity. Comput Mech 50:835–854. doi:10.1007/s00466-012-0761-3
56. Takizawa K, Montes D, Fritze M, McIntyre S, Boben J, Tezduyar TE (2013) Methods for FSI modeling of spacecraft parachute dynamics and cover separation. Math Models Methods Appl Sci 23:307–338. doi:10.1142/S0218202513400058
57. Takizawa K, Tezduyar TE, Boben J, Kostov N, Boswell C, Buscher A (2013) Fluid–structure interaction modeling of clusters of spacecraft parachutes with modified geometric porosity. Comput Mech 52:1351–1364. doi:10.1007/s00466-013-0880-5
58. Takizawa K, Tezduyar TE, Buscher A, Asada S (2013) Space–time interface-tracking with topology change (ST-TC). Comput Mech. doi:10.1007/s00466-013-0935-7
59. Brooks AN, Hughes TJR (1982) Streamline upwind/petrov-galerkin formulations for convection dominated flows with particular emphasis on the incompressible navier-stokes equations. Comput Methods Appl Mech Eng 32:199–259
60. Hughes TJR (1995) Multiscale phenomena: green's functions, the dirichlet-to-neumann formulation, subgrid scale models, bubbles, and the origins of stabilized methods. Comput Methods Appl Mech Eng 127:387–401
61. Hughes TJR, Oberai AA, Mazzei L (2001) Large eddy simulation of turbulent channel flows by the variational multiscale method. Phys Fluids 13:1784–1799
62. Bazilevs Y, Calo VM, Cottrell JA, Hughes TJR, Reali A, Scovazzi G (2007) Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. Comput Methods Appl Mech Eng 197:173–201
63. Bazilevs Y, Akkerman I (2010) Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual-based variational multiscale method. J Comput Phys 229:3402–3414
64. Bazilevs Y, Calo VM, Zhang Y, Hughes TJR (2006) Isogeometric fluid–structure interaction analysis with applications to arterial blood flow. Comput Mech 38:310–322
65. Akkerman I, Bazilevs Y, Calo VM, Hughes TJR, Hulshoff S (2008) The role of continuity in residual-based variational multiscale modeling of turbulence. Comput Mech 41:371–378
66. Bazilevs Y, Gohean JR, Hughes TJR, Moser RD, Zhang Y (2009) Patient-specific isogeometric fluid-structure interaction analysis of thoracic aortic blood flow due to implantation of the Jarvik 2000 left ventricular assist device. Comput Methods Appl Mech Eng 198:3534–3550
67. Bazilevs Y, Hsu M-C, Kiendl J, Benson DJ (2012) A computational procedure for pre-bending of wind turbine blades. Int J Numer Methods Eng 89:323–336
68. Korobenko A, Hsu M-C, Akkerman I, Bazilevs Y (2013) Aerodynamic simulation of vertical-axis wind turbines. J Appl Mech. doi:10.1115/1.4024415
69. Bazilevs Y, Hughes TJR (2007) Weak imposition of dirichlet boundary conditions in fluid mechanics. Comput Fluids 36:12–26
70. Bazilevs Y, Michler C, Calo VM, Hughes TJR (2010) Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes. Comput Methods Appl Mech Eng 199:780–790
71. Hsu M-C, Akkerman I, Bazilevs Y (2012) Wind turbine aerodynamics using ALE-VMS: validation and role of weakly enforced boundary conditions. Comput Mech 50:499–511
72. Takizawa K, Yabe T, Tsugawa Y, Tezduyar TE, Mizoe H (2007) Computation of free-surface flows and fluid–object interactions with the cip method based on adaptive meshless soroban grids. Comput Mech 40:167–183. doi:10.1007/s00466-006-0093-2
73. Takizawa K, Tanizawa K, Yabe T, Tezduyar TE (2007) Ship hydrodynamics computations with the CIP method based on adaptive soroban grids. Int J Numer Methods Fluids 54:1011–1019. doi:10.1002/fld.1466

74. Cruchaga MA, Celentano DJ, Tezduyar TE (2007) A numerical model based on the Mixed Interface-Tracking/Interface-Capturing Technique (MITICT) for flows with fluid–solid and fluid–fluid interfaces. Int J Numer Methods Fluids 54:1021–1030. doi:10.1002/fld.1498
75. Cruchaga M, Celentano D, Tezduyar T (2001) A moving lagrangian interface technique for flow computations over fixed meshes. Comput Methods Appl Mech Eng 191:525–543. doi:10.1016/S0045-7825(01)00300-0
76. Sethian J (1999) Level set methods and fast marching methods. Cambridge University Press, Cambridge
77. Kees CE, Akkerman I, Farthing MW, Bazilevs Y (2011) A conservative level set method suitable for variable-order approximations and unstructured meshes. J Comput Phys 230:4536–4558
78. Akkerman I, Bazilevs Y, Benson DJ, Farthing MW, Kees CE (2012) Free-surface flow and fluid–object interaction modeling with emphasis on ship hydrodynamics. J Appl Mech 79:010905
79. Tezduyar TE, Behr M, Mittal S, Johnson AA (1992) Computation of unsteady incompressible flows with the finite element methods—space–time formulations, iterative strategies and massively parallel implementations. In: Smolinski P, Liu WK, Hulbert G, Tamma K (eds) New methods in transient analysis, PVP-Vol. 246/AMD-Vol. 143. ASME, New York, pp 7–24
80. Johnson AA, Tezduyar TE (1994) Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. Comput Methods Appl Mech Eng 119:73–94. doi:10.1016/0045-7825(94)00077-8
81. Hughes TJR, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: cad, finite elements, nurbs, exact geometry, and mesh refinement. Comput Methods Appl Mech Eng 194:4135–4195
82. Bazilevs Y, Hughes TJR (2008) Nurbs-based isogeometric analysis for the computation of flows about rotating components. Comput Mech 43:143–150
83. Takizawa K, Henicke B, Puntel A, Kostov N, Tezduyar TE (2013) Computer modeling techniques for flapping-wing aerodynamics of a locust. Comput Fluids 85:125–134. doi:10.1016/j.compfluid.2012.11.008
84. Takizawa K, Tezduyar TE (2014) Space–time computation techniques with continuous representation in time (ST-C). Comput Mech 53:91–99. doi:10.1007/s00466-013-0895-y
85. Cruchaga M, Celentano D, Tezduyar T (2002) Computation of mould filling processes with a moving Lagrangian interface technique. Commun Numer Methods Eng 18:483–493. doi:10.1002/cnm.506
86. Cruchaga MA, Celentano DJ, Tezduyar TE (2005) Moving-interface computations with the edge-tracked interface locator technique (ETILT). Int J Numer Methods Fluids 47:451–469. doi:10.1002/fld.825
87. Cruchaga MA, Celentano DJ, Tezduyar TE (2007) Collapse of a liquid column: numerical simulation and experimental validation. Comput Mech 39:453–476. doi:10.1007/s00466-006-0043-z
88. Kleefsman KMT, Fekken G, Veldman AEP, Iwanowski B, Buchner B (2005) A volume-of-fluid based simulation method for wave impact problems. J Comput Phys 206:363–393
89. Fridsma G (1968) A systematic study of the rough-water performance of planing boats. Davidson Laboratory Report 1275
90. Longo J, Stern F (2005) Uncertainty assessment for towing tank tests with example for surface combatant dtmb model 5415. J Ship Res 49:55–68
91. Garcia J, Oñate E (2003) An unstructured finite element solver for ship hydrodynamics problems. J Appl Mech 70:18–26
92. Longo J, Shao J, Irvine M, Stern F (2007) Phase-averaged PIV for the nominal wake of a surface ship in regular head waves. J Fluids Eng 129:524–541
93. McCormick ME (2010) Ocean engineering mechanics with applications. Cambridge University Press, Cambridge
94. Hansen SO et al (2006) The Hardanger bridge: static and dynamic wind tunnel tests with a section model. Technical report, Prepared for Norwegian Public Roads Administration

Computational Wind-Turbine Analysis with the ALE-VMS and ST-VMS Methods

Yuri Bazilevs, Kenji Takizawa, Tayfun E. Tezduyar, Ming-Chen Hsu, Nikolay Kostov and Spenser McIntyre

Abstract We provide an overview of the aerodynamic and FSI analysis of wind turbines the first three authors' teams carried out in recent years with the ALE-VMS and ST-VMS methods. The ALE-VMS method is the variational multiscale version of the Arbitrary Lagrangian–Eulerian (ALE) method. The VMS components are from the residual-based VMS (RBVMS) method. The ST-VMS method is the VMS version of the Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) method. The techniques complementing these core methods include weak enforcement of the essential boundary conditions, NURBS-based isogeometric analysis, using NURBS basis functions in temporal representation of the rotor motion, mesh motion and also in remeshing, rotation representation with constant angular velocity, Kirchhoff–Love shell modeling of the rotor-blade structure, and full FSI coupling. The analysis cases include the aerodynamics of wind-turbine rotor and tower and the FSI that accounts for the deformation of the rotor blades. The specific wind turbines considered are NREL 5MW, NREL Phase VI and Micon 65/13M, all at full scale, and our analysis for NREL Phase VI and Micon 65/13M includes comparison with the experimental data.

Y. Bazilevs (✉)
Structural Engineering, University of California, San Diego, 9500 Gilman Drive,
La Jolla, CA 92093, USA
e-mail: jbazilevs@ucsd.edu

K. Takizawa
Department of Modern Mechanical Engineering and Waseda Institute for Advanced Study, Waseda University, 1-6-1 Nishi-Waseda, Shinjuku-ku, Tokyo 169-8050, Japan

T. E. Tezduyar · N. Kostov · S. McIntyre
Mechanical Engineering, Rice University, 6100 Main Street, Houston, TX 77005, USA

M.-C. Hsu
Department of Mechanical Engineering, Iowa State University, 2025 Black Engineering,
Ames, IA 50011, USA

1 Introduction

Countries around the world are putting substantial effort into the development of wind energy technologies. The ambitious wind energy goals put pressure on the wind energy industry research and development to significantly enhance the current wind generation capabilities in a short period of time and decrease the associated costs. This calls for transformative concepts and designs (e.g., floating offshore wind turbines) that must be created and analyzed with high-precision methods and tools. These include complex-geometry, 3D, time-dependent, multi-physics predictive simulation methods and software that will play an increasingly important role as the demand for wind energy grows.

Currently most wind-turbine aerodynamics and aeroelasticity simulations are performed using low-fidelity methods, such as the Blade Element Momentum (BEM) theory for the rotor aerodynamics employed in conjunction with simplified structural models of the wind-turbine blades and tower (see, e.g., [1, 2]). These methods are very fast to implement and execute. However, the cases involving unsteady flow, turbulence, 3D details of the wind-turbine blade and tower geometry, and other similarly-important features, are beyond their range of applicability.

To obtain high-fidelity results for wind turbines, 3D modeling is essential. However, simulation of wind turbines at full scale engenders a number of challenges: the flow is fully turbulent, requiring highly accurate methods and increased grid resolution. The presence of fluid boundary layers, where turbulence is created, complicates the situation further. Wind-turbine blades are long and slender structures, with complex distribution of material properties, for which the numerical approach must have good approximation properties and avoid locking. Wind-turbine simulations involve moving and stationary components, and the fluid–structure coupling must be accurate, efficient and robust to preclude divergence of the computations. These explain the current, modest nature of the state-of-the-art in wind-turbine simulations.

Fluid–structure interaction (FSI) simulations at full scale are essential for accurate modeling of wind turbines. The motion and deformation of the wind-turbine blades depend on the wind speed and air flow, and the air flow patterns depend on the motion and deformation of the blades. In order to simulate the coupled problem, the equations governing the air flow and the blade motions and deformations need to be solved simultaneously, with proper kinematic and dynamic conditions coupling the two physical systems. Without that the modeling cannot be realistic: unsteady blade deformation affects aerodynamic efficiency and noise generation, and response to wind gusts. Flutter analysis of large blades operating in offshore environments is of great importance and cannot be accomplished without FSI.

In recent years, several attempts were made to address the above mentioned challenges and to raise the fidelity and predictability levels of wind-turbine simulations. Standalone aerodynamics simulations of wind-turbine configurations in 3D were reported in [3–6], while standalone structural analyses of rotor blades of complex geometry and material composition, but under assumed wind-load conditions or wind-load conditions coming from separate aerodynamic computations were reported in [7–11]. In a recent work [12] it was shown that coupled FSI modeling and simulation of wind turbines is important for accurately predicting their mechanical behavior at full scale.

To address the above mentioned challenges one should employ a combination of numerical techniques, which are general, accurate, robust and efficient for the targeted class of problems. Such techniques are summarized in what follows, with some of them described in greater detail in the body of this book chapter.

Isogeometric Analysis (IGA), first introduced in [13] and further expanded on in [14–20], is adopted as the geometry modeling and simulation framework for wind turbines in some of the examples presented here. We use the IGA based on NURBS (non-uniform rational B-splines), which are more efficient than standard finite elements for representing complex, smooth geometries, such as wind-turbine blades. The IGA was successfully employed for computation of turbulent flows [21–26], nonlinear structures [10, 27–31], and FSI [32–35], and, in most cases, gave a clear advantage over standard low-order finite elements in terms of solution accuracy per-degree-of-freedom. This is in part attributable to the higher-order smoothness of the basis functions employed. Flows about rotating components are naturally handled in an isogeometric framework because all conic sections, and in particular, circular and cylindrical shapes, are represented exactly [36].

The blade structure is governed by the isogeometric rotation-free shell formulation with the aid of the bending-strip method [10]. The method is appropriate for thin-shell structures comprised of multiple C^1- or higher-order continuous surface patches that are joined or merged with continuity no greater than C^0. The Kirchhoff—Love shell theory that relies on higher-order continuity of the basis functions is employed in the patch interior as in [31]. Although NURBS-based IGA is employed in this work, other discretizations such as T-splines [19, 20] or subdivision surfaces [37–39], are perfectly suited for the proposed structural modeling method.

In addition, an isogeometric representation of the analysis-suitable geometry can be used in generating tetrahedral and hexahedral meshes for computations with the finite element method (FEM). In this article, we use tetrahedral meshes generated that way in wind-turbine computations with the ALE-VMS and ST-VMS methods. The ALE-VMS method [5, 34] is the variational multiscale (VMS) version of the Arbitrary Lagrangian–Eulerian (ALE) method [40]. The VMS components are from the residual-based VMS (RBVMS) method given in [21, 26, 41, 42]. The ST-VMS method [43, 44] is the VMS version of the Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) method [45–49]. Earlier it was called "DSD/SST-VMST" (i.e. the version with the VMS turbulence model) in [43]. The original DSD/SST formulation was named "DSD/SST-SUPS" in [43] (i.e. the version with the SUPG/PSPG stabilization), which was also called "ST-SUPS" in [50].

The ALE-VMS method originated from the RBVMS formulation of incompressible turbulent flows proposed in [21] for stationary meshes, and may be thought of as an extension of the RBVMS method to moving meshes. As such, it was presented for the first time in [34] in the context of FSI. Although ALE-VMS gave reasonably good results for several important turbulent flows, it was evident in [21, 24] that to obtain accurate results for wall-bounded turbulent flows the method required relatively fine resolution of the boundary layers. This fact makes ALE-VMS a somewhat costly technology for full-scale wall-bounded turbulent flows at high Reynolds numbers, which are characteristic of the present application. For this reason, weakly-enforced essential boundary condition formulation was introduced in [51], which significantly improved the performance of the ALE-VMS formulation in the presence of unresolved boundary layers [22, 23, 26]. The weak boundary condition formulation may be thought of as an extension of Nitsche's method [52] to the Navier–Stokes equations of incompressible flows. Another interpretation of the weak boundary condition formulation is that it is a discontinuous Galerkin method (see, e.g., [53]), where the continuity of the basis functions is enforced everywhere in the domain interior, but not at the domain boundary.

The DSD/SST formulation was introduced in [45–47] as a general-purpose interface-tracking (moving-mesh) technique for flows with moving boundaries and interfaces, including FSI and flows with moving objects. Its stabilization components are the Streamline-Upwind/Petrov-Galerkin (SUPG) [54] and Pressure-Stabilizing/Petrov-Galerkin (PSPG) [45, 55] stabilizations. It also includes the "LSIC" (least-squares on incompressibility constraint) stabilization. Some of the earliest FSI computations with the DSD/SST formulation were reported in [56] for vortex-induced vibrations of a cylinder and in [57] for flow-induced vibrations of a flexible, cantilevered pipe (1D structure with 3D flow). The DSD/SST formulation has been used extensively in 3D computations of parachute FSI, starting with the 3D computations reported in [58] and evolving to computations with direct coupling [59]. New versions of the DSD/SST formulation introduced in [49] are the core technologies of the Stabilized ST FSI (SSTFSI) technique, which was also introduced in [49]. The ST-VMS method and SSTFSI technique, combined with a number of special techniques (see [60–63] and references therein) have been used in some of the most challenging parachute FSI computations (see [60, 64–66] and references therein), and also in a good number of patient-specific cardiovascular FSI and fluid mechanics computations (see [61–63, 67] and references therein). Computations with the SSTFSI technique also received a substantial attention in research related to iterative solution of large linear systems [68, 69].

In application of the DSD/SST formulation to flows with moving objects, the Shear–Slip Mesh Update Method (SSMUM) [70–72] has been very instrumental. The SSMUM was first introduced for computation of flow around two high-speed trains passing each other in a tunnel (see [70]). The challenge was to accurately and efficiently update the meshes used in computations based on the DSD/SST formulation and involving two objects in fast, linear relative motion. The idea behind the SSMUM was to restrict the mesh moving and remeshing to a thin layer of elements between the objects in relative motion. The mesh update at each time step can be accomplished by a "shear" deformation of the elements in this layer, followed by a "slip" in node connectivities. The slip in the node connectivities, to an extent, un-does the deformation of the elements and results in elements with better shapes than those that were shear-deformed. Because the remeshing consists of simply re-defining the node connectivities, both the projection errors and the mesh generation cost are minimized. A few years after the high-speed train computations, the SSMUM was implemented for objects in fast, rotational relative motion and applied to computation of flow past a rotating propeller [71] and flow around a helicopter [72].

The ST-VMS method was successfully tested on computation of wind-turbine rotor aerodynamics in [6, 73, 74]. Those computations did not include a wind-turbine tower, and therefore a mesh update method was not required. In [75], the ST-VMS method was applied to computation of wind-turbine rotor and tower aerodynamics. The presence of a tower requires a mesh update method that can handle the fast, rotational relative motion between the rotor and tower. The SSMUM would have been an option, but we decided to use a method that is more general. We use NURBS basis functions for the temporal representation of the rotor motion, mesh motion and also in remeshing. This is essentially the same computational technology used in the ST-VMS computations of flapping-wing aerodynamics reported in [76–79]. We named it "ST/NURBS Mesh Update Method (STNMUM)" in [75]. The motion of the rotor surface mesh created from the NURBS geometry is represented by quadratic temporal NURBS basis functions, with sufficient number of temporal patches for one rotation. This enables us to represent the circular paths associated with the rotor motion exactly and, with a "secondary mapping" [43, 44, 50, 76], specify a constant angular velocity corresponding to the invariant speeds along those paths. Given the motion of the surface mesh, we compute meshes that serve as temporal-control points. This is done by creating with an automatic mesh generator a new mesh at the central control point of the temporal patch, and computing the meshes at the other two control points by using the mesh moving technique [49, 80–83] developed earlier in conjunction with the DSD/SST method. The STNMUM allows us to do mesh computations with longer time in between, but get the mesh-related information for each ST slab, such as the coordinates and their time derivatives, from the temporal representation whenever we need. This approach where the mesh-related information is computed "directly" was called in [75] "Direct Temporal Representation (DTR)." In an alternative approach, we can obtain the mesh-related data after first computing the finite element meshes associated with each ST slab by interpolation from the temporal NURBS representation of the mesh. This approach was called "Interpolated-Mesh Temporal Representation (IMTR)" in [75]. For better mesh resolution, we use layers of thin elements near the blade surfaces. These layers of elements are created with a special mesh generation process and are not part of what we create with the automatic mesh generation process. They undergo rigid-body motion with the rotor. Despite the fast, rotational relative motion between the rotor and tower, the computations reported in [75] were carried out by using an automatic mesh generator only a total of 6 times during an entire computation.

We refer the interested reader to [50, 74] for the following methods that are not reviewed in this article: ALE-VMS and ST-VMS methods, formulation for weakly-enforced essential boundary conditions, structural mechanics formulation, which is based on the Kirchhoff–Love thin-shell theory and the bending-strip method (see [10, 12, 31]), FSI coupling, mesh update, and a method for pre-bending of wind-turbine blades, which was recently proposed in [11].

In Sect. 2, we present the sliding-interface formulation from [36, 84, 85], which enables the simulation of rotor–tower interaction. The formulation was used in [85]

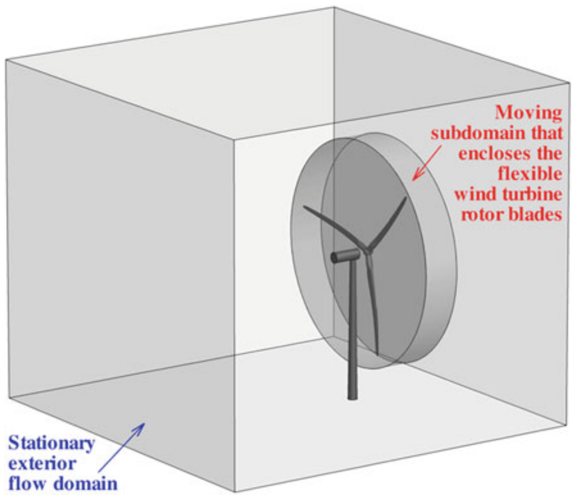

Moving subdomain that encloses the flexible wind turbine rotor blades
Stationary exterior flow domain

Fig. 1 Setup for the simulation of a full machine. The interior moving subdomain, which encloses the wind turbine rotor, and the exterior stationary subdomain, which houses the nacelle and tower

for ALE-VMS aerodynamic simulations of the National Renewable Energy Lab (NREL) Phase VI wind turbine (see [86]) for comparison to the extensive set of experimental data available for this test case. We also present those simulations in Sect. 2. In Sect. 3, we describe, from [75], the ST-VMS computations of the wind-turbine rotor and tower aerodynamics. NURBS basis functions are used in temporal representation of the rotor and volume mesh motion and in remeshing. Simulations of the Micon 65/13M wind turbine with FSI, reported earlier in [87], are described in Sect. 4. We end with concluding remarks in Sect. 5.

2 Sliding-Interface Formulation and Rotor–Tower Interaction

2.1 Sliding-Interface Formulation

In order to simulate the full wind turbine configuration and investigate the rotor–tower interaction, we consider an approach that makes use of a moving subdomain, which encloses the entire wind turbine rotor, and a stationary subdomain that contains the rest of the wind turbine (see Fig. 1). The two domains are in relative motion and share a sliding cylindrical interface. The meshes on each side of the interface are nonmatching because of the relative motion (see Fig. 2). As a result, a numerical procedure is needed to impose the continuity of the kinematics and tractions at the stationary and rotating subdomain interface despite the fact that the interface discretizations are incompatible. Such a procedure was developed in [36] in the context of IGA for computing flows about rotating components. The advantage of IGA for

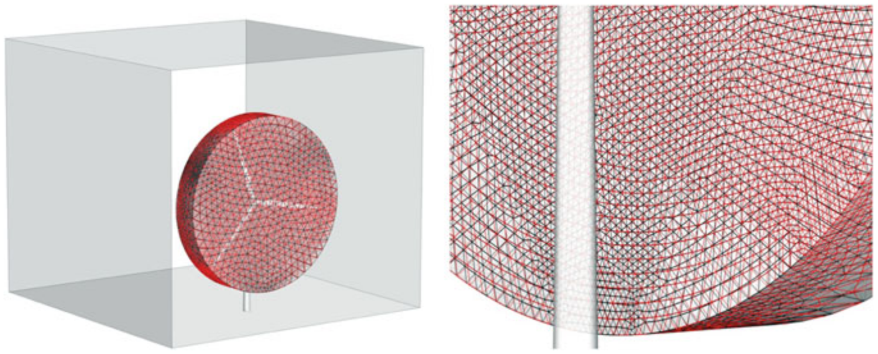


Fig. 2 Nonmatching meshes at the sliding interface between the stationary and moving subdomains. *Left* Full domain. *Right* Zoom on the sliding interface

rotating-component flows is that the cylindrical sliding interfaces are represented exactly and no geometry errors are incurred. In the case of standard FEM employed here, the geometric compatibility is only approximate. The sliding-interface coupling was successfully tested on the NREL Phase VI wind turbine in [85] and is presented in what follows.

Let the subscripts S and M denote the quantities pertaining to the fluid mechanics problem on the stationary and moving subdomains, respectively. The subdomain that encloses the rotor rotates with it, and the interior of the rotating subdomain is allowed to deflect to accommodate the motion of the blades. However, the motion of the outer boundary of the rotor subdomain is restricted to a rigid rotation to maintain geometric compatibility with the stationary subdomain. To enforce the compatibility of the flow kinematics and tractions at the sliding interface, we add the following terms to the ALE-VMS formulation, which is now assumed to hold in both the stationary and moving subdomains:

$$
\begin{aligned}
&-\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \bigcap (\Gamma_t)_{\mathrm{SI}}} \left(\mathbf{w}_{\mathrm{S}}^h - \mathbf{w}_{\mathrm{M}}^h\right) \cdot \frac{1}{2} \left(\boldsymbol{\sigma}_{\mathrm{S}} \mathbf{n}_{\mathrm{S}} - \boldsymbol{\sigma}_{\mathrm{M}} \mathbf{n}_{\mathrm{M}}\right) \mathrm{d}\Gamma \\
&-\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \bigcap (\Gamma_t)_{\mathrm{SI}}} \frac{1}{2} \left(\delta\boldsymbol{\sigma}_{\mathrm{S}} \mathbf{n}_{\mathrm{S}} - \delta\boldsymbol{\sigma}_{\mathrm{M}} \mathbf{n}_{\mathrm{M}}\right) \cdot \left(\mathbf{u}_{\mathrm{S}}^h - \mathbf{u}_{\mathrm{M}}^h\right) \mathrm{d}\Gamma \\
&-\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \bigcap (\Gamma_t)_{\mathrm{SI}}} \mathbf{w}_{\mathrm{S}}^h \cdot \rho \left\{\left(\mathbf{u}_{\mathrm{S}}^h - \hat{\mathbf{u}}_{\mathrm{S}}^h\right) \cdot \mathbf{n}_{\mathrm{S}}\right\}_{-} \left(\mathbf{u}_{\mathrm{S}}^h - \mathbf{u}_{\mathrm{M}}^h\right) \mathrm{d}\Gamma \\
&-\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \bigcap (\Gamma_t)_{\mathrm{SI}}} \mathbf{w}_{\mathrm{M}}^h \cdot \rho \left\{\left(\mathbf{u}_{\mathrm{M}}^h - \hat{\mathbf{u}}_{\mathrm{M}}^h\right) \cdot \mathbf{n}_{\mathrm{M}}\right\}_{-} \left(\mathbf{u}_{\mathrm{M}}^h - \mathbf{u}_{\mathrm{S}}^h\right) \mathrm{d}\Gamma
\end{aligned}
$$

$$+\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \bigcap (\Gamma_t)_{\mathrm{SI}}} \frac{C_I^B \mu}{h_n} \left(\mathbf{w}_{\mathrm{S}}^h - \mathbf{w}_{\mathrm{M}}^h\right) \cdot \left(\mathbf{u}_{\mathrm{S}}^h - \mathbf{u}_{\mathrm{M}}^h\right) \mathrm{d}\Gamma = 0, \tag{1}$$

where $\delta\boldsymbol{\sigma}$ is given by $\delta\boldsymbol{\sigma}\left(\mathbf{w}, q\right)\mathbf{n} = 2\mu\boldsymbol{\varepsilon}(\mathbf{w})\mathbf{n} + q\mathbf{n}$. $(\Gamma_t)_{\mathrm{SI}}$ is the sliding interface, and $\{\mathcal{A}\}_-$ denotes the negative part of $\mathcal{A}$, that is, $\{\mathcal{A}\}_- = \mathcal{A}$ if $\mathcal{A} < 0$ and $\{\mathcal{A}\}_- = 0$ if $\mathcal{A} \geq 0$. The sliding-interface formulation may be seen as a DG method, where the continuity of the basis function is enforced everywhere in the interior of the two subdomains, but not at the sliding interface between them. The structure of the terms on the sliding interface is similar to that of the weak enforcement of essential boundary conditions. The significance of each term is explained in detail in [36]. In the current application, $\hat{\mathbf{u}}_{\mathrm{S}}^h = \mathbf{0}$, because the subdomain S is stationary. However, the formulation is able to handle situations where both subdomains are in motion.

Remark 1 Nonmatching interface discretizations in the FSI and sliding-interface problems necessitate the use of interpolation or projection of kinematic and traction data between the nonmatching surface meshes (see, e.g., [43, 44, 88], where [44] is more comprehensive than [43]). A computational procedure, which can simultaneously handle the data transfer for IGA and FEM discretizations, was proposed in [88]. The procedure also includes a robust approach in identifying “closest points” for arbitrary shaped surfaces. While such interface projections are rather straightforward for weakly-coupled FSI algorithms, they require special techniques [44, 49, 60] for strongly-coupled, direct and quasi-direct methods [44, 49, 59, 60, 89], which become monolithic for matching discretizations.

2.2 NREL Phase VI Wind Turbine

The computational results in this section make use of the ALE-VMS technique and are taken from [85]. The sliding-interface formulation is applied to the simulation of the full NREL Phase VI wind turbine configuration, including the rotor (blades and hub), nacelle and tower. The tower is composed of two cylinders with diameters of 0.6096 and 0.4064 m that are connected with a short conical section. The tower height is 11.144 m above the wind tunnel floor. The detailed geometry of the tower and nacelle can be found in Hand et al. [86]. For this study, wind speeds of 7 and 10 m/s were selected from the experimental sequence S. The experimental sequence S setup consists the wind turbine rotor in the upwind configuration, 0° yaw angle, 0° cone angle, rotational speed of 72 rpm, and blade tip pitch angle of 3°. The air density and viscosity are 1.23 kg/m^3 and 1.78×10^{-5} kg/(m·s), respectively.

Figure 3 shows the mesh resolution used in the computation. The mesh is highly refined near the rotor, nacelle and tower, as well as downstream of the wind turbine to better capture the wake turbulence. The mesh is comprised of 6,835,647 linear elements and 1,603,377 nodes. The size of the first boundary-layer element in the

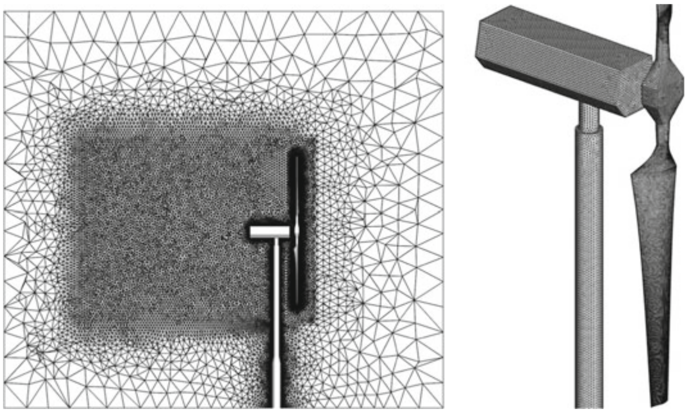


Fig. 3 Meshes used in the full-wind-turbine simulation. *Left* 2D cut at $x = 0$ to show the flow domain mesh quality. *Right* Rotor, nacelle, and tower surface mesh

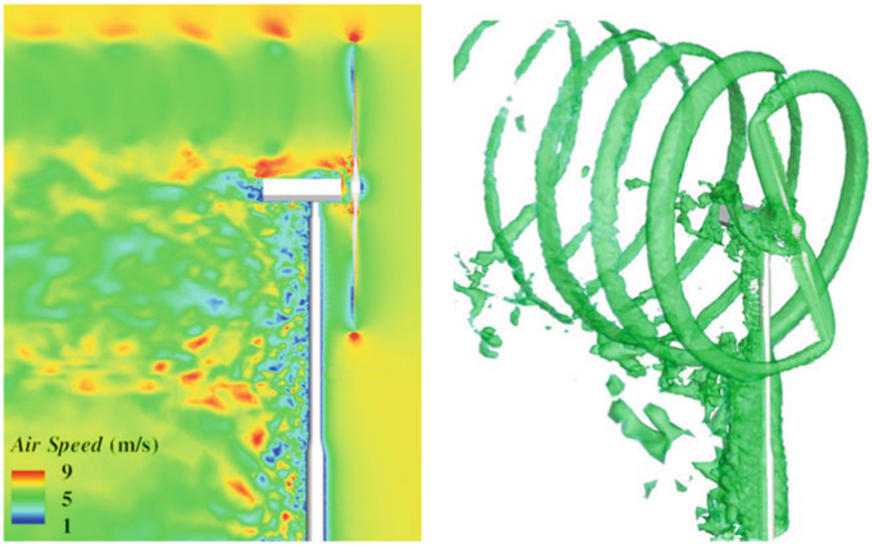


Fig. 4 Air speed planar distribution and isosurfaces at an instant for the 7 m/s case

wall-normal direction is 0.002 m, and 15 layers of prismatic elements were generated with a growth ratio of 1.2. The time step size is set to 1.0×10^{-5} s.

Figures 4 and 5 show the flow visualization of the full-wind-turbine simulations of the 7 and 10 m/s cases, respectively. The flow structures are different between the two cases. The tip vortex for the 7 m/s case decays very slowly as it is convected downstream, while the tip vortex breaks down quickly for the 10 m/s case. No visible discontinuities are present in the flow field at the sliding interface, which indicates that the method correctly handles the kinematic compatibility conditions.

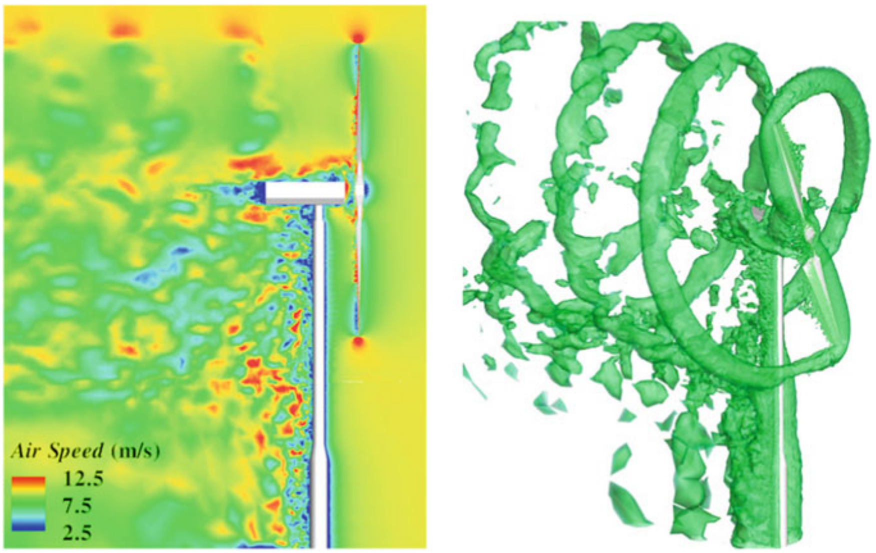


Fig. 5 Air speed planar distribution and isosurfaces at an instant for the 10 m/s case

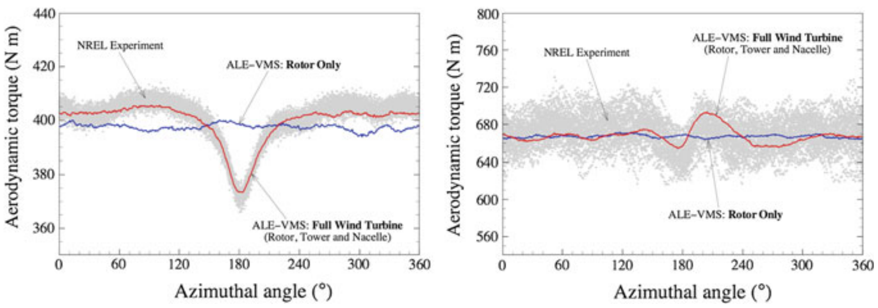


Fig. 6 Single-blade aerodynamic torque over a full revolution for 7 m/s (*left*) and 10 m/s (*right*) cases. The 180° azimuthal angle corresponds to the instant when the blade passes in front the tower. The tower effect is clearly pronounced in the 7 m/s case. It is also present in the 10 m/s case, but is not as significant. The results in both cases are in very good agreement with the experimental data

To see the influence of the tower, the single-blade aerodynamic torque over a full revolution is plotted in Fig. 6 for both 7 and 10 m/s cases. The results of the full-wind-turbine computations are compared with the experimental data, as well as with the results of the rotor-only computations. For the full-wind-turbine simulation of the 7 m/s case, Fig. 6 clearly shows the drop in the aerodynamic torque at an instant when the blade passes in front of the tower, which corresponds to the azimuthal angle of 180°. The drop in the torque is about 8 % relative to its value when the blade is away from the tower. These results are in good agreement with the experimental data. The rotor-only computation, which is also shown in the figure, is obviously unable to predict this feature, which may be important for the transient structural response of the blades. It should be noted, however, that the cycle-averaged aerodynamic torque is nearly identical for the full-wind-turbine and the rotor-only simulations. The picture is completely different for the 10 m/s case, where the influence of the tower, although clearly present, is a lot less pronounced.

3 ST-VMS Computation of the Wind-Turbine Rotor and Tower Aerodynamics

This section is from [75].

3.1 Rotation Representation with Constant Angular Velocity

We use quadratic NURBS functions, as described in [43, 44, 50, 76], to represent a circular arc. We discretize time and position as follows:

$$t = \sum_{\alpha=1}^{n_{\text{ent}}} T^{\alpha}(\Theta_t(\theta))t^{\alpha}, \qquad \mathbf{x} = \sum_{\alpha=1}^{n_{\text{ent}}} T^{\alpha}(\Theta_x(\theta))\mathbf{x}^{\alpha}. \tag{2}$$

Here n_{ent} is the number of temporal element nodes, T^{α} is the basis function, $\Theta_t(\theta)$ and $\Theta_x(\theta)$ are the secondary mappings for time and position, and t^{α} and $\mathbf{x}^{\alpha}$ are the time and position values corresponding to the basis function T^{α}. The basis functions could be finite element or NURBS basis functions. For the circular arc, $n_{\text{ent}} = 3$ and they are quadratic NURBS. The secondary mapping concept above was introduced in [43], and the velocity can be expressed as follows:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \left(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_x}\frac{\mathrm{d}\Theta_x}{\mathrm{d}\theta}\mathbf{x}^{\alpha}\right)\left(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_t}\frac{\mathrm{d}\Theta_t}{\mathrm{d}\theta}t^{\alpha}\right)^{-1}, \tag{3}$$

leading to

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \left(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_x}\mathbf{x}^{\alpha}\right)\left(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_t}t^{\alpha}\right)^{-1}\left(\frac{\mathrm{d}\Theta_x}{\mathrm{d}\theta}\frac{\mathrm{d}\theta}{\mathrm{d}\Theta_t}\right). \tag{4}$$

Thus, the speed along the path can be specified only by modifying the secondary mapping. For a circular arc, two methods were introduced in [44, 76]; one is modifying the secondary mapping for position and the other one is modifying both such that $\frac{\mathrm{d}t}{\mathrm{d}\theta}$ is constant. We note that, in theory, the secondary mapping selections do not make any difference as long as the relationship $\frac{\mathrm{d}\Theta_x}{\mathrm{d}\Theta_t}$ is the same. In our implementation, to keep the process general, we search for the parametric coordinate θ by using an iterative solution method [44, 50, 76]. We use the latter set of the secondary mappings, having constant $\frac{\mathrm{d}t}{\mathrm{d}\theta}$. For the IMTR, we find the parametric coordinate corresponding to each time level and interpolate the position to obtain the corresponding mesh. For the DTR, we first calculate time corresponding to each integration point, including the time step size because of the jump term, and then calculate Θ_x and Θ_t to interpolate the position and velocity from Eqs. (2) and (4).

3.2 Geometry Construction

The geometry construction for the wind-turbine rotor blade and hub was described in [5, 6], and also partially in [73, 75]. For completeness we repeat some of that information here. The geometry of the rotor blade is based on the NREL 5MW offshore baseline wind turbine reported in [90]. A 61 m blade is attached to a hub with radius of 2 m, making the total rotor radius, R, 63 m. The blade is composed of several airfoil types. The first portion of the blade is a perfect cylinder. Farther away from the root the cylinder is smoothly blended into a series of DU (Delft University) airfoils. Starting at 44.55 m from the root and all the way to the tip, the NACA64 profile is used. For each cross-section, we use quadratic NURBS to represent the 2D airfoil shape. The weights of the NURBS functions are set to unity. The weights are adjusted near the root to represent the circular cross-sections exactly. The cross-sections are lofted along the blade axis direction, also using quadratic NURBS and unit weights. This geometry-construction process yields a smooth blade surface with a relatively small number of input parameters, which is an advantage of the isogeometric representation. Images of the airfoil types used in the wind-turbine rotor blade and the final blade including the twisting cross-sections can be found in [5, 6, 73]. Starting from this rotor surface geometry, we generate a quadratic NURBS surface with G^2 and G^1 continuity between the patches around and along the blade, respectively. The tower geometry was created based on the tower design specified for the NREL 5MW wind turbine, which describes a circular tower with a height of 87.6 m, a base diameter of 6 m, and a top diameter of 3.87 m. This geometry was generated by lofting between NURBS curves for the top and base of the tower. The rotor axis is 90° to the tower, and there is no tilt or precone. The distance between the tower axis and the point where the three blade axes intersect is 5 m. For most of the blade, the clearance from the tower is in the range 2.3–2.8 m.

3.3 Problem Setup

We compute the aerodynamics of the rotor with and without its tower for a given rotor shape and wind speed and a specified rotor speed. The wind speed is uniform at 9 m/s and the rotor speed is 1.08 rad/s, giving a tip speed ratio of 7.55 (see [91] for

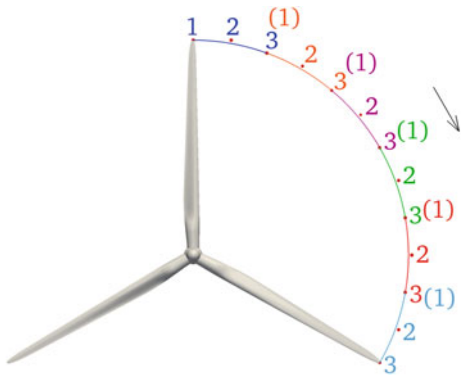


Fig. 7 Path of a blade tip with temporal patches and control point numbering local to each patch. A control point at the start of a patch and colocated with a control point at the end of the previous patch is in parentheses. *Patch colors* 1 *blue*, 2 *orange*, 3 *purple*, 4 *green*, 5 *red*, 6 *teal*

wind-turbine terminology). We use air properties at standard sea-level conditions. The Reynolds number (based on the cord length at $\frac{3}{4}R$ and the relative velocity there) is approximately 12 million. At the inflow boundary the velocity is set to the wind velocity, at the outflow boundary the stress vector is set to zero, and at the top, side, and bottom boundaries slip conditions are imposed.

3.4 Rotor Motion

The circular turbine rotation is represented with temporal NURBS basis functions and secondary mapping, described in Sect. 3.1. Because the three blades of the turbine are 120° apart, rotational geometric periodicity is used such that a full 360° rotation is defined by three identical 120° segments. Each 120° segment is divided into six patches to keep the mesh distortion under control. Each patch has three temporal-control points. The six temporal patches and their control points are shown in Fig. 7.

3.5 Surface Mesh

The rotor surface mesh is generated by discretizing the NURBS surface geometry at each knot intersection, subdividing the knot spans into quadrilateral finite elements in a structured way, and subdividing the quadrilateral elements into two triangles. Small adjustments are made to improve the mesh near the hub. The surface mesh position is calculated at each temporal-control point shown in Fig. 7. Figure 8 shows the rotor surface at the three temporal-control points of the first patch. We note that control points 1 and 3 lie on the path traveled by the points on the blades and a portion of the hub at the start and end of the 20° rotation, but control point 2 lies outside the circular arc. This means that the temporal-control mesh 2 is deformed

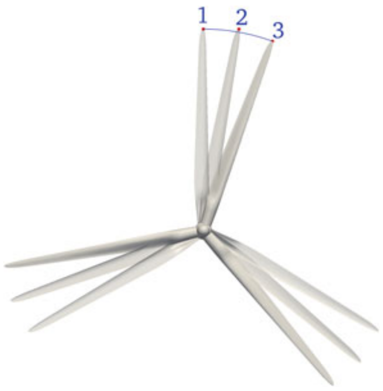


Fig. 8 Rotor surface at the three temporal-control points of the first patch

compared to the temporal-control meshes 1 and 3. A temporal-control mesh 2 has to be generated for the part of the surface between the hub cross-sections rotating with the blades and fixed to the tower. The tower surface mesh is generated from the NURBS representation of the surface by using an unstructured triangular mesh generator and matched with the previously generated hub mesh at the intersection. The rotor surface mesh has 34,087 nodes and 68,112 triangles. The tower surface mesh has 6,952 nodes and 13,806 triangles.

3.6 Volume Mesh

3.6.1 Boundary-Layer Mesh

The layers of thin elements near the blades are generated by extruding the NURBS surface geometry into NURBS volume representation, subdividing the knot spans into hexahedral finite elements in a structured way, and subdividing the hexahedral elements into six tetrahedral elements. The resulting boundary-layer mesh for each blade consists of four layers with a first-layer thickness of about 2.85×10^{-2} m and a total thickness of about 2.85×10^{-1} m, 52 nodes in the circumferential direction around the blade, and approximately 145 nodes in the longitudinal direction. The tower boundary-layer mesh is generated by extruding the tower surface mesh to layers of prismatic elements, which are then subdivided into three tetrahedral elements each. It consists of four layers, with a first-layer thickness of 2.85×10^{-2} m and a total thickness of 3.0×10^{-1} m. The blade and tower boundary-layer meshes do not undergo any mesh deformation. This maintains the mesh quality in the boundary-layer regions. Figure 9 shows the tower and blade boundary-layer meshes.

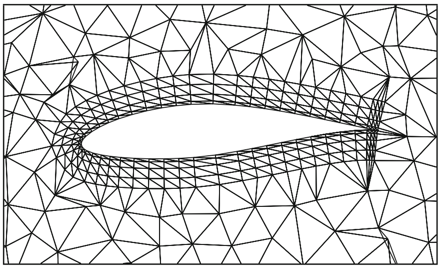
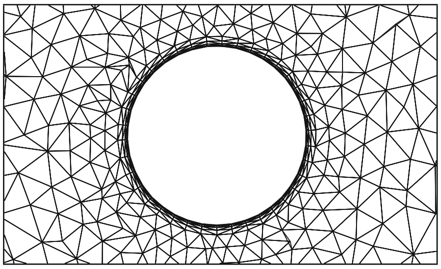



Fig. 9 *Left* Boundary-layer mesh at $\frac{3}{4}R$. *Right* Tower boundary-layer mesh

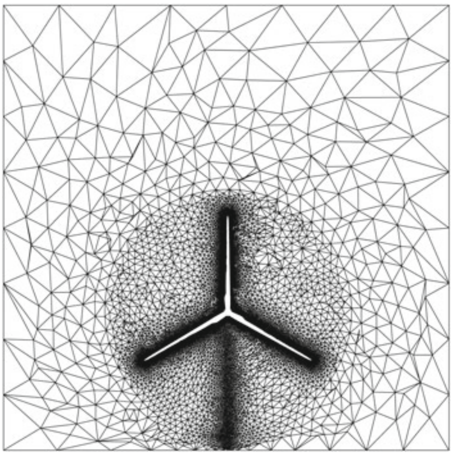


Fig. 10 A cut plane of temporal-control Mesh 1 of patch 1 for Mesh 3

3.6.2 Overall Mesh

Three different meshes are used in the computations: Mesh 1, Mesh 2, and Mesh 3. Mesh 2 has both the rotor and the tower, with boundary-layer mesh only for the blades. Mesh 1 has only the rotor, and is identical to Mesh 2 except the tower is filled with volume elements. Mesh 3 has both the rotor and the tower, with boundary-layer mesh for both the blades and the tower, and a mesh refinement region downstream of the tower. All three meshes have an outer, coarser region, with an inner cylindrical refinement region surrounding the rotor. This inner refinement region includes most of the tower for Mesh 2 and Mesh 3, and the mesh refinement region downstream of the tower for Mesh 3. Figure 10 illustrates, as an example, a cut plane of Mesh 3. The inflow and outflow boundaries are at $3.79R$ and $10.35R$ from the hub center, respectively. The side, top, and bottom boundaries are at $2.29R$, $3.17R$, and $1.43R$, respectively (see Fig. 10). The volume mesh is generated once per patch using an automatic mesh generator (a total of 6 times). The mesh is generated at control point 2 of each patch to minimize mesh distortion between control points. We note that only the mesh in the inner cylindrical refinement region surrounding the rotor is generated for each patch. The outer, coarser mesh is generated only once, and is kept the same when the inner meshes are generated for each patch. The mesh moving technique [49, 80–83] developed earlier in conjunction with the DSD/SST method is used for computing the mesh position for control points 1 and 3. The outer surfaces of the boundary-layer meshes serve as the boundaries where we specify the inner boundary conditions for the mesh motion. The external boundaries of the computational domain serve as the boundaries where we specify the outer boundary conditions, with zero displacement. In the elasticity equations of the mesh moving technique, a Young's modulus of 1.0, a Poisson's ratio of -0.20, and a stiffening exponent of 1.5 are used. We use 1,500 GMRES [92] iterations for each step of the mesh motion, with diagonal preconditioner. Each 10° range of motion is computed over 40 steps. The approximate number of nodes for Mesh 1, Mesh 2 and Mesh 3 are 465,000, 440,000 and 595,000.

3.7 Computational Conditions

In the ST-VMS computations, the stabilization parameters are given by Eq. (7) in [49] for τ_{M} $(=\tau_{\mathrm{SUPS}}) = \tau_{\mathrm{SUPG}}$ and Eq. (19) in [75] for ν_{C} $(=\nu_{\mathrm{LSIC}}) = \nu_{\mathrm{LSIC-HRGN}}$. They are both used with $h_{\mathrm{RGN}} = h_{\mathrm{RGNT}}$, given by Eq. (15) in [75], which was originally introduced in [76]. The DTR and IMTR approaches are used on all three meshes. Least-squares projection is used to interpolate the velocity and pressure between temporal patches. Because the boundary-layer meshes and the tower and rotor surface meshes remain identical between temporal patches, the velocity values are transferred exactly for those nodes. The time-step size is 2.23×10^{-3} s (145 time steps per patch), with four nonlinear iterations per time-step. First we develop the flow field for 500 time steps while the rotor is static, ramping up the inflow velocity during the first 300 steps from zero to the wind speed using a cosine ramp. During this flow-development stage, we use 150, 150, 200, and 400 GMRES iterations for the four nonlinear iterations. In computations with the rotor in motion, we use 150, 150, 200, and 400 GMRES iterations for Mesh 1, and 150, 250, 350, and 500 GMRES iterations for Mesh 2 and Mesh 3. With the GMRES iterations in flow computations, we use nodal-block-diagonal preconditioner. The mesh is partitioned based on the METIS algorithm [93] to improve parallel efficiency in the computations.

3.8 Results

Figure 11 shows the torque for Mesh 1 with the DTR approach, for the last 360° rotation of a blade, with the rotation amount measured from the orientation seen in Fig. 7. For reference purposes, Fig. 11 includes the NREL data. The torque is within 8 % of the NREL data. Figure 12 shows the torque for the last 80° rotation of a single blade of Mesh 1 with the DTR approach, compared with the torque from an earlier, single-blade computation [73] using the TGI option of ν_{C} (=ν_{LSIC}). The single-blade computation has the same blade geometry, wind speed, and rotor speed, but has a single-blade mesh in a rotationally-periodic domain. It has a more refined boundary-layer mesh and a time-step size that is approximately five times smaller. The higher torque seen for the single-blade computation may be due to the fact that the computation was carried out for a much shorter duration, only 80° of rotation versus 1,080° for the Mesh 1 computation. Therefore the current computation likely represents a more settled torque value. The higher torque for the single-blade computation may also be due the fact that the computation was carried out using a computational domain with significantly nearer lateral boundaries. Figures 13 and 14 show the torque for all three meshes with the DTR and IMTR approaches. As can be seen from these figures, Mesh 1 (no tower) has a very stable torque, while Mesh 2 and Mesh 3 (with tower) exhibit a significant but expected drop in torque each time a blade passes the tower. Figure 15 shows, for each of the three meshes, the torque obtained with the DTR and IMTR approaches. The figure illustrates that the DTR and IMTR approaches result in a nearly identical torque magnitude for all three meshes. Figure 16 shows the torque for Mesh 1 with the DTR approach, using two different time-step sizes: 2.23×10^{-3} s (145 time steps per patch) and 4.49×10^{-3} s (72 time steps per patch). Doubling the time-step size still yields a comparable torque value, within 10 % of the value for the smaller time-step size. We also carried out a computation with the convective form of the ST-VMS formulation (see Eq. (8.17) in [44]), but with a smaller time-step size: 4.46×10^{-4} s (725 time steps per patch). Figure 17 shows the torque for Mesh 2 with the DTR approach and the conservative and convective forms of the ST-VMS formulation. The conservative-form computation is with the standard time-step size: 2.23×10^{-3} s (145 time steps per patch). Figure 18 shows the torque for the individual blades of Mesh 2 with the DTR approach. The figure clearly shows the expected torque drop for each blade as it passes the tower, while the other two blades maintain relatively constant torque. Figure 19 shows the torque for 10 equal-length spanwise sections of a blade of Mesh 2 with the DTR approach. Greatest amount of torque is generated in sections 6–9 of the blade, while section 10 at the tip and the other lower sections generate less torque. Figure 20 shows a volume rendering of the vorticity for Mesh 2 with the DTR approach. The flow patterns vary considerably along each blade length, illustrating the necessity to carry out the computations in 3D.

Figure 21 shows the pressure coefficient at $0.90R$ for the last 0° orientation of a blade of Mesh 2, with the DTR and IMTR approaches, with the last 0° orientation being common between the two computations. There is very little difference in the pressure coefficient around the blades between the DTR and IMTR approaches. Figure 22 shows the pressure coefficient at $0.90R$ for the last 180° orientation of a blade of Mesh 1, Mesh 2 and Mesh 3, with the DTR approach, with the last 180° orientation being common between Mesh 2 and Mesh 3 computations. Averaged torque (in MN·m) for the last 360° rotation for Mesh 1, 2 and 3 are 2.31, 2.34 and 2.39 for the DTR approach, and 2.32, 2.34 and 2.35 for the IMTR approach. The values show that the difference in torque between the DTR and IMTR approaches,

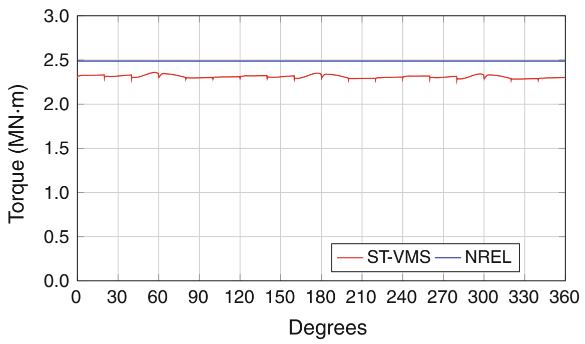


Fig. 11 Torque for Mesh 1 with the DTR approach, compared with the NREL data

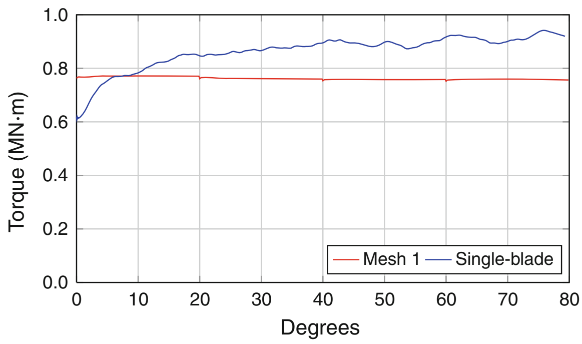


Fig. 12 Torque for a single blade of Mesh 1 with the DTR approach, compared with the torque from an earlier single-blade computation [73] using the TGI option of ν_{C} $(=\nu_{\mathrm{LSIC}})$

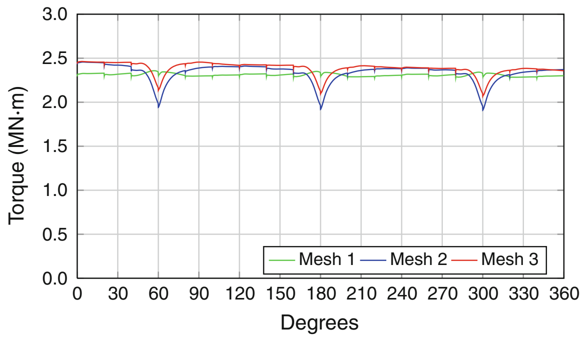


Fig. 13 Torque for Mesh 1, Mesh 2 and Mesh 3 with the DTR approach

and between Mesh 2 and Mesh 3, is rather small. The difference in torque between Mesh 1 and Mesh 2 and 3 illustrates effect of the tower.

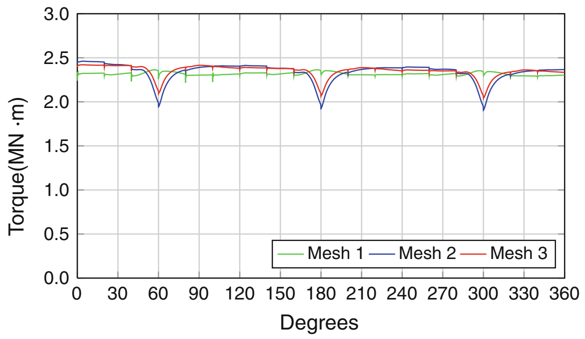


Fig. 14 Torque for Mesh 1, Mesh 2 and Mesh 3 with the IMTR approach

4 Micon 65/13M Wind Turbine with a CX-100 Blade

This section is adapted from [87]. We simulate the Micon 65/13M wind turbine at field test conditions [94]. Micon 65/13M is a three-blade, horizontal-axis, fixed-pitch, upwind turbine with the total rotor diameter of 19.3 m and rated power of 100 kW. The hub is located at the height of 23 m. The wind turbine stands on a tubular steel tower, with a base diameter of 1.9 m. The drive train generator operates at 1,200 rpm, while the rotor spins at a nominal speed of 55 rpm. The Micon 65/13M wind turbine was used for the Long-Term Inflow and Structural Testing (LIST) program [95] initiated by Sandia National Laboratories in 2001 to explore the use of carbon fiber in wind turbine blades. Three experimental blade prototypes, GX-100, CX-100 and TX-100, were developed specifically for this project. We use the CX-100 conventional carbon-spar blade design [94, 96]. The NREL S821, S819 and S820 airfoils are used to define the blade geometry. The details of the blade geometry definition are provided in Table 1.

4.1 Eigenfrequency Analysis of the CX-100 Blade

The blade structure is comprised of five primary sections: leading edge, trailing edge, root, spar cap, and shear web. The sections are shown in Fig. 23. Each section is further subdivided into zones, each consisting of a multilayer composite layup. There is a total of 32 zones with constant total thickness and unique laminate stacking. The effective material properties for each of the zones are computed using the procedures described in [50, 74]. All 32 zones are identified on the blade surface and are shown in Fig. 23. For more details of the material composition of the CX-100 blade see [87]. We perform eigenfrequency calculations of the CX-100 blade using three quadratic NURBS meshes. The coarsest mesh has 1,846 elements, while the finest mesh has 18,611. The mesh statistics are summarized in Table 2. The eigenfrequency results

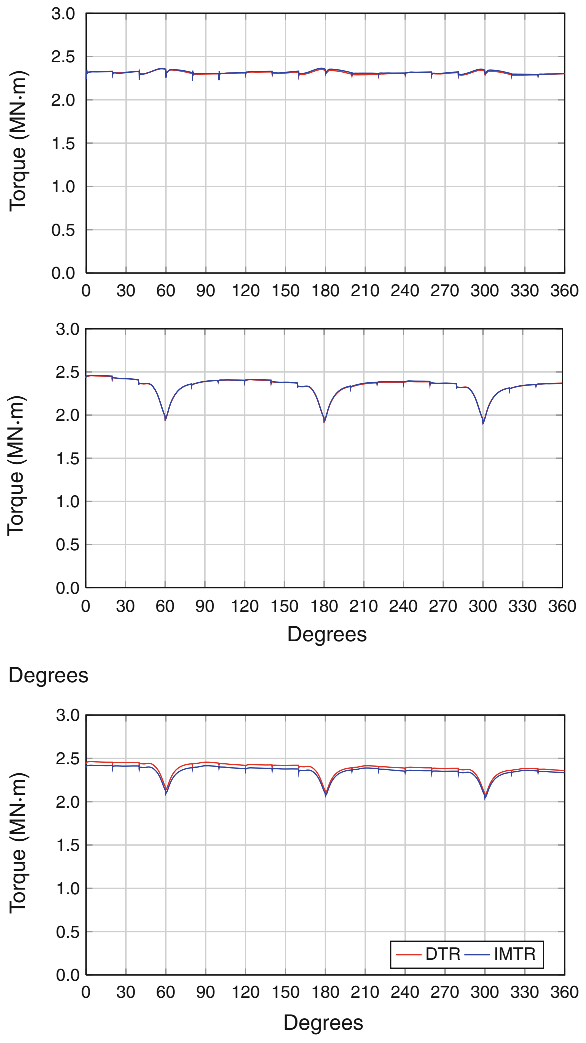


Fig. 15 Torque with the DTR and IMTR approaches for Mesh 1, Mesh 2, and Mesh 3

are compared with the experimental data from [97, 98]. We compute the case with free boundary conditions and the case when the blade is clamped at the root. In both cases, the computed natural frequencies are in good agreement with the experimental data (see Tables 3 and 4). The medium mesh shows a good balance between the computational cost and accuracy. For this reason, this mesh is chosen for the FSI computations presented here. The mode shapes computed using the medium mesh for the clamped case are shown in Fig. 24.

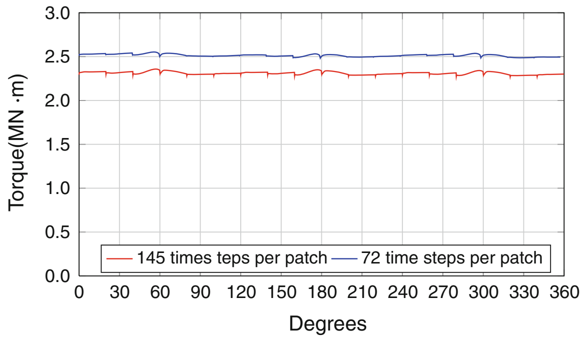


Fig. 16 Torque for Mesh 1 with the DTR approach, using two different time-step sizes: 2.23×10^{-3} s (145 *time steps per patch*) and 4.49×10^{-3} s (72 *time steps per patch*)

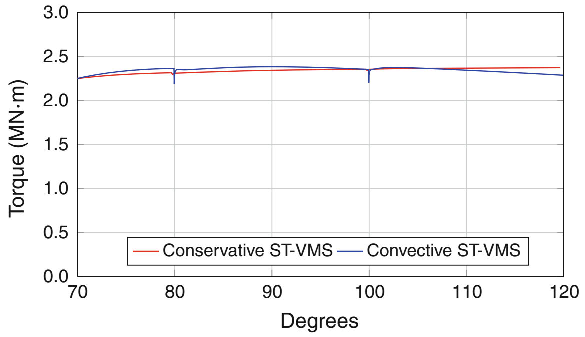


Fig. 17 Torque for Mesh 2 with the DTR approach and the conservative and convective forms of the ST-VMS formulation. *The time-step sizes* 4.46×10^{-4} s (725 *time steps per patch*) for the convective form and 2.23×10^{-3} s (145 *time steps per patch*) for the conservative form. The torques are from the same period in a rotation cycle, but the conservative-form torque is from the last 360° of the computation, and the convective-form torque is from a recently-started, ongoing computation

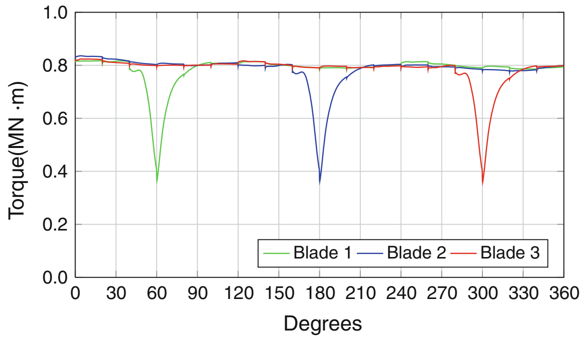


Fig. 18 Torque for the individual blades of Mesh 2 with the DTR approach

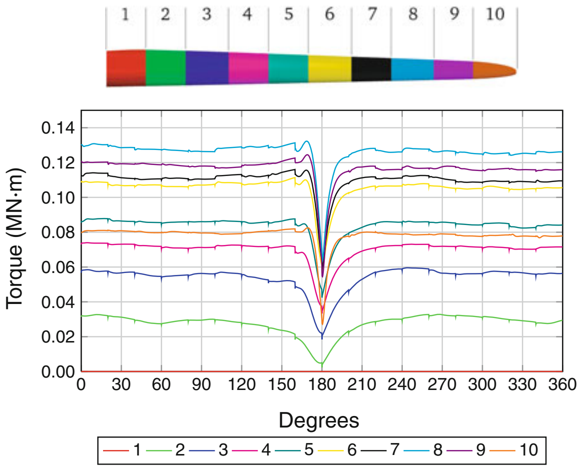


Fig. 19 Torque for 10 equal-length spanwise sections of a blade of Mesh 2 with the DTR approach

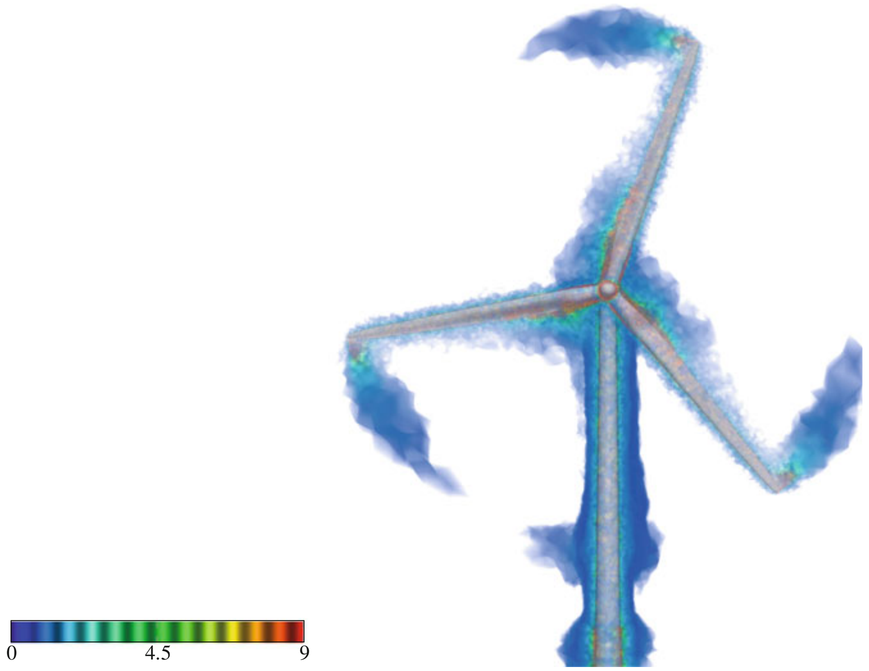


Fig. 20 Volume rendering of the vorticity (in s^{-1}) from the last 360° of the computation for Mesh 2 with the DTR approach

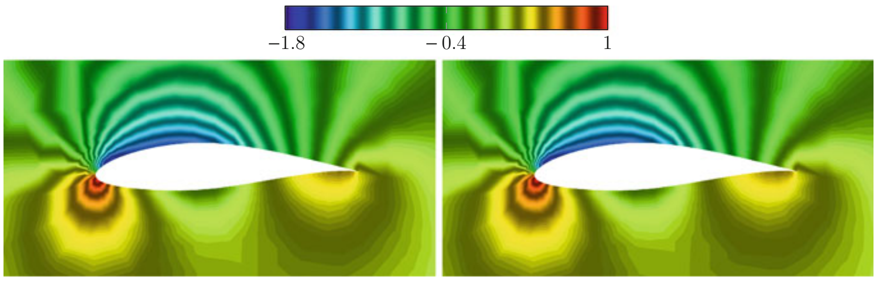


Fig. 21 Pressure coefficient at $0.90R$ for the last 0° orientation of a blade of Mesh 2, with the DTR (*left*) and IMTR (*right*) approaches

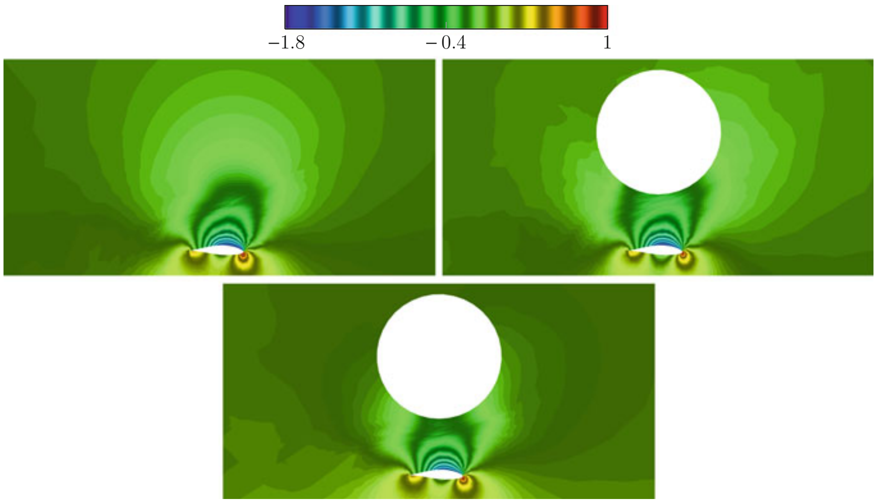


Fig. 22 Pressure coefficient at $0.90R$ for the last 180° orientation of a blade of Mesh 1, Mesh 2, and Mesh 3, with the DTR approach

4.2 Aerodynamics and FSI Computations

In this section, we present aerodynamic and FSI simulations. For both cases, a constant inflow wind speed of 10.5 m/s and fixed rotor speed of 55 rpm are prescribed. These correspond to the operating conditions reported for the field tests in [94]. The air density and viscosity are 1.23 kg/m^3 and 1.78×10^{-5} kg/(m·s), respectively. Zero traction boundary conditions are prescribed at the outflow and no-penetration boundary conditions are prescribed at the top, bottom, and side surfaces of the outer (stationary) computational domain. No-slip boundary conditions are prescribed at the rotor, nacelle, and tower, and are imposed weakly.

Figure 25 shows the computational domain and mesh used in this study. The mesh consists of 5,134,916 linear elements, which are triangular prisms in the rotor

Table 1 CX-100 blade with “RNodes” (m), “Chord” (m), “AeroTwst” (°), and “Airfoil” type

RNodes	Chord	AeroTwst	Airfoil
0.200	0.356	29.6	Cylinder
0.600	0.338	24.8	Cylinder
1.000	0.569	20.8	Cylinder
1.400	0.860	17.5	NREL S821
1.800	1.033	14.7	NREL S821
2.200	0.969	12.4	NREL S821
3.200	0.833	8.3	NREL S821
4.200	0.705	5.8	NREL S819
5.200	0.582	4.0	NREL S819
6.200	0.463	2.7	NREL S819
7.200	0.346	1.4	NREL S819
8.200	0.232	0.4	NREL S819
9.000	0.120	0.0	NREL S820

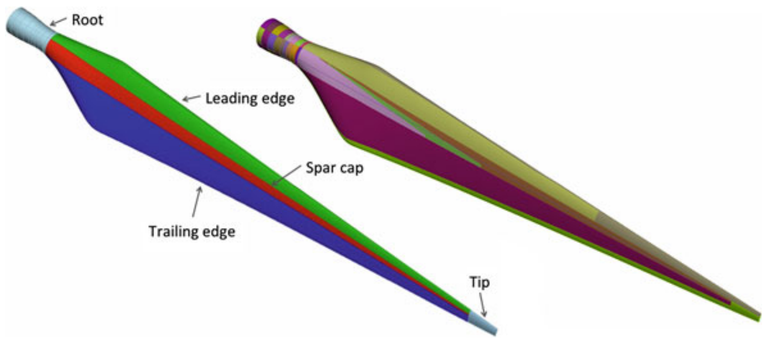


Fig. 23 *Left* Five primary sections of the CX-100 blade; *Right* 32 distinct material zones of the CX-100 blade

Table 2 NURBS blade meshes used in the eigenfrequency analysis

	Control points	Elements
Mesh 1	3,469	1,846
Mesh 2	7,411	4,647
Mesh 3	25,896	18,611

boundary layers and tetrahedra everywhere else in the domain. The mesh is refined in the rotor and tower regions for better flow resolution near the wind turbine. The size of the first element in the wall-normal direction is 0.002 m, and 15 layers of prismatic elements were generated with a growth ratio of 1.2. Figure 25 shows a 2D blade cross-section at 70 % spanwise station to illustrate the boundary-layer mesh used in the computations. The time-step size is set to 3.0×10^{-5} s. In Fig. 26 the time history of the aerodynamic torque is plotted. As can be seen from the plot, using FSI, we

Table 3 Comparison of experimentally measured and computed natural frequencies (in Hz) for the free case

	Mode 1	Mode 2	Mode 3
Mesh 1	8.28	15.92	19.26
Mesh 2	8.22	15.61	18.21
Mesh 3	8.22	15.6	18.01
Experiment	7.6–8.2	15.7–18.1	20.2–21.3

Mode 1 is the first flapwise mode, Mode 2 is the first edgewise mode, and Mode 3 is the second flapwise mode

Table 4 Comparison of experimentally measured and computed natural frequencies (in Hz) for the clamped case

	Mode 1	Mode 2	Mode 3
Mesh 1	4.33	11.82	19.69
Mesh 2	4.29	11.61	19.08
Mesh 3	4.27	11.54	18.98
Experiment	4.35	11.51	20.54

Modes 1–3 are the first three flapwise bending modes

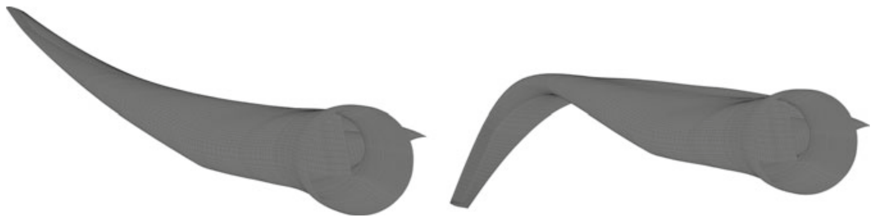


Fig. 24 First (*left*) and second (*right*) flapwise bending mode for the clamped case

capture the high frequency oscillations caused by the bending and torsional motions of the blades. In the case of the rigid blade the only high-frequency oscillations in the torque curve are due to the trailing-edge turbulence. For the rigid blade case the effect of the tower on the aerodynamic torque is more pronounced, while in the case of FSI it is not as visible due to the relatively high torque oscillations. The "dips" in the aerodynamic torque can be seen at 60°, 180°, and 300° azimuthal angle, which is precisely when one of the three blades is passing the tower. The computed values of the aerodynamic torque are plotted together with field test results from [94]. The upper and lower dashed lines indicate the aerodynamic torque bounds, while the middle dashed line gives its average value. Both the aerodynamic and FSI results compare very well with the field test data. Figure 27 shows the relative wind speed at the 70 % spanwise station rotated to the reference configuration to illustrate the blade deflection and complexity of boundary-layer turbulent flow. Figure 28 shows the flow field as the blade passes the tower.

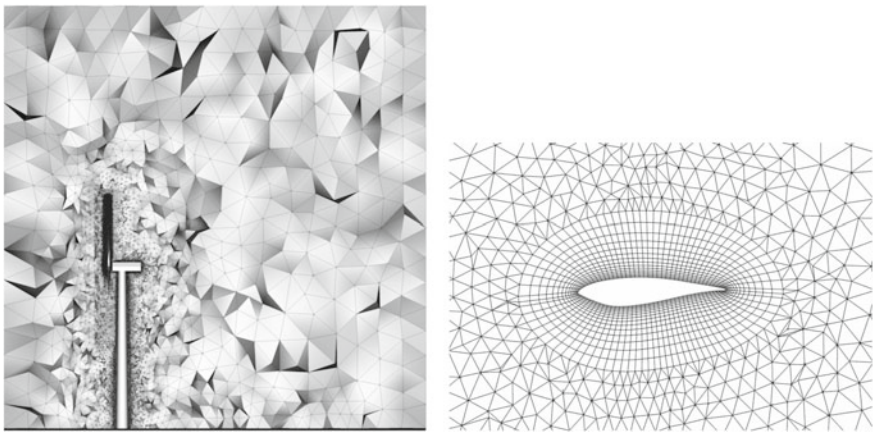


Fig. 25 *Left* Computational domain and mesh with the refined inner region for better flow resolution near the rotor; *Right* 2D blade cross-section at $r/R = 70\ \%$ and the boundary-layer mesh

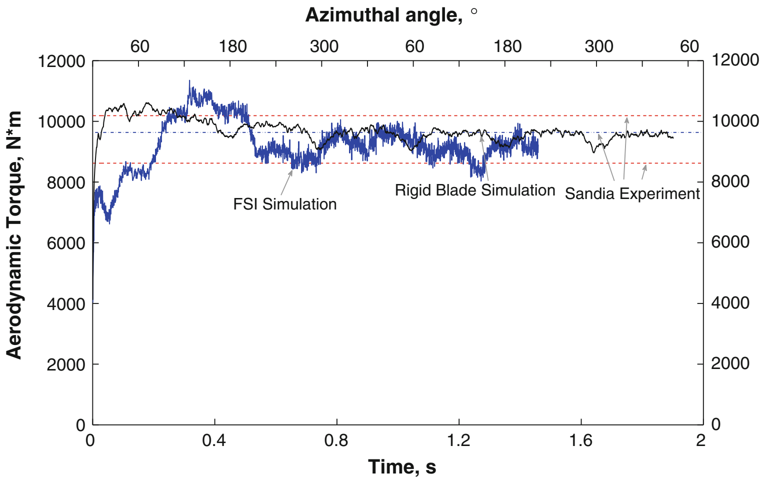


Fig. 26 Aerodynamic torque for the FSI and rigid-blade simulations. The experimental range for the torque and its average are provided for comparison and are plotted using *dashed lines*

5 Concluding Remarks

We provided an overview of the aerodynamic and FSI analysis of wind turbines carried out in recent years with the ALE-VMS and ST-VMS methods. The techniques complementing these core methods include weak enforcement of the essential boundary conditions, NURBS-based isogeometric analysis, using NURBS basis functions in temporal representation of the rotor motion, mesh motion and also in remeshing, rotation representation with constant angular velocity, Kirchhoff–Love shell mod-

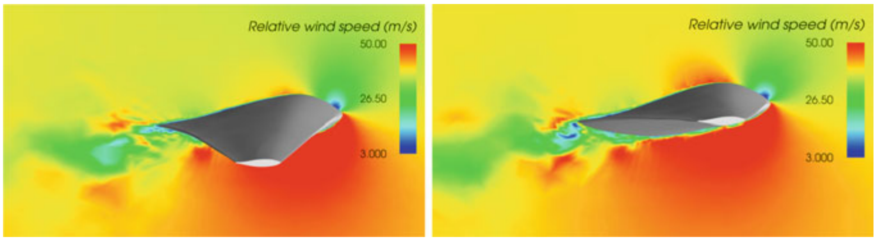


Fig. 27 Relative wind speed at the 70 % spanwise station for the FSI simulation at $t = 0.86$ s (*left*) and $t = 1.06$ s (*right*). The blade deflection is clearly visible

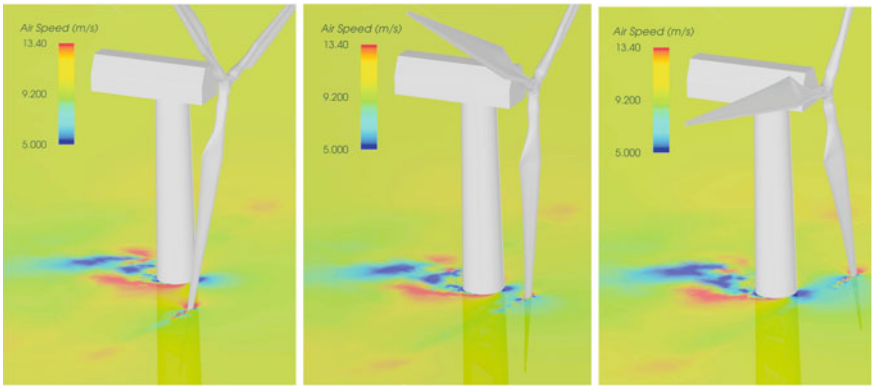


Fig. 28 Wind speed contours at 80 % spanwise station as the blade passes the tower

eling of the rotor-blade structure, and full FSI coupling. Some of these techniques were included in our overview. The wind-turbine analysis cases presented include the aerodynamics of wind-turbine rotor and tower and the FSI that accounts for the deformation of the rotor blades. The specific wind turbines considered were NREL 5MW, NREL Phase VI and Micon 65/13M, all at full scale. In the case of NREL Phase VI and Micon 65/13M we also presented a successful comparison with the experimental data. Overall, this article demonstrates that the ALE-VMS and ST-VMS methods, together with some new supporting techniques, have brought the aerodynamic and FSI analysis of wind turbines to a new level, where such analyses can contribute more to simulation-based design and testing.

Acknowledgments We wish to thank the Texas Advanced Computing Center (TACC) and the San Diego Supercomputing Center (SDSC) for providing HPC resources that have contributed to the research results reported in this article. The first author acknowledges the support of the NSF CAREER Award, the NSF Award CBET-1306869, and the Air Force Office of Scientific Research Award FA9550-12-1-0005. The ST-VMS part of the work was supported by ARO grants W911NF-09-1-0346 and W911NF-12-1-0162 (third author) and Rice–Waseda Research Agreement (second author).

References

1. Jonkman JM, Buhl ML (2005) "FAST user's guide", Technical Report NREL/EL-500-38230. National Renewable Energy Laboratory, Golden, CO
2. Jonkman J, Butterfield S, Musial W, Scott G (2009) "Definition of a 5-MW reference wind turbine for offshore system development", Technical Report NREL/TP-500-38060. National Renewable Energy Laboratory, Golden, CO
3. Sørensen NN, Michelsen JA, Schreck S (2002) Navier-Stokes predictions of the NREL Phase VI rotor in the NASA Ames 80 ft $\times$ 120 ft wind tunnel. Wind Energy 5:151–169
4. Pape AL, Lecanu J (2004) 3D Navier-Stokes computations of a stall-regulated wind turbine. Wind Energy 7:309–324
5. Bazilevs Y, Hsu M-C, Akkerman I, Wright S, Takizawa K, Henicke B, Spielman T, Tezduyar TE (2011) 3D simulation of wind turbine rotors at full scale. Part I: geometry modeling and aerodynamics. Int J Numer Meth Fluids 65:207–235. doi:10.1002/fld.2400
6. Takizawa K, Henicke B, Tezduyar TE, Hsu M-C, Bazilevs Y (2011) Stabilized space-time computation of wind-turbine rotor aerodynamics. Comput Mech 48:333–344. doi:10.1007/s00466-011-0589-2
7. Kong C, Bang J, Sugiyama Y (2005) Structural investigation of composite wind turbine blade considering various load cases and fatigue life. Energy 30:2101–2114
8. Hansen MOL, Sørensen JN, Voutsinas S, Sørensen N, Madsen HA (2006) State of the art in wind turbine aerodynamics and aeroelasticity. Prog Aerosp Sci 42:285–330
9. Jensen FM, Falzon BG, Ankersen J, Stang H (2006) Structural testing and numerical simulation of a 34 m composite wind turbine blade. Compos Struct 76:52–61
10. Kiendl J, Bazilevs Y, Hsu M-C, Wüchner R, Bletzinger K-U (2010) The bending strip method for isogeometric analysis of Kirchhoff-Love shell structures comprised of multiple patches. Comput Methods Appl Mech Eng 199:2403–2416
11. Bazilevs Y, Hsu M-C, Kiendl J, Benson DJ (2012) A computational procedure for pre-bending of wind turbine blades. Int J Numer Meth Eng 89:323–336
12. Bazilevs Y, Hsu M-C, Kiendl J, Wüchner R, Bletzinger K-U (2011) 3D simulation of wind turbine rotors at full scale. Part II: fluid-structure interaction modeling with composite blades. Int J Numer Meth Fluids 65:236–253
13. Hughes TJR, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. Comput Methods Appl Mech Eng 194:4135–4195
14. Cottrell JA, Reali A, Bazilevs Y, Hughes TJR (2006) Isogeometric analysis of structural vibrations. Comput Methods Appl Mech Eng 195:5257–5297
15. Bazilevs Y, da Veiga LB, Cottrell JA, Hughes TJR, Sangalli G (2006) Isogeometric analysis: approximation, stability and error estimates for h-refined meshes. Math Models Methods Appl Sci 16:1031–1090
16. Cottrell JA, Hughes TJR, Reali A (2007) Studies of refinement and continuity in isogeometric structural analysis. Comput Meth Appl Mech Eng 196:4160–4183
17. Cottrell JA, Hughes TJR, Bazilevs Y (2009) Isogeometric analysis: toward integration of CAD and FEA. Wiley, Chichester
18. Eva*n*s JA, Bazilevs Y, Babuš*k*a I, Hughes TJR (2009) n-Widths, supinfs, and optimality ratios for the k-version of the isogeometric finite element method. Comput Methods Appl Mech Eng 198:1726–1741
19. Dörfel MR, Jüttler B, Simeon B (2010) Adaptive isogeometric analysis by local h-refinement with T-splines. Comput Methods Appl Mech Eng 199:264–275
20. Bazilevs Y, Calo VM, Cottrell JA, Evans JA, Hughes TJR, Lipton S, Scott MA, Sederberg TW (2010) Isogeometric analysis using T-splines. Comput Methods Appl Mech Eng 199:229–263
21. Bazilevs Y, Calo VM, Cottrell JA, Hughes TJR, Reali A, Scovazzi G (2007) Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. Comput Methods Appl Mech Eng 197:173–201

22. Bazilevs Y, Michler C, Calo VM, Hughes TJR (2007) Weak dirichlet boundary conditions for wall-bounded turbulent flows. Comput Methods Appl Mech Eng 196:4853–4862
23. Bazilevs Y, Michler C, Calo VM, Hughes TJR (2010) Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes. Comput Methods Appl Mech Eng 199:780–790
24. Akkerman I, Bazilevs Y, Calo VM, Hughes TJR, Hulshoff S (2008) The role of continuity in residual-based variational multiscale modeling of turbulence. Comput Mech 41:371–378
25. Hsu M-C, Bazilevs Y, Calo VM, Tezduyar TE, Hughes TJR (2010) Improving stability of stabilized and multiscale formulations in flow simulations at small time steps. Comput Methods Appl Mech Eng 199:828–840. doi:10.1016/j.cma.2009.06.019
26. Bazilevs Y, Akkerman I (2010) Large eddy simulation of turbulent Taylor-Couette flow using isogeometric analysis and the residual-based variational multiscale method. J Comput Phys 229:3402–3414
27. Elguedj T, Bazilevs Y, Calo VM, Hughes TJR (2008) B-bar and F-bar projection methods for nearly incompressible linear and nonlinear elasticity and plasticity using higher-order nurbs elements. Comput Methods Appl Mech Eng 197:2732–2762
28. Lipton S, Evans JA, Bazilevs Y, Elguedj T, Hughes TJR (2010) Robustness of isogeometric structural discretizations under severe mesh distortion. Comput Methods Appl Mech Eng 199:357–373
29. Benson DJ, Bazilevs Y, De Luycker E, Hsu M-C, Scott M, Hughes TJR, Belytschko T (2010) A generalized finite element formulation for arbitrary basis functions: from isogeometric analysis to XFEM. Int J Numer Meth Eng 83:765–785
30. Benson DJ, Bazilevs Y, Hsu M-C, Hughes TJR (2010) Isogeometric shell analysis: the Reissner-Mindlin shell. Comput Methods Appl Mech Eng 199:276–289
31. Kiendl J, Bletzinger K-U, Linhard J, Wüchner R (2009) Isogeometric shell analysis with Kirchhoff-Love elements. Comput Methods Appl Mech Eng 198:3902–3914
32. Zhang Y, Bazilevs Y, Goswami S, Bajaj C, Hughes TJR (2007) Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow. Comput Methods Appl Mech Eng 196:2943–2959
33. Bazilevs Y, Calo VM, Zhang Y, Hughes TJR (2006) Isogeometric fluid-structure interaction analysis with applications to arterial blood flow. Comput Mech 38:310–322
34. Bazilevs Y, Calo VM, Hughes TJR, Zhang Y (2008) Isogeometric fluid-structure interaction: theory, algorithms, and computations. Comput Mech 43:3–37
35. Isaksen JG, Bazilevs Y, Kvamsdal T, Zhang Y, Kaspersen JH, Waterloo K, Romner B, Ingebrigtsen T (2008) Determination of wall tension in cerebral artery aneurysms by numerical simulation. Stroke 39:3172–3178
36. Bazilevs Y, Hughes TJR (2008) NURBS-based isogeometric analysis for the computation of flows about rotating components. Comput Mech 43:143–150
37. Cirak F, Ortiz M, Schröder P (2000) Subdivision surfaces: a new paradigm for thin shell analysis. Int J Numer Meth Eng 47:2039–2072
38. Cirak F, Ortiz M (2001) Fully c^1-conforming subdivision elements for finite deformation thin shell analysis. Int J Numer Meth Eng 51:813–833
39. Cirak F, Scott MJ, Antonsson EK, Ortiz M, Schröder P (2002) Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision. Comput Aided Des 34:137–148
40. Hughes TJR, Liu WK, Zimmermann TK (1981) Lagrangian-Eulerian finite element formulation for incompressible viscous flows. Comput Methods Appl Mech Eng 29:329–349
41. Hughes TJR (1995) Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles, and the origins of stabilized methods. Comput Methods Appl Mech Eng 127:387–401
42. Hughes TJR, Oberai AA, Mazzei L (2001) Large eddy simulation of turbulent channel flows by the variational multiscale method. Phys Fluids 13:1784–1799
43. Takizawa K, Tezduyar TE (2011) Multiscale space-time fluid-structure interaction techniques. Comput Mech 48:247–267. doi:10.1007/s00466-011-0571-z

44. Takizawa K, Tezduyar TE (2012) Space-time fluid-structure interaction methods. Math Models Methods Appl Sci 22:1230001. doi:10.1142/S0218202512300013
45. Tezduyar TE (1992) Stabilized finite element formulations for incompressible flow computations. Adv Appl Mech 28:1–44. doi:10.1016/S0065-2156(08)70153-4
46. Tezduyar TE, Behr M, Liou J (1992) A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space-time procedure: I. The concept and the preliminary numerical tests. Comput Methods Appl Mech Eng 94:339–351. doi:10.1016/0045-7825(92)90059-S
47. Tezduyar TE, Behr M, Mittal S, Liou J (1992) A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. Comput Methods Appl Mech Eng 94:353–371. doi:10.1016/0045-7825(92)90060-W
48. Tezduyar TE (2003) Computation of moving boundaries and interfaces and stabilization parameters. Int J Numer Meth Fluids 43:555–575. doi:10.1002/fld.505
49. Tezduyar TE, Sathe S (2007) Modeling of fluid-structure interactions with the space-time finite elements: solution techniques. Int J Numer Meth Fluids 54:855–900. doi:10.1002/fld.1430
50. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Computational fluid-structure interaction: methods and applications. Wiley, Chichester, West Sussex, United Kingdom
51. Bazilevs Y, Hughes TJR (2007) Weak imposition of Dirichlet boundary conditions in fluid mechanics. Comput Fluids 36:12–26
52. Nitsche J (1971) Uber ein variationsprinzip zur losung von Dirichlet-problemen bei verwendung von teilraumen, die keinen randbedingungen unterworfen sind. Abh Math Univ Hamburg 36:9–15
53. Arnold DN, Brezzi F, Cockburn B, Marini LD (2002) Unified analysis of discontinuous Galerkin methods for elliptic problems. SIAM J Numer Anal 39:1749–1779
54. Brooks AN, Hughes TJR (1982) Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. Comput Methods Appl Mech Eng 32:199–259
55. Tezduyar TE, Mittal S, Ray SE, Shih R (1992) Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements. Comput Methods Appl Mech Eng 95:221–242. doi:10.1016/0045-7825(92)90141-6
56. Mittal S, Tezduyar TE (1992) A finite element study of incompressible flows past oscillating cylinders and aerofoils. Int J Numer Meth Fluids 15:1073–1118. doi:10.1002/fld.1650150911
57. Mittal S, Tezduyar TE (1995) Parallel finite element simulation of 3d incompressible flows—fluid-structure interactions. Int J Numer Meth Fluids 21:933–953. doi:10.1002/fld.1650211011
58. Kalro V, Tezduyar TE (2000) A parallel 3D computational method for fluid-structure interactions in parachute systems. Comput Methods Appl Mech Eng 190:321–332. doi:10.1016/S0045-7825(00)00204-8
59. Tezduyar TE, Sathe S, Keedy R, Stein K (2006) Space-time finite element techniques for computation of fluid-structure interactions. Comput Methods Appl Mech Eng 195:2002–2027. doi:10.1016/j.cma.2004.09.014
60. Takizawa K, Tezduyar TE (2012) Computational methods for parachute fluid-structure interactions. Arch Comput Meth Eng 19:125–169. doi:10.1007/s11831-012-9070-4
61. Tezduyar TE, Takizawa K, Brummer T, Chen PR (2011) Space-time fluid-structure interaction modeling of patient-specific cerebral aneurysms. Int J Numer Meth Biomed Eng 27:1665–1710. doi:10.1002/cnm.1433
62. Takizawa K, Bazilevs Y, Tezduyar TE (2012) Space-time and ALE-VMS techniques for patient-specific cardiovascular fluid-structure interaction modeling. Arch Comput Meth Eng 19:171–225. doi:10.1007/s11831-012-9071-3
63. Takizawa K, Schjodt K, Puntel A, Kostov N, Tezduyar TE (2012) Patient-specific computer modeling of blood flow in cerebral arteries with aneurysm and stent. Comput Mech 50:675–686. doi:10.1007/s00466-012-0760-4

64. Takizawa K, Fritze M, Montes D, Spielman T, Tezduyar TE (2012) Fluid-structure interaction modeling of ringsail parachutes with disreefing and modified geometric porosity. Comput Mech 50:835–854. doi:10.1007/s00466-012-0761-3
65. Takizawa K, Montes D, Fritze M, McIntyre S, Boben J, Tezduyar TE (2013) Methods for FSI modeling of spacecraft parachute dynamics and cover separation. Math Models Methods Appl Sci 23:307–338. doi:10.1142/S0218202513400058
66. Takizawa K, Tezduyar TE, Boben J, Kostov N, Boswell C, Buscher A (2013) Fluid-structure interaction modeling of clusters of spacecraft parachutes with modified geometric porosity. Comput Mech 52:1351–1364. doi:10.1007/s00466-013-0880-5
67. Takizawa K, Schjodt K, Puntel A, Kostov N, Tezduyar TE (2013) Patient-specific computational analysis of the influence of a stent on the unsteady flow in cerebral aneurysms. Comput Mech 51:1061–1073. doi:10.1007/s00466-012-0790-y
68. Manguoglu M, Takizawa K, Sameh AH, Tezduyar TE (2011) Nested and parallel sparse algorithms for arterial fluid mechanics computations with boundary layer mesh refinement. Int J Numer Meth Fluids 65:135–149. doi:10.1002/fld.2415
69. Manguoglu M, Takizawa K, Sameh AH, Tezduyar TE (2011) A parallel sparse algorithm targeting arterial fluid mechanics computations. Comput Mech 48:377–384. doi:10.1007/s00466-011-0619-0
70. Tezduyar T, Aliabadi S, Behr M, Johnson A, Kalro V, Litke M (1996) Flow simulation and high performance computing. Comput Mech 18:397–412. doi:10.1007/BF00350249
71. Behr M, Tezduyar T (1999) The Shear-slip mesh update method. Comput Methods Appl Mech Eng 174:261–274. doi:10.1016/S0045-7825(98)00299-0
72. Behr M, Tezduyar T (2001) Shear-slip mesh update in 3D computation of complex flow problems with rotating mechanical components. Comput Methods Appl Mech Eng 190:3189–3200. doi:10.1016/S0045-7825(00)00388-1
73. Takizawa K, Henicke B, Montes D, Tezduyar TE, Hsu M-C, Bazilevs Y (2011) Numerical-performance studies for the stabilized space-time computation of wind-turbine rotor aerodynamics. Comput Mech 48:647–657. doi:10.1007/s00466-011-0614-5
74. Bazilevs Y, Hsu M-C, Takizawa K, Tezduyar TE (2012) ALE-VMS and ST-VMS methods for computer modeling of wind-turbine rotor aerodynamics and fluid-structure interaction. Math Models and Methods Appl Sci 22:1230002. doi:10.1142/S0218202512300025
75. Takizawa K, Tezduyar TE, McIntyre S, Kostov N, Kolesar R, Habluetzel C (2014) Space-time VMS computation of wind-turbine rotor and tower aerodynamics. Comput Mech 53:1–15. doi:10.1007/s00466-013-0888-x
76. Takizawa K, Henicke B, Puntel A, Spielman T, Tezduyar TE (2012) Space-time computational techniques for the aerodynamics of flapping wings. J Appl Mech 79:010903. doi:10.1115/1.4005073
77. Takizawa K, Henicke B, Puntel A, Kostov N, Tezduyar TE (2012) Space-time techniques for computational aerodynamics modeling of flapping wings of an actual locust. Comput Mech 50:743–760. doi:10.1007/s00466-012-0759-x
78. Takizawa K, Kostov N, Puntel A, Henicke B, Tezduyar TE (2012) Space-time computational analysis of bio-inspired flapping-wing aerodynamics of a micro aerial vehicle. Comput Mech 50:761–778. doi:10.1007/s00466-012-0758-y
79. Takizawa K, Henicke B, Puntel A, Kostov N, Tezduyar TE (2013) Computer modeling techniques for flapping-wing aerodynamics of a locust. Comput Fluids 85:125–134. doi:10.1016/j.compfluid.2012.11.008
80. Tezduyar TE, Behr M, Mittal S, Johnson AA (1992) Computation of unsteady incompressible flows with the finite element methods—space-time formulations, iterative strategies and massively parallel implementations. In: New methods in transient analysis, PVP-vol 246/AMD-vol 143, ASME, New York, pp 7–24
81. Tezduyar T, Aliabadi S, Behr M, Johnson A, Mittal S (1993) Parallel finite-element computation of 3D flows. Computer 26:27–36. doi:10.1109/2.237441
82. Johnson AA, Tezduyar TE (1994) Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. Comput Meth Appl Mech Eng 119:73–94. doi:10.1016/0045-7825(94)00077-8

83. Tezduyar TE (2001) Finite element methods for flow problems with moving boundaries and interfaces. Arch Comput Meth Eng 8:83–130. doi:10.1007/BF02897870
84. Hsu M-C, Bazilevs Y (2012) Fluid-structure interaction modeling of wind turbines: simulating the full machine. Comput Mech 50:821–833
85. Hsu M-C, Akkerman I, Bazilevs Y (2013) Finite element simulation of wind turbine aerodynamics: validation study using NREL Phase VI experiment. Wind Energy, published online. doi:10.1002/we.1599
86. Hand MM, Simms DA, Fingersh LJ, Jager DW, Cotrell JR, Schreck S, Larwood SM (2001) "Unsteady aerodynamics experiment Phase VI: wind tunnel test configurations and available data campaigns", Technical Report NREL/TP-500-29955. National Renewable Energy Laboratory, Golden, CO
87. Korobenko A, Hsu M-C, Akkerman I, Tippmann J, Bazilevs Y (2013) Structural mechanics modeling and fsi simulation of wind turbines. Math Models and Methods Appl Sci 23:249–272
88. Bazilevs Y, Hsu M-C, Scott MA (2012) Isogeometric fluid-structure interaction analysis with emphasis on non-matching discretizations, and with application to wind turbines. Comput Meth Appl Mech Eng 249–252:28–41
89. Tezduyar TE, Sathe S, Stein K (2006) Solution techniques for the fully-discretized equations in computation of fluid-structure interactions with the space-time formulations. Comput Meth Appl Mech Eng 195:5743–5753. doi:10.1016/j.cma.2005.08.023
90. Jonkman J, Butterfield S, Musial W, Scott G (2009) "Definition of a 5-MW reference wind turbine for offshore system development", Technical Report NREL/TP-500-38060, National Renewable Energy Laboratory
91. Spera DA (1994) Introduction to modern wind turbines. In: Spera DA (ed) Wind turbine technology: fundamental concepts of wind turbine engineering, pp 47–72 (ASME Press)
92. Saad Y, Schultz M (1986) GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comput 7:856–869
93. Karypis G, Kumar V (1998) A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J Sci Comput 20:359–392
94. Zayas JR, Johnson WD (2008) 3X-100 blade field test, Report of the Sandia National Laboratories. Wind Energy Technology Department, Sandia
95. Sutherland JH, Jones PL, Neal BA (2001) The long-term inflow and structural test program. In: Proceedings of the 2001 ASME wind energy symposium, p 162
96. Berry D, Ashwill T (2007) Design of 9-meter carbon-fiberglass prototype blades: CX-100 and TX-100", Report of the Sandia national laboratories, New Mexico, USA
97. White JR, Adams DE, Rumsey MA (2011) Modal analysis of CX-100 rotor blade and Micon 65/13 wind turbine. In: Structural dynamics and renewable energy, vol 1, Conference proceedings of the society for experimental mechanics series 10
98. Marinone T, LeBlanc B, Harvie J, Niezrecki C, Avitabile P (2012) Modal testing of a 9 m CX-100 turbine blade. In: Topics in experimental dynamics substructuring and wind turbine dynamics, vol 2, Conference proceedings of the society for experimental mechanics series 27

Part VII
Partitioned Method and Parallelization Techniques

Scaling Up Multiphysics

Rainald Löhner and Joseph D. Baum

Abstract The present paper summarizes trends in supercomputing and the consequences they will have on coupled problems in computational mechanics. The appearance of parallel machines with more than 10^6 cores implies that the prevalent 'scalar-pre, simple parallel solve, scalar-post' environment will have to give way to a completely scalable simulation pipeline. The grid size alone will force distributed parallelization of meshing, domain splitting, load balancing and post-processing. Possible ways of addressing parallel meshing and dynamic load balancing for multiphysics are treated, and examples shown.

1 Introduction 1: Computational Sciences

Computational sciences (i.e. computation-based sciences) have become the third pillar of the empirical sciences (besides the traditional experiments and theory). It has become inconceivable to carry out experiments or develop a new theory without the heavy support of calculations during all stages of research. To name a few: Any large-scale experiment in physics (e.g. particle physics), aeronautics (e.g. wind-tunnel), automotive (e.g. wind tunnel), naval engineering (e.g. water tunnel), civil engineering (e.g. ultimate loads), telecom engineering (e.g. efficiency of mobile communications) is nowadays preceded by a lengthy series of pre-experiment calculations (for example to make sure that measuring devices are operating in their expected ranges) and followed by another lengthy series of post-experiment calculations (for example to

R. Löhner (✉)
Center for Computational Fluid Dynamics, George Mason University,
M.S. 6A2, Fairfax, VA 22030-4444, USA
e-mail: rlohner@gmu.edu

J. D. Baum
Advanced Technology Group, SAIC, McLean, VA 22020, USA
e-mail: joseph.d.baum@saic.com

understand in depth the phenomena observed). The same trend can be observed in all the manufacturing industries, and increasingly in the medical and life sciences. Any car, truck, train, ship, airplane, large building, bridge, skyscraper, computer, medical device or, for that matter, consumer product of value will see considerable calculation based design and optimization during its development. Computational mechanics is also increasingly being used not only in the design, optimization and verification of finished products, but also for the optimization of manufacturing processes.

The current trend in each of the fields that comprise the computational sciences (structural/ thermo/ fluid-dynamics, electromagnetics, material science, chemistry, etc.) is to increase physical fidelity (either by considering more scales or **linking/ coupling** different disciplines), improve accuracy and robustness, and push the range of feasible (credible) problem classes. Furthermore, high-end computing of this kind is increasingly being used to provide data bases for fast-running engineering models. This new area, also known as real-time computing, either interpolates from these detailed data bases, or extracts fundamental modes of the system to obtain a reduced order model (ROM).

Increased physical fidelity and accuracy in almost all fields in applied sciences (i.e. physics, chemistry, biology, medicine, etc.) and engineering (civil, mechanical, aerospace, naval, electrical, telecom, etc.) naturally imply large computing requirements.

2 Introduction 2: The Red Shift

Scientific computing in general and supercomputing in particular have taken advantage of a remarkable combination of advances over the last three decades: on the one hand the number of transistors per area has doubled every 18 months (so-called Moore's law), and clockrates increased by two orders of magnitude. For programmers and users, this perfect combination led to an almost utopian environment: one could safely assume that without any further effort, the speed of codes would automatically increase due to clockrates and larger register/ cache/ memory, and larger problems could be run due to larger memory. However, due to physical limitations, these trends could not continue unabated. Given that heat generation increases with the third power of the clockrate, a clear limit is in sight here. In fact, anyone who has inspected a recent motherboard can testify that most of the mechanical engineering effort is devoted to chip cooling. Over the last five years, clockrates have stalled at 2.0–3.0 GHz, and all indications are that, if anything, they will decrease in the future. As far as packing more transistors per area, indications are that Moore's law will continue for the foreseeable future (a decade). The only way to higher CPU performance (loosely speaking: more floating point operations per second [FLOPS]) is then via massive parallelism. This can be achieved (and is pursued) at the level of the chip (either via many cores or via specialized hardware, e.g. GPUs), via a network of chips, or via a combination of both approaches. In fact, most of the Top 500 supercomputers at present use a combination of this kind to achieve outstanding performance.

The biggest problem facing supercomputer designers (and scientific programmers) is that the speed increases of subcomponents continue to advance at very different rates: peak processor speed advances much faster than memory transfer rates, which in turn advance much faster than DRAM latencies, which in turn advance much faster than the interconnect switches between processors. This so-called 'red shift' has led to a crisis for computer architects: current designs are driven by complex latency-hiding mechanisms.

Field solvers, which are commonly used for computational fluid and solid mechanics as well as electromagnetics, only perform a limited number of operations for the data that is brought in and out of memory. The gains obtainable via better compilers, prefetching, mixing floating point operations and memory fetches, etc. have already largely been exploited during the last decade. At present, even at the chip or GPU level, field solvers are already limited by memory transfer rates, as evidenced by recent studies [30]. This implies that for field solvers, this 'red shift' is particularly worrisome.

In order to see what impact this 'red shift' has on current large-scale production machines, the FEM-FCT algorithm [28, 31] as implemented in the FEFLO code was tried on two representative systems: Jaguar and Diamond. FEM-FCT is used on a daily basis for many blast-structure interaction simulations [25, 45, 46].

Diamond is an SGI ICE machine composed of 1,920 nodes (or blades). Each node is composed of 2 Quad-core Xeon processors Nehalem E5440 series, giving a total of 15,360 cores. The Xeon processors run at 2.8 GHz. This machine reaches a peak performance of 172 Teraflops. The main memory has a capacity of 45 Terabytes. The interconection network is composed of two Infiniband interconnect (10/20 GB/s), one interconnect is dedicated to I/O and the other to MPI traffic.

Jaguar, at the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, is a CRAY machine composed of 18,688 compute nodes in addition to dedicated login/service nodes. Each compute node contains dual hex-core AMD Opteron 2435 (Istanbul) processors running at 2.6 GHz, 16 GB of DDR2-800 memory, and a SeaStar 2+ router with a peak bandwidth of 57.6 GB/s. The resulting partition contains 224,256 processing cores, 300TB of memory, and a peak performance of 2.3 petaflops. The operating system is the Cray Linux Environment.

The case, shown in Fig. 1, was a blast in a room. This testcase has been used for many hardware benchmarks over the years. The particular mesh used had about `nelem=40 Mels`.

Figures 2 and 3 show a compilation of timings obtained for different numbers of domains and cores per domain. Note that once a core deals with more than `nelem=300,000`, CPU performance asymptotes out to the values expected for the chips used in these machines ($O(10^{-6})$ $[sec/elem/step]$ or $O(0.55 \cdot 10^{-5})$ $[sec/pt/step]$). However, for a very small number of elements per domain, there appears a constant delay per timestep of $T_{MPI} = O(0.1)$ $[sec]$ (!). Given that we have $O(10^2)$ MPI messages/exchanges per timestep, it appears that every time a message is started via an MPI call, a latency of $T_l = O(10^{-3})$ $[sec]$ is incurred.

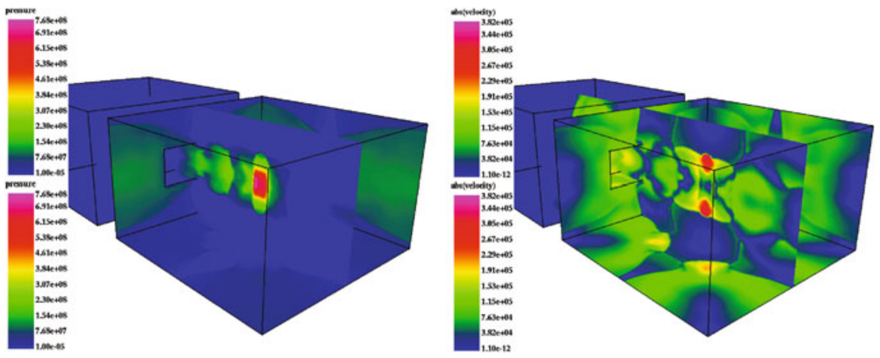


Fig. 1 Blast in room: Pressures and velocities

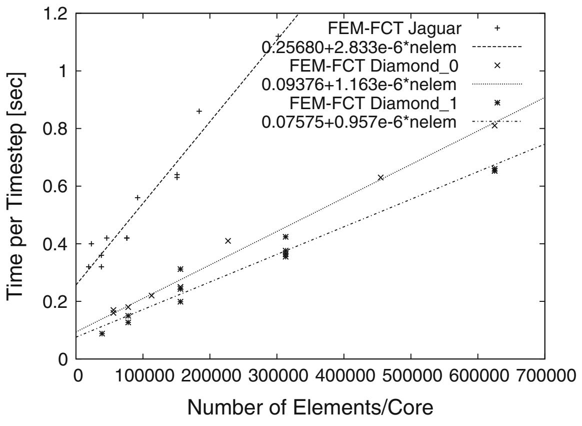


Fig. 2 Blast in room: Timings (absolute)

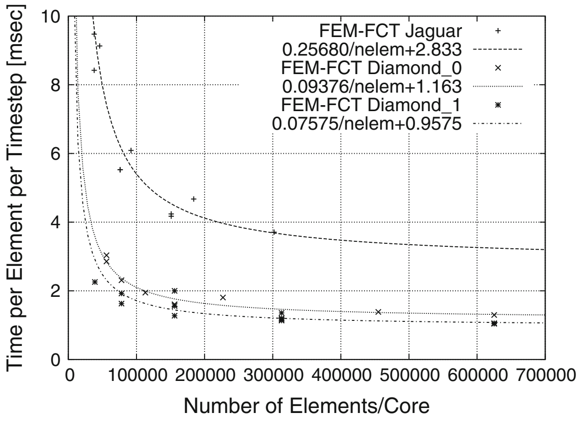


Fig. 3 Blast in room: Timings (relative)

After an extensive analysis [29] it was found that the communication times between processors were the reason for this poor performance. The sobering conclusion of this analysis is that, at present, the limiting speed of CFD solvers is given by the MPI communication network. And compared to raw processor speed and RAM transfer rates, the speed of the MPI communication network is advancing at a slower pace. If these trends continue, the 'useful mesh size' per processor will **increase**, not decrease (!). While these conclusions have been stated before, we predict that as more users get access to hundreds of thousands of cores and the domain size per core starts shrinking, they will also encounter the 'minimum timestep barrier' reported here.

Let us ponder the consequences of this barrier for the particular case of computational fluid dynamics (similar consequences can be drawn for other fields). The current trend here is to migrate from the quasi-steady Reynolds-Averaged Navier Stokes (RANS) description to the Large-Eddy Simulation (LES) of flows. This implies an increase in mesh size of 3–6 orders of magnitude, and an increase in the number of timesteps to achieve convergence / statistical data of 3–4 orders of magnitude. The increase in mesh size may be easily absorbed by machines with millions of cores. But the requirement of $O(10^7–10^8)$ timesteps, coupled with a minimum timestep barrier of (optimistically) $T_{MPI} = O(0.01)$ $[sec]$ means that no matter how large the machine, an LES run will take $T_{LES} = O(10^5–10^6)$ $[sec]$, i.e. **two weeks** under the best of circumstances. Such long running times will clearly hinder the useful range of applicability.

3 The Simulation Pipeline

Any scientific calculation based on the solution of partial differential equations follows the so-called simulation pipeline: domain definition, imposition of boundary conditions, mesh generation, solver, possible mesh adaptation, post-processing of results. To date, only the solvers have been migrated to systems with thousands (or, in some cases, millions) of cores. The current workflow considers a scalar, large memory machine with powerful graphics for the domain definition (CAD) and the imposition of boundary conditions. The input data is then used to generate a mesh on a large, shared-memory multicore machine (`ncore` < 64). The splitting of the domain into pieces is also accomplished on such machines. The domain files are then transferred to the massively parallel machine (`ncore>10,000`) and the solver is run. Finally, the results are assembled and post-processed on large-memory, scalar machines that are of the same class as those used for pre-processing. This 'scalar-pre, simple parallel solve, scalar-post' environment will no longer be possible once machines with more than 10^6 cores become widespread. The grid size alone will force parallelization of meshing. The same will happen to post-processing, as well as domain splitting and load balancing.

In the sequel, we show ways of obtaining parallel grid generation and dynamic load balancing for multiphysics.

4 Parallel Grid Generation

As stated before, many solvers have been ported to distributed parallel machines while grid generators have, in general, lagged behind. One can cite several reasons for this:

(a) For many applications the CPU requirements of grid generation are orders of magnitude less than those of field solvers, i.e. it does not matter if the user has to wait several hours for a grid;
(b) (Scalar) grid generators have achieved a high degree of maturity, generality and widespread use, leading to the usual inertia of workflow ('modus operandi') and aversion to change;
(c) In recent years, low-cost machines with few cores but very large memories have enabled the generation of large grids with existing (scalar) software; and
(d) In many cases it is possible to generate a mesh that is twice (2^d times) as coarse as the one desired for the simulation. This coarse mesh is then h-refined globally.

Parallel unstructured grid generation has been pursued since the early 1990s [1, 3–8, 10, 13, 14, 16, 18, 24, 27, 36, 37, 42, 43, 47, 53].

The two most common ways of generating unstructured grids are the Advancing Front Technique (AFT) [12, 17, 22–24, 32, 38–41] and the Generalized Delaunay Triangulation (GDT) [2–4, 7, 15, 34, 47, 50, 51]. The AFT introduces one element at a time, while the GDT introduces a new point at a time. Thus, both of these techniques are, in principle, scalar by nature, with a large variation in the number of operations required to introduce a new element or point. While coding and data structures may influence the scalar speed of the 'core' AFT or GDT, one often finds that for large-scale applications, the evaluation of the desired element size and shape in space, given by background grids, sources or other means [31] consumes the largest fraction of the total grid generation time. Furthermore, the time required for mesh improvements (and any unstructured grid generator needs them) is in many cases higher than the core AFT or GDT modules. Typical speeds for the complete generation of a mesh (surface, mesh, improvement) on current Intel Xeon chips with 3.2 GHz and sufficient memory are of the order of 0.5–2.0 Mels/min. Therefore, it would take approximately 2,000 minutes (i.e. 1.5 days) to generate a mesh of 10^9 elements, a common size in computational fluid dynamics and computational electromagnetics. Assuming perfect parallelization, this task could be performed in the order of a minute on 2,000 processors, clearly showing the need for parallel mesh generation.

The easiest form of achieving volume-based parallelism is by using a grid to define the regions to be meshed by each processor. Optimally, this domain-defining grid (DDG) should have the same surface triangulation as the desired, fine mesh, but could be significantly coarser in the interior. In this way, the definition of the domain to be gridded is unique, something that is notoriouly difficult to achieve by other means (such as background grids, bins or octrees). This domain-defining grid is then partitioned according to the estimated number of elements to be generated, allowing for a balanced distribution of work among the processors. The domain defining grid is also used to redistribute the elements and points after grid generation, and during the subsequent mesh improvement. A parallel grid generator based on these ideas has been developed over the last 3 years [26]. Figure 4a considers a typical example, taken from a blast simulation carried out for an office complex. Figure 4b–d show the trace of the domain defining grid partition on the surface as well as the fronts after the parallel grid generation passes using 64 domains (mpi processors) for a finer mesh. Table 1 gives a compilation of timings for different mesh sizes, domains and processors on different machines.

One may note that:

(a) Generating the 121 M mesh on one 8-core shared memory node (i.e. `nproc = 1, nprol = 8`) is slower than the distributed memory equivalent (i.e. `nproc = 8, nprol = 1`);
(b) The number of elements per core should exceed a minimum value (typically of the order of 2–4 Mels) in order to reach a generation speed per core that is acceptable;
(c) The local OMP scaling improves as the number of elements in each domain is increased;
(d) It only takes on the order of five minutes to generate a mesh of 121 Mels on 256 cores (`nproc = 32, npro = 8`), and on the order of 40 minutes to generate a mesh of 1,010 Mels on 512 cores (`nproc = 32, nprol = 8`).

These timings, and those of other large cases that required parallel mesh generation, show that the proposed approach is scalable and able to produce large grids of high quality in a modest amount of clocktime. With parallel grid generation entering production, a major impediment to a completely scalable simulation pipeline (grid generation, solvers, post-processing) has been removed, opening the way for truly large-scale computations using unstructured, body-fitted grids.

5 Load Balancing for Multiphysics

For field solvers, which are commonly used for computational fluid and solid mechanics as well as electromagnetics, the classic way to distribute work among many distributed memory processors is via domain decomposition. Given that the work requirements are proportional to the number of elements/points in a domain, the aim is to achieve subdomains of equal size while minimizing communication. As the communication between processors is proportional to the surface points of each subdomain, the aim is to minimize surface-to-volume ratios, which is achieved by keeping the domains as contiguous (non-split) and 'spherical' as possible. Techniques commonly used for domain splitting include the advancing front methods, coordinate- and moment- recursive bisection, and space-filling curve subdivisions [11, 19, 20, 33, 35, 44, 48, 52].

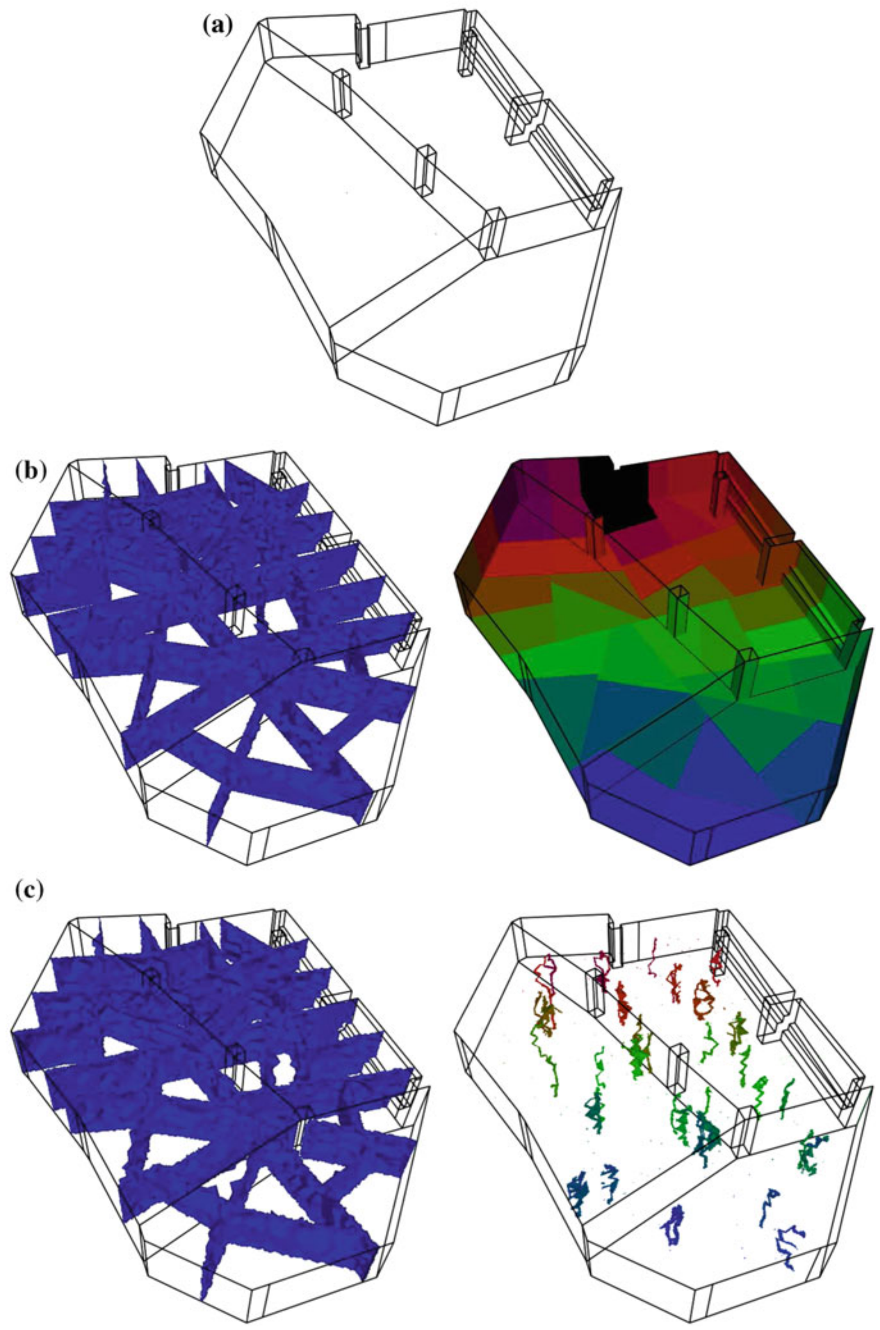


Fig. 4 Garage: **a** outline of geometry, **b**–**d** internal surface of DDG partition and remaining front after each pass

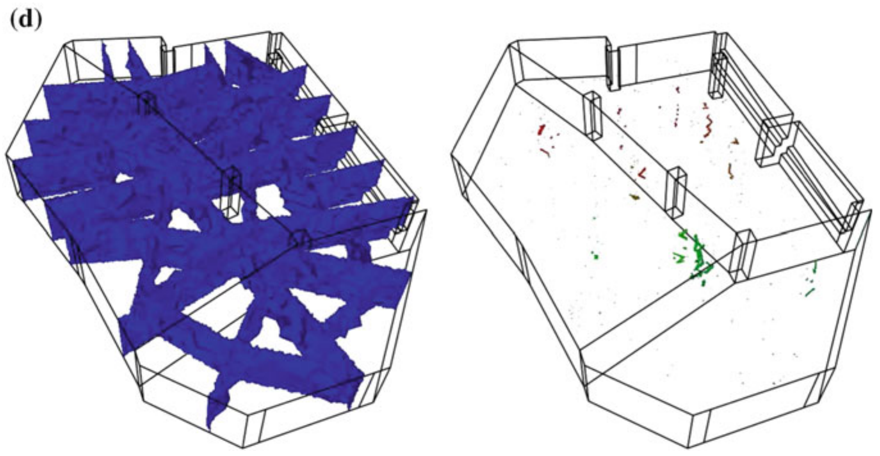


Fig. 4 (Continued)

Table 1 Garage

Machine	nproc	nprol	ncore	nelem	CPU [sec]	AbsSpeed [els/sec]	RelSpeed [els/sec/core]
Xeon(1)	1	8	8	120 M	2,293	52,333	6,542
SGI ITL	8	1	8	121 M	1,605	75,389	9,423
SGI ITL	8	8	64	121 M	516	234,496	3,664
Cry AMD	8	1	8	121 M	2,512	48,169	6,021
Cry AMD	16	1	16	121 M	1,954	61,924	3,870
Cry AMD	32	1	32	121 M	1,118	100,082	3,128
SGI ITL	16	1	16	121 M	1,048	115,458	7,216
SGI ITL	16	2	32	121 M	667	181,409	5,669
SGI ITL	16	4	64	121 M	407	297,297	4,645
SGI ITL	16	8	128	121 M	329	367,781	2,873
SGI ITL	32	1	32	121 M	646	187,306	5,853
SGI ITL	32	2	64	121 M	427	283,372	4,427
SGI ITL	32	4	128	121 M	346	349,710	2,732
SGI ITL	32	8	256	121 M	316	383,030	1,496
Cry AMD	64	1	64	972 M	6,048	160,714	2,511
SGI ITL	64	8	512	1010 M	2,504	403,354	788

With the possibility of computing larger problems, the desire to compute physics of ever increasing complexity also emerged. Some current flow applications include a traditional (i.e. unreacting) flow solver, chemical reactions, moving embedded or immersed bodies, particles, etc. A timestep for such an application may proceed as follows:

- Identify the (new) position of embedded/immersed bodies, obtaining the new boundary conditions/geometric parameters required;

- Advance the chemical reactions one timestep, obtaining the source-terms for the flow solver;
- Update the particles one timestep, obtaining the source-terms for the flow solver;
- Advance the flowfield one timestep.

The order of these operations is not mandatory and will vary among field solvers. What is important, though, is that at the end of each of these steps a synchronization among processors is required: the calculation can not proceed until all processors have completed each step in turn. Therefore, for optimal performance **the load should be balanced in each step**.

This inherent requirement of all multiphysics solvers implies that, compared to simple field solvers (e.g. just flow), the potential for load imbalances and suboptimal performance increases substantially.

In the sequel, we describe one possible way to achieve near-optimal load balance for multiphysics solvers. The key idea is to subdivide the global domain into more subdomains than processors. These 'oversampled' or 'oversplit' subdomains are then grouped together in an optimal way so as to achieve the best load balance possible.

5.1 Splitting Algorithm

Assume a multiphysics application with n_s sequential (physics) stages $s_i, i = 1, n_s$ whose work (CPU) requirements $w_i(e)$ per element can be quantified. The splitting algorithm for N_p processors (mpi domains) then proceeds as follows:

S1 Obtain the CPU requirement for each stage in each element: $w_i(e)$
S2 Obtain the sum (or max) of these CPU requirements for each element:
$w_t(e) = \sum_{i=1}^{n_s} w_i(e);\;\; w_t(e) = max(w_i(e)),\;\; i = 1, n_s$
S3 Using $w_t(e)$ and any of the usual domain splitting techniques, subdivide the domain into $m \cdot N_p$ sub-subdomains $\Omega_j,\;\; j = 1, m \cdot N_p$;
S4 Sum up the work for each of the stages s_i for each sub-subdomain Ω_j:
$W_i^j = \sum w_i(e),\;\; e \in \Omega_j$
S5 Obtain the (desired) average work for each stage s_i for a subdivision into the desired N_p processors/ subdomains:
$A_i = (1/N_p) \sum w_i(e)$
S6 Agglomerate the $m \cdot N_p$ sub-subdomains into N_p subdomains, attempting to exceed the desired averages A_i as little as possible.

While Steps S.1–S.5 are intuitive, Step S.6 requires a more careful consideration. The following multipass strategy seems to give acceptable results:

H1 Initialize a 'used domain marker' for the sub-subdomains $U^j = 0,\;\; j = 1, m \cdot N_p$
H2 Initialize total work counts for each of the desired N_p subdomains:
$V_i^k = 0,\;\; i = 1, n_s,\;\; k = 1, N_p$
H3 Loop over the increases in allowed average work: A_i

```
H4 Put the unassigned sub-subdomains ($U^j = 0$) into a heap-list according to the
   work counts $W_i^j \;\; i = 1, n_s \;\; j = 1, m \cdot N_p$;
H5 Retrieve the next sub-subdomain $j$ from the top of the heap list;
H6 If: $U^j > 0$ (the sub-subdomain $j$ has been used): skip                (Goto H5)
H7 Loop over the desired subdomains $k = 1, N_p$
   If: $V_i^k = 0$ (the subdomain has not been initialized):
   - Add: $V_i^k = V_i^k + W_i^j$
   - Mark: $U^j = k$
   - Exit                                                                     (Goto H5)
   Else:
   - If: $A_i < V_i^k + W_i^j \;\;,\;\; i = 1, n_s$: skip                      (Goto H5)
   - Add: $V_i^k = V_i^k + W_i^j$
   - Mark: $U^j = k$
   - Exit                                                                     (Goto H5)
   Endif
H8 If: items remain in the heap list:                                         (Goto H5)
H9 If there are unassigned sub-subdomains ($U^j = 0$):
   - Increase the allowed average work: $A_i$
   - Allocate the remaining sub-subdomains                                    (Goto H3)
```

While this basic technique will balance the work properly, it does not attempt to minimize the resulting surface-to-volume ratios. This means that many unconnected sub-subdomains may end up belonging to a subdomain/processor. One can alleviate this shortcoming by improving the selection criterion for the allocation of sub-subdomains to domains in Step H7 above. The key idea is to favour those subdomains k that satisfy the condition $A_i > V_i^k + W_i^j, \;\; i = 1, n_s$ and are as close as possible to (preferably overlap) the sub-subdomain j being retrieved from the heap-list. This may be implemented as follows:

```
- H70: Initialize closest/best/optimal subdomain and overlap distance: $k_{opt} = 0$,
  $d_{opt} = 0$
- H71: Initialize uninitialized domain marker: $k_0 = N_p + 1$
- H72: Loop over the desired subdomains $k = 1, N_p$
  If: $V_i^k = 0$ (the subdomain has not been initialized):
  - Set : $k_0 = min(k_0, k)$
  Else:
  - If: $A_i > V_i^k + W_i^j \;\;,\;\; i = 1, n_s$:
  - Compare overlap zone (update $k_{opt}, d_{opt}$)
  - Endif
  Endif
- H73: If: $k_{opt} > 0$ (i.e. a 'best' subdomain has been found):
  - Add: $V_i^k = V_i^k + W_i^j$
  - Mark: $U^j = k$
  - Exit                                                                      (Goto H5)
  Endif
```

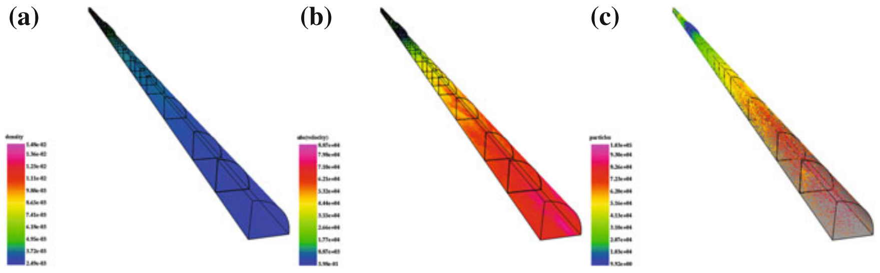


Fig. 5 Blast in tube with particles

```
- H74: If: $k_0 > 0$ (i.e. an uninitialized subdomain has been found):
    - Add: $V_i^k = V_i^k + W_i^j$
    - Mark: $U^j = k$
    - Exit                                                    (Goto H5)
  Endif
```

In order to estimate the 'closeness' or 'overlap' of the different sub-subdomains, a simple bounding box approach has been used.

5.2 Example: Detonation in Tube with Particles

This example considers a relatively long tube where a detonation occurs. As the blast wave reaches the walls of the tube, particles are introduced into the flowfield. The number of elements is of $O(5 \cdot 10^7)$, while the number of particles eventually reaches $O(2 \cdot 10^6)$. The problem was run with 16 distributed memory (mpi) processes/domains, and 8 shared memory cores (OpenMP) per domain, i.e. a total of 256 cores. For the present purpose, the problem was run for 1,000 steps, with a re-split using the method described above every 100 steps. The splitting obtained at the end of the 1,000 steps may be discerned in Fig. 5a–c. Note that there are many more domains than mpi-processes/domains, but that, as expected, the basic moment-based recursive bisection has still produced 'slices' along the tube. The breakdown of times is approximately as follows: flow solver 60 %, particle update 30 %, repartitioning and renumbering 10 %. This implies that the parallel repartioning does not lead to an excessive increase in CPU requirements while allowing for a much better load balance.

6 Conclusions and Outlook

The present paper has summarized trends in supercomputing and the consequences they will have on coupled problems in computational mechanics. In particular, the 'red-shift' observed in the performance increase of hardware subcomponents implies that heavy emphasis should be placed on field solvers that minimize the access to memory per timestep update.

The trend towards parallel machines with more than 10^6 cores implies that the prevalent 'scalar-pre, simple parallel solve, scalar-post' environment will have to give way to a completely scalable simulation pipeline. The grid size alone will force distributed parallelization of meshing, domain splitting, load balancing and post-processing.

Possible ways of addressing parallel meshing and dynamic load balancing for multiphysics we treated. The examples shown indicate that these approaches have to potential to be core building blocks of a completely scalable simulation pipeline.

Much remains to be done in this field. It involves a lot of coding and debugging. It is perhaps not as refined as the development of high order methods or new turbulence models. But without it, further advances in coupled, large-scale problems will not occur.

Acknowledgments Over the course of the last 25 years CIMNE has become an internationally recognized center where, at any given time, one can find students, post-docs, research scientists, professors and visitors from all over the world working on many topics. For the last 20 years, my family and I (RL) have had the immense fortune of being able to visit CIMNE and Barcelona every summer. I have always reserved the scientific topics that time did not permit me to consider during the academic year for those wonderful weeks at CIMNE, which I consider among the most productive and happy of my life. Therefore, I thank all the many members of CIMNE whom I had the privilege to work with, and in particular Prof. Oñate for his friendship, contagious enthusiasm and vision.

References

1. Andrae H, Ivanov E, Gluchshenko O, Kudryavtsev A (2008) Automatic parallel generation of tetrahedral grids by using a domain decomposition approach. J Comp Math Math Phys 48(8):1448–1457
2. Baker TJ (1989) Developments and trends in three-dimensional mesh generation. Appl Num Math 5:275–304
3. Blelloch GE, Hardwick JC, Miller GL, Talmor D (1999) Design and implementation of a practical parallel delaunay algorithm. Algorithmica 24:243–269
4. Chew LP, Chrisochoides N, Sukup F (1997) Parallel constrained delaunay meshing. In: Proceedings of 1997 workshop on trends in unstructured mesh generation, Northwestern University, Evanston June (1997)
5. Chrisochoides N (2005) Parallel mesh generation. In: Bruaset AM, Tveito A (eds) Numerical solution of partial differential equations on parallel computers. Springer, Berlin, pp 237–259
6. Chrisochoides N, Nave D (1999) Simultaneous mesh generation and partitioning for Delaunay meshes. In: Proceedings of 8th international meshing roundtable, South Lake Tahoe, pp 55–66

7. Chrisochoides N, Nave D (2003) Parallel Delaunay mesh generation kernel. Int J Num Meth Eng 58:161–176
8. de Cougny HL, Shephard MS, Ozturan C (1994) Parallel three-dimensional mesh generation. Comput Syst Eng 5:311–323
9. de Cougny H, Shephard M (1999) Parallel volume meshing using face removals and hierarchical repartitioning. Comp Meth Appl Mech Eng 174(3–4):275–298
10. de Cougny HL, Shephard MS, Ozturan C (1995) Parallel three-dimensional mesh generation on distributed memory MIMD computers. Tech Rep SCOREC Rep # 7, Rensselaer Polytechnic Institute
11. Flower J, Otto S, Salama M (1990) Optimal mapping of irregular finite element domains to parallel processors. 239–250
12. Frykestig J (1994) Advancing front mesh generation techniques with application to the finite element method. Pub. 94:10, Chalmers University of Technology, Göteborg
13. Galtier J, George PL (1997) Prepartitioning as a way to mesh subdomains in parallel. In: Special symposium on trends in unstructured mesh generation (ASME/ASCE/SES), pp 107–122
14. George PL (1999) Tet meshing: construction, optimization and adaptation. In: Proceedings of 8th international meshing roundtable, South Lake Tahoe, October 1999
15. George PL, Hecht F, Saltel E (1991) Automatic mesh generator with specified boundary. Comp Meth Appl Mech Eng 92:269–288
16. Ivanov EG, Andrae H, Kudryavtsev AN (2006) Domain decomposition approach for automatic parallel generation of tetrahedral grids. Int Math J Comp Meth App Math 6(2):178–193
17. Jin H, Tanner RI (1993) Generation of unstructured tetrahedral meshes by the advancing front technique. Int J Num Meth Eng 36:1805–1823
18. Kadow C, Walkington N (2003) Design of a projection-based parallel Delaunay mesh generation and refinement algorithm. In: Proceedings of fourth symposium on trends in unstructured mesh generation, Albuquerque, 2003
19. Karypis G, Kumar V (1999) Parallel multilevel k-way partitioning scheme for irregular graphs. SIAM Rev 41(2):278–300
20. Karypis G, Kumar V (1998) A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. J Parallel Distrib Comput 48:71–85
21. Ko S-H, Kim N, Kim J, Thota A, Jha S (2010) Efficient runtime environment for coupled multiphysics simulations: dynamic resource allocation and load-balancing. In: Procedings of 10th IEEE/ACM international conference on cluster, cloud and grid computing (CCGrid), 17–20 May (2010)
22. Löhner R (1988) Some useful data structures for the generation of unstructured grids. Comm Appl Num Meth 4:123–135
23. Löhner R (1996) Extensions and improvements of the advancing front grid generation technique. Comm Num Meth Eng 12:683–702
24. Löhner R (2001) A parallel advancing front grid generation scheme. Int J Num Meth Eng 51:663–678
25. Löhner R (2008) Applied CFD techniques, 2nd edn. Wiley, Chichester
26. Löhner R (2013) A 2nd generation parallel advancing front grid generator. AIAA-2013-0147
27. Löhner R, Camberos J, Merriam M (1992) Parallel unstructured grid generation. Comp Meth Appl Mech Eng 95:343–357
28. Löhner R, Baum JD (2012) 40 years of FCT: status and directions. In: Kuzmin D, Löhner R, Turek S (eds) Flux-corrected transport, 2nd edn. Springer, New York, pp 119–143
29. Löhner R, Baum JD (2014) On maximum achievable speeds for field solvers. Int J Num Meth Heat Fluid Flow (to appear)
30. Löhner R, Corrigan A, Wichmann K-R, Wall W (2013) On the achievable speeds of finite difference solvers on CPUs and GPUs. AIAA-2013-2852
31. Löhner R, Luo H, Baum JD, Rice D (2008) Improvements in speed for explicit, transient compressible flow solvers. Int J Num Meth Fluids 56(12):2229–2244
32. Löhner R, Parikh P (1988) Three-dimensional grid generation by the advancing front method. Int J Num Meth Fluids 8:1135–1149

33. Löhner R, Ramamurti R, Martin D (1993) A parallelizable load balancing algorithm. AIAA-93-0061
34. Marcum DL, Weatherill NP (1995) Unstructured grid generation using iterative point insertion and local reconnection. AIAA J 33(9):1619–1625
35. Mehrota P, Saltz J, Voigt R (eds) (1992) Unstructured scientific computation on scalable multiprocessors. MIT Press, Cambridge
36. Okusanya T, Peraire J (1996) Parallel unstructured mesh generation. In: Proceedings of 5th internatinal conference number grid generation in CFD and related fields, Mississippi, April 1996
37. Okusanya T, Peraire J (1997) 3-D parallel unstructured mesh generation. In: Proceedings of joint ASME/ASCE/SES summer meeting 1997
38. Peraire J, Morgan K, Peiro J (1990) Unstructured finite element mesh generation and adaptive procedures for CFD. AGARD-CP-464, p 18
39. Peraire J, Morgan K, Peiro J (1992) Adaptive remeshing in 3-D. J Comp Phys 103(2):269–285
40. Peraire J, Peiro J, Formaggia L, Morgan K, Zienkiewicz OC (1988) Finite element Euler calculations in three dimensions. Int J Num Meth Eng 26:2135–2159
41. Peraire J, Vahdati M, Morgan K, Zienkiewicz OC (1987) Adaptive remeshing for compressible flow computations. J Comp Phys 72:449–466
42. Said R, Weatherill NP, Morgan K, Verhoeven NA (1999) Distributed parallel delaunay mesh generation. Comp Meth Appl Mech Eng 177:109–125
43. Shostko A, Löhner R (1995) Three-dimensional parallel unstructured grid generation. Int J Num Meth Eng 38:905–925
44. Simon H (1991) Partitioning of unstructured problems for parallel processing. NASA Ames Tech Rep RNR-91-008
45. Stück A, Camelli F, Löhner R (2010) Adjoint-based design of shock mitigation devices. Int J Num Meth Fluids 64:443–472
46. Togashi F, Baum JD, Mestreau E, Löhner R, Sunshine D (2010) Numerical simulation of long-duration blast wave evolution in confined facilities. Shock Waves 20(5):409–424
47. Tremel U, Sorensen KA, Hitzel S, Rieger H, Hassan O, Weatherill NP (2006) Parallel remeshing of unstructured volume grids for CFD applications. Int J Num Meth Fluids 53(8):1361–1379
48. Vidwans A, Kallinderis Y, Venkatakrishnan V (1993) A parallel load balancing algorithm for 3-D adaptive unstructured grids. AIAA-93-3313-CP
49. Walshaw C, Cross M (2000) Parallel optimisation algorithms for multi-level mesh partitioning. Parallel Comput 26:1635–1660
50. Weatherill NP, Hassan O (1994) Efficient three-dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints. Int J Num Meth Eng 37:2005–2039
51. Weatherill NP (1992) Delaunay triangulation in computational fluid dynamics. Comp Math Appl 24(5/6):129–150
52. Williams D (1990) Performance of dynamic load balancing algorithms for unstructured grid calculations. CalTech Rep C3P913
53. Yoshimura S, Nitta H, Yagawa G, Akiba H (1998) Parallel automatic mesh generation method of ten-million nodes problem using fuzzy knowledge processing and computational geometry. In: Proceedings of 4th world cong comp mech Buenos Aires, Argentina, July 1998

Partitioned Solution of Coupled Stochastic Problems

Mohammad Hadigol, Alireza Doostan, Hermann G. Matthies and Rainer Niekamp

Abstract This work is concerned with the propagation of uncertainty across coupled problems with high-dimensional random inputs. A stochastic model reduction approach based on low-rank separated representations is proposed for the partitioned treatment of the uncertainty space. The construction of the coupled solution is achieved though a sequence of approximations with respect to the dimensionality of the random inputs associated with each individual subproblem and not the combined dimensionality, hence drastically reducing the overall computational cost. The coupling between the sub-domain solutions is done via the classical Finite Element Tearing and Interconnecting (FETI) method, thus providing a well suited framework for parallel computing. A high-dimensional stochastic problem, a coupled 2D elliptic PDE with random diffusion coefficient, has been considered in this paper to study the performance and accuracy of the proposed stochastic coupling approach.

1 Introduction

For coupled problems a partitioned solution procedure, which allows the re-use of the subproblem solvers and the accompanying software, has a long history and well-developed procedures, see e.g. [18] and the references therein. Here we combine this idea with that of uncertainty propagation, which is steadily gaining in importance in order to obtain realistic predictions of the effects of such uncertainties and quantify their impact on Quantities of Interest (QoI). Uncertainty quantification (UQ), an emerging field in computational engineering and science, is concerned with the development of rigorous and efficient solutions to this exercise. It is quite common

M. Hadigol · Alireza Doostan
University of Colorado, CO 80309, Boulder, USA

H. G. Matthies (✉) · R. Niekamp
TU Braunschweig, 38092 Braunschweig, Germany
e-mail: wire@tu-bs.de

to model such uncertainties probabilistically, where uncertain parameters are represented by random variables (RVs), fields, or processes. Several techniques have been developed to study the propagation of such uncertainties in engineering systems, e.g. see [1, 2, 14, 16, 17, 20, 25, 26] and the references therein. Among these techniques, stochastic spectral methods based on polynomial chaos expansions (PCE) have received special attention due to their advantages over traditional UQ techniques such as perturbation-based and Monte Carlo sampling (MCS) methods, as these schemes may converge much faster than MCS methods [14]. These spectral or functional approximation methods are based on expanding the solution of interest as functions of *known* RVs. The coefficients of these expansions are then computed, for instance, via Galerkin projection, referred to as stochastic Galerkin (SG), or pseudo-spectral collocation or stochastic collocation (SC), see e.g. [21, 27].

Already in the treatment of a single system, the high-dimensional problems which result from PC-based techniques pose computational challenges. The problem is even compounded when dealing with coupled stochastic systems. This may be a common situation when one is dealing with UQ of engineering problems involving, for instance, coupled domains or scales with independent sources of uncertainty. In the past few years, several alternative methods have been proposed to address the the cost associated to the high-dimensionality of standard PC methods, e.g. [3, 5, 9, 15, 21, 27, 28]. Here we want to extend these methods for the analysis of coupled systems, but through a partitioned approach which hopefully will mitigate the cost of coupling, following [12].

In the cases of coupled UQ problems, the solution depends on all random inputs; hence has even more coupling, and a direct application of PC expansion techniques looks at first glance daunting. While integration of PC expansions with standard domain decomposition (DD) techniques may partially reduce the overall computational complexity by partitioning the physical space, e.g. see [10, 24], expansions (or sampling) with respect to the combined set of random inputs is still required. Instead, we propose an approach that additionally enables a *partitioned treatment* of the stochastic space; that is, the solution is computed through a sequence of approximations with respect to the random inputs associated with each individual sub-domain, and not the combined set of random inputs. To this end, we propose a stochastic expansion based on the so-called *low-rank* or *separated* representations and demonstrate how it can be obtained.

Model reduction techniques based on separated or low-rank representations, one of which is a.k.a. canonical decomposition, of high-dimensional stochastic functions have been recently proposed, see e.g. [4, 6, 13, 22], where in some instances the low-rank approach may generate *additional* coupling in an otherwise uncoupled (in the stochastic variables) collocation approach. We here adopt a special form of separated representations for the stochastic computation of coupled domain problems.

The plan for this chapter is as follows: In Sect. 2 we recall basic partitioned algorithms exemplified on an abstract coupled system, which may depend on parameters. The case when these parameters are RVs is addressed in Sect. 3. In the following Sect. 4 we bring these two strands of computational techniques together to develop a partitioned solution algorithm to compute a separated low-rank approximation. A computational example is shown in Sect. 5, and concluding remarks are offered in Sect. 6.

2 Coupled Problems

A coupled stochastic system is a special case of coupled parametric systems, where the parameters are random variables. For completeness, it is worthwhile to recall how a partitioned solution strategy works for a coupled system when these parameters have a fixed value, which we concentrate on first.

2.1 One Single System

To introduce the problem and notation, where we follow [11], first look at a single system which will be denoted abstractly as

$$\frac{\partial}{\partial t}u(t) + A(q; u(t)) = f(q; t), \tag{1}$$

where $u(t) \in \mathcal{U}$ describes the state of the system at time $t \in [0, T]$ lying in a Hilbert space $\mathcal{U}$ (for the sake of simplicity), A is a—possibly non-linear—operator modelling the physics of the system, and $f \in \mathcal{U}^*$ is some external influence (action / excitation / loading). The model depends on some parameters $q \in \mathcal{Q}$ which are uncertain. For our purposes here it will be sufficient to look at the stationary case when $\partial u/\partial t = 0$ and $\partial f/\partial t = 0$:

$$A(q; u) = f(q). \tag{2}$$

Often an example such as Eq. (2) is the stationary condition of some functional or potential Π on $\mathcal{U}$, i.e. it is equivalent to

$$\delta_u \Pi(q; u) = A(q; u) - f(q) = 0. \tag{3}$$

Later we will assume such a situation for our numerical example.

To have a concrete example of Eq. (1), consider the diffusion equation

$$\frac{\partial}{\partial t}u(x, t) - \operatorname{div}(\kappa(x) \cdot \nabla u(x, t)) = f(x, t), \quad x \in \mathcal{G}, \tag{4}$$

with appropriate boundary and initial conditions, where $\mathcal{G} \subset \mathbb{R}^n$ is a suitable domain. The diffusing quantity is $u(x, t)$ (heat, concentration) and the term $f(x, t)$ models sinks and sources. The parameter q in question could for example be the diffusion tensor κ, or the initial conditions $u(x, 0)$. The stationary case of Eq. (4) is well-known to be the gradient of

$$\Pi(u) = \frac{1}{2} \int_{\mathcal{G}} \nabla u(x) \cdot \kappa(x) \cdot \nabla u(x)\, \mathrm{d}x - \int_{\mathcal{G}} u(x) f(x)\, \mathrm{d}x, \tag{5}$$

where we have assumed homogeneous Dirichlet boundary conditions in Eq. (5) for simplicity. Later in the numerical example two such systems will be coupled at the common part of the boundary.

Focusing on the stationary case Eq. (2), assume that we are also given an iterative solver—convergent for all fixed values of q—which generates successive iterates for $k = 0, \ldots$ converging to the solution $u^*(q)$.

$$u^{(k+1)}(q) = S(q; u^{(k)}(q), R(q; u^{(k)}(q)), \quad \text{with } u^{(k)}(q) \to u^*(q), \tag{6}$$

where S is one cycle of the solver which may also depend on the iteration counter k, $u^{(q)}$ is some starting vector, and $R(q; u^{(k)}(q)) := f(q) - A(q; u^{(k)})$ is the residuum of Eq. (1). Obviously, when the residuum vanishes—$R(q; u^*(q)) = 0$—the mapping S has a fixed point $u^*(q) = S(q; u^*(q), 0)$.

2.2 Partitioned Solution for Coupled Systems

We now turn to coupled systems, where we follow [18], and for the sake of simplicity, we look at only two systems—again in abstract form—which are coupled:

$$A_I(q_I; u_I, u_{II}, \lambda) = f_I(q_I), \qquad A_{II}(q_{II}; u_{II}, u_I, \lambda) = f_{II}(q_{II}), \tag{7}$$

where u_I, u_{II} and q_I, q_{II} are the state and parameters of system I or II. Later again we will assume that the two equations in Eq. (7) are the partial derivatives of some functionals $\Pi_I(q_I; u_I, u_{II})$ and $\Pi_{II}(q_{II}; u_{II}, u_I)$. The Lagrange multiplier of the coupling is λ, and the coupling condition is

$$C(q_I, q_{II}; u_I, u_{II}) = 0, \quad \text{or rather} \quad \forall\lambda : \langle C(q_I, q_{II}; u_I, u_{II}), \lambda\rangle_\Lambda = 0. \tag{8}$$

Often the system I in Eq. (7) does not depend on u_{II}, and vice versa, which may make the coupling a bit 'looser'. It makes no difference for what is to follow.

A partitioned solution procedure assumes that we have solvers separately for the two equations in Eq. (7); more precisely that in the first equation, for u_{II} and λ, q_I, q_{II} fixed, that equation can be solved by a given procedure—iterating a map like in Eq. (6) to convergence $u_I^{k+1} = S_I(q_I; f_I, u_I^k, u_{II}, \lambda)$, and vice versa for the second equation. Additionally we assume that for fixed q_I, q_{II} and u_I, u_{II}, the coupling Eq. (8) determines λ, again produced by an iterator

$\lambda^{k+1} = \Lambda(q_I, q_{II}; u_I, u_{II}, \lambda^k)$. Out of these building blocks partitioned solution procedures can be constructed, see [18] and the references therein. The simplest of

Algorithm 1 Nonlinear block-Jacobi iteration

```
for $k = 0, 1, \dots$ to convergence do
    $u_I^{k+1} := S_I(q_I; f_I, u_I^k, u_{II}^k, \lambda^k)$;
    $u_{II}^{k+1} := S_{II}(q_{II}; f_{II}, u_I^k, u_{II}^k, \lambda^k)$;
    $\lambda^{k+1} := \Lambda(q_I, q_{II}; u_I^k, u_{II}^k, \lambda^k)$;
end for
```

these—and hence sufficient to explain the principal idea—is a block-Jacobi iteration, shown in Algorithm 1. The three statements in the loop in Algorithm 1 may be executed concurrently or in parallel. But often the algorithm is executed serially. Then—because it involves no further cost per step and is frequently much faster—one usually switches to the block-Gauss-Seidel variant where results are used as soon as they become available, e.g. u_I^{k+1} from the first statement will be used in the second instead of u_I^k, resulting in a block-Gauss-Seidel algorithm.

We look at the simplest instance, namely two linear symmetric systems defined by stiffness matrices on a finite dimensional space, i.e. a version of Eq. (7) after a possible discretisation:

$$\begin{bmatrix} K_I(q_I) & 0 & C_I^T \\ 0 & K_{II}(q_{II}) & -C_{II}^T \\ C_I & -C_{II} & 0 \end{bmatrix} \begin{bmatrix} u_I \\ u_{II} \\ \lambda \end{bmatrix} = \begin{bmatrix} f_I(q_I) \\ f_{II}(q_{II}) \\ 0 \end{bmatrix}. \tag{9}$$

Due to the symmetry, the equations Eq. (9) are the stationarity conditions for the deterministic Lagrangian in Eq. (10), namely

$$\delta_{u_I}\Pi = 0,\ \delta_{u_{II}}\Pi = 0,\ \delta_\lambda \Pi = 0,$$

a saddle-point for *fixed* (q_I, q_{II}):

$$\Pi(q_I, q_{II}; u_I, u_{II}, \lambda) = \Pi_I(q_I; u_I) + \Pi_{II}(q_{II}; u_{II}) + \lambda^T(C_I u_I - C_{II} u_{II}) := \\ \left(\frac{1}{2} u_I^T K_I(q_I) u_I - u_I^T f_I(q_I)\right) + \left(\frac{1}{2} u_{II}^T K_{II}(q_{II}) u_{II} - u_{II}^T f_{II}(q_{II})\right) + \lambda^T(C_I u_I - C_{II} u_{II}). \tag{10}$$

A block-Gauss-Seidel type algorithm for solving Eq. (9) is the FETI method, which is used here; detailed descriptions may be found at e.g. [7, 8, 23]. For nonlinear systems it may be worthwhile to use more advanced methods than block-Gauss-Seidel, e.g quasi Newton methods; see [18] for an analysis of such cases in particular for fluid-structure interaction. This finishes our brief review on the partitioned solution of coupled systems.

3 Stochastic Problems

Consider the stationary version of Eq. (1) shown in Eq. (2), where one now is interested in capturing the dependence on q. To make it clear that we regard both Eqs. (1) and (2) as equations in $\mathcal{U}^*$, we also denote the equivalent weak form in Eq. (11):

$$\forall v \in \mathcal{U}: \quad \langle A(u; q), v\rangle = \langle f(q), v\rangle. \tag{11}$$

In the sequel of this section, we follow the presentation in [19].

3.1 Parametric Dependence

We assume that q may be *uncertain*, and is modelled as a $\mathcal{Q}$-valued random variable (RVs), hence the state $u(q, f)$, which is a function of q and f, is a $\mathcal{U}$-valued RV. Uncertainty quantification (UQ) is the propagation of uncertainty in q and f to a corresponding one in u. We therefore take $(\Omega, \mathfrak{A}, \mathbb{P})$, a probability space with expectation operator $\mathbb{E}(\cdot)$, such that Ω is the set of all possible *realisations*, $\mathfrak{A}$ is a σ-algebra of measurable events (subsets of Ω), and $\mathbb{P}$ is a probability measure defined on $\mathfrak{A}$.

Assume for simplicity that $q(\omega)$ has finite variance, i.e. $\|q(\omega)\|_{\mathcal{Q}} \in \mathcal{S} := L_2(\Omega)$—with inner product of two random variables (RVs) $r_1, r_2 \in \mathcal{S}$ defined as usual as $\langle r_1, r_2\rangle_{\mathcal{S}} := \mathbb{E}(r_1 r_2) = \int_\Omega r_1(\omega) r_2(\omega)\, \mathbb{P}(\mathrm{d}\omega)$—so that

$$q \in L_2(\Omega, \mathcal{Q}) \cong \mathcal{Q} \otimes L_2(\Omega) = \mathcal{Q} \otimes \mathcal{S} =: \mathscr{Q},$$

where we regard the tensor product—here and later—as completion in the norm induced by the inner product on $\mathscr{Q}$ by $\langle\langle q_1 \otimes r_1, q_2 \otimes r_2\rangle\rangle_{\mathscr{Q}} := \langle q_1, q_2\rangle_{\mathcal{Q}} \langle r_1, r_2\rangle_{\mathcal{S}}$ and extended by linearity. The system model is now

$$A(q(\omega); u(\omega)) = f(q(\omega)) \qquad \text{a.s. in } \omega \in \Omega, \tag{12}$$

and the state $u = u(\omega)$ becomes a $\mathcal{U}$-valued random variable (RV), an element of the tensor space $\mathscr{U} := \mathcal{U} \otimes \mathcal{S}$. We may write this also as a variational statement, cf. Eq. (11), which facilitates both theory and numerical approximation, e.g. [1, 2, 16, 17]

$$\forall v \in \mathscr{U}: \quad \langle\langle A(q; u), v\rangle\rangle = \langle\langle f, v\rangle\rangle \quad (= \mathbb{E}(\langle f, v\rangle_{\mathcal{U}})), \tag{13}$$

so that under certain assumptions one obtains a theory for well-posed stochastic partial differential equations (SPDEs). As the input data q, right hand side f, and solution u are elements of tensor product spaces, this points to the later use of low-rank tensor approximations for efficient approximation algorithms, which will be crucial for the separated representation. In case that Eq. (2) is a gradient as in Eq. (3), this will carry over to Eq. (13), which is equivalent to

$$\forall v \in \mathscr{U} : \quad \langle\!\langle \delta_u \Pi(q, u), v \rangle\!\rangle = \langle\!\langle A(q; u), v \rangle\!\rangle - \langle\!\langle f, v \rangle\!\rangle = 0, \tag{14}$$

where the functional / potential on $\mathscr{U} = \mathcal{U} \otimes \mathcal{S}$ is

$$\Pi(q, u) := \mathbb{E}\,(\Pi(q; u)). \tag{15}$$

3.2 Stochastic Approximations

Assume that the operator equation $A(q; u) = f(q)$ has already been discretised by your favourite method—e.g. FEM or FVM or similar. This is essentially the choice of a finite-dimensional subspace $\mathcal{U}_N = \mathrm{span}\{\boldsymbol{v}_n\}_{n=1}^N \subset \mathcal{U}$, with $u \approx \sum_n u_n(q)\, \boldsymbol{v}_n \in \mathcal{U}_N$.

More importantly, a discretisation of q and $u(q)$ is needed. Stochastic processes and random parameter fields usually need *infinitely* many RVs $\{\xi_1(\omega), \xi_2(\omega), \ldots\}$, so that $q = q(\xi_1(\omega), \xi_2(\omega), \ldots) \in \mathscr{Q}$, where the $\xi_m \in \mathcal{S}$ are *known* RVs. A similar representation may be obtained for $f(q)$ [19]. Hence the solution is also a function of the ξ_m: $u(\xi_1, \xi_2, \ldots) = \sum_n u_n(\xi_1, \xi_2, \ldots)\boldsymbol{v}_n$. We discretise further by truncation to a *finite* number of RVs: $\boldsymbol{\xi}(\omega) = [\xi_1(\omega), \ldots \xi_M(\omega)] \in \mathbb{R}^M$, so that $q = q(\boldsymbol{\xi}) = q(\boldsymbol{\xi}(\omega))$ and

$$u = u(\boldsymbol{\xi}) = u(\boldsymbol{\xi}(\omega)) \approx \sum_n u_n(\boldsymbol{\xi})\, \boldsymbol{v}_n. \tag{16}$$

Based on this, for actual numerical computations, a representation of the stochastic aspect has to be chosen. Some frequently explored possibilities are computing the resulting *distributions* through *Fokker-Planck* type equations, establishing relations for the *moments* of the results, representing the solution through *samples*—the well known *(quasi) Monte Carlo* methods—and *functional* or *spectral approximation* where the solution is represented as a function of *known* RVs. Here we want to employ low-rank methods or separated representations, and these fall in the last (two) group(s). We chose functions (i.e. *known* RVs) $\mathrm{span}\{X_\beta\}_{\beta=1}^B = \mathcal{S}_B \subset \mathcal{S}$, and make the 'ansatz' in Eq. (16):

$$u_n(\boldsymbol{\xi}) \approx \sum_\beta u_n^\beta X_\beta(\omega), \tag{17}$$

hence giving the state u as a high-dimensional function

$$u(\boldsymbol{\xi}) = \sum_{n,\beta} u_n^\beta\, X_\beta(\boldsymbol{\xi})\boldsymbol{v}_n = \sum_{n,\beta} u_n^\beta\, \boldsymbol{v}_n \otimes X_\beta \in \mathcal{U}_N \otimes \mathcal{S}_B \subset \mathcal{U} \otimes \mathcal{S}. \tag{18}$$

If we take this *ansatz* Eq. (18) and insert it into Eq. (13), the residuum $R(q(\boldsymbol{\xi}); u(\boldsymbol{\xi}))$ will usually not vanish for all $\boldsymbol{\xi}$, as the finite set of functions $\{X_\beta\}$ can not match all possible parametric variations of $u(\boldsymbol{\xi})$. To determine the coefficients u_n^β, one may choose another set of functions (known RVs) $\Xi_\alpha(\boldsymbol{\xi}) \in \mathcal{S}$ for projection, so that the *weighted residual* vanishes:

$$\forall\alpha, k: \quad \langle\!\langle \boldsymbol{v}_k \otimes \Xi_\alpha, R(q(\boldsymbol{\xi}); u(\boldsymbol{\xi}))\rangle\!\rangle = \langle\!\langle \boldsymbol{v}_k \otimes \Xi_\alpha, f(\boldsymbol{\xi}) - A(q(\boldsymbol{\xi}); \sum_{n,\beta} u_n^\beta X_\beta(\boldsymbol{\xi})\boldsymbol{v}_n)\rangle\!\rangle = 0, \tag{19}$$

yielding a generally coupled system of equations of size $N \times B$ for the $\mathbf{u} := \{u_n^\beta\}$, the *tensor* coefficients representing the solution $u(\boldsymbol{\xi})$.

This general *Galerkin* method—also called the method of weighted residuals—usually comes in the flavours of a *Bubnov-Galerkin* method where $\Xi_\alpha = X_\alpha$ and the system of equations is *coupled* for all u_n^β, or as a *Petrov-Galerkin* method with $\Xi_\alpha \neq X_\alpha$. In the latter case, a frequent choice is $\Xi_\alpha(\omega) = \delta(\omega - \omega_\alpha)$, i.e. collocation / interpolation at the points ω_α, where $\delta(\omega - \omega_\alpha)$ is the *Dirac*-δ at ω_α. By additionally ensuring the *Kronecker*-δ property $X_\beta(\omega_\alpha) = \delta_{\beta,\alpha}$, one obtains the quasi-deterministic *uncoupled* collocation conditions

$$\forall\alpha, k: \quad \langle \boldsymbol{v}_k, f(\omega_\alpha) - A(q(\omega_\alpha); \sum_n u_n^\alpha \boldsymbol{v}_n)\rangle = 0, \tag{20}$$

which can be solved directly for the u_n^α for each α independently with the solver Eq. (6), i.e. B systems of N uncoupled equations—which are just samples at ω_α.

In the Bubnov-Galerkin case, if additionally the equation is a gradient as in Eq. (14)—which we will assume from now on—the Eq. (19) is equivalent with minimising the potential Π in Eq. (15) over the subspace $\mathcal{U}_N \otimes \mathcal{S}_B \subset \mathcal{U} \otimes \mathcal{S}$. In Sect. 3.3 we will look at low-rank tensor representations, for a rank-R tensor these may be seen as multi-linear maps $F_R \in \mathscr{L}^R(\mathcal{U} \times \mathcal{S}, \mathcal{U} \otimes \mathcal{S})$, i.e. $F_R : \mathcal{U}^R \times \mathcal{S}^R \to \mathscr{U} = \mathcal{U} \otimes \mathcal{S}$; they give a formal way to describe such representations. Then we will replace the functional by the composition $\Pi \circ F_r$ or similar for some r, and thereby pose the minimisation problem on $\mathcal{U}^r \times \mathcal{S}^r$.

3.3 Greedy Methods for Low-Rank Approximations

In the case of minimising the potential Π in Eq. (15) over $\mathcal{U}_N \otimes \mathcal{S}_B$, we want to represent $u(\boldsymbol{\xi})$ from Eq. (18) with a small or low *tensor rank* R as

$$u(\boldsymbol{\xi}) \approx \sum_{r=1}^{R} \boldsymbol{w}^r \otimes \phi^r = \sum_{r=1}^{R} \phi^r(\boldsymbol{\xi})\boldsymbol{w}^r. \tag{21}$$

One may observe that the tensor (u_n^α) has $N \times B$ terms, whereas a *canonical decomposition* such as Eq. (21) has $R \times (N + B)$. If $R \ll \min(N, B)$, which we hope for and what we mean by a *low-rank* separated representation, then $R \times (N + B)$

$\ll (N \times B)$. This not only saves memory, but most importantly also computation when we can operate on the terms in Eq. (21) directly.

Assume that we have already found $R-1$ terms in the approximation Eq. (21), then for the next step define $u^R(\boldsymbol{\xi}) = \sum_{r=1}^{R-1} \phi^r(\boldsymbol{\xi})\boldsymbol{w}^r$, and the incremental potential Π_R

$$\Pi_R(\boldsymbol{w}^R, \phi^R) := \Pi(u^R + \boldsymbol{w}^R \otimes \phi^R). \tag{22}$$

The new terms w_R, ϕ_R to be found are determined by minimising the incremental potential Π_R w.r.t. w_r and ϕ_R; the stationarity condition is for all α, k:

$$\langle \delta_{w^R} \Pi_R(\boldsymbol{w}^R, \phi_R), \boldsymbol{v}_k \rangle_{\mathcal{U}} = \langle\!\langle \delta_u \Pi(q(\boldsymbol{\xi}); u^R(\boldsymbol{\xi}) + \phi^R(\boldsymbol{\xi})\boldsymbol{w}^R), \boldsymbol{v}_k \otimes \phi^R(\boldsymbol{\xi}) \rangle\!\rangle = 0, \tag{23}$$

$$\langle \delta_{\phi^R} \Pi_R(\boldsymbol{w}^R, \phi^R), X_\alpha \rangle_{\mathcal{S}} = \langle\!\langle \delta_u \Pi(q(\boldsymbol{\xi}); u^R(\boldsymbol{\xi}) + \phi^R(\boldsymbol{\xi})\boldsymbol{w}^R), \boldsymbol{w}^R \otimes X_\alpha(\boldsymbol{\xi}) \rangle\!\rangle = 0. \tag{24}$$

In contrast to Eq. (19), the system in Eq. (23) is of size N—determining $\boldsymbol{w}^R$ —and the system of Eq. (24) is of size B determining ϕ^R. In more detail, Eq. (23) —to be solved for the N unknowns $(w_n^R)_{n=1}^N$ in $\boldsymbol{w}^R = \sum_{n=1}^N w_n^R \boldsymbol{v}_n$ —reads:

$$\begin{aligned} \forall k: \quad & \langle\!\langle A(q(\boldsymbol{\xi}); u^R(\boldsymbol{\xi}) + \phi^R(\boldsymbol{\xi})\boldsymbol{w}^R) - f(\boldsymbol{\xi}), \boldsymbol{v}_k \otimes \phi^R \rangle\!\rangle \\ & = \langle \mathbb{E}\left([A(q(\boldsymbol{\xi}); u^R(\boldsymbol{\xi}) + \phi^R(\boldsymbol{\xi}) \sum_n w_n^R \boldsymbol{v}_n) - f(\boldsymbol{\xi})]\phi^R(\boldsymbol{\xi}) \right), \boldsymbol{v}_k \rangle_{\mathcal{U}} = 0; \end{aligned} \tag{25}$$

one may observe that the operator is not a sample, but some average weighted with ϕ^R. Similarly, the Eq. (24) to determine the B unknowns $(\phi_\beta^R)_{\beta=1}^B$ in $\phi^R(\boldsymbol{\xi}) = \sum_\beta \phi_\beta^R X_\beta(\boldsymbol{\xi})$ is in more detail

$$\begin{aligned} \forall \alpha: \quad & \langle\!\langle A(q(\boldsymbol{\xi}); u^R(\boldsymbol{\xi}) + \phi^R(\boldsymbol{\xi})\boldsymbol{w}^R) - f(\boldsymbol{\xi}), \boldsymbol{w}^R \otimes X_\alpha \rangle\!\rangle \\ & = \mathbb{E}\left(\langle A(q(\boldsymbol{\xi}); u^R(\boldsymbol{\xi}) + (\sum_\beta \phi_\beta^R X_\beta(\boldsymbol{\xi}))\boldsymbol{w}^R) - f(\boldsymbol{\xi}), \boldsymbol{w}^R \rangle_{\mathcal{U}} X_\alpha(\boldsymbol{\xi}) \right) = 0. \end{aligned} \tag{26}$$

The basic greedy rank-one updating algorithm—it may also be called the basic ingredient in separated representation, proper generalised decomposition (PGD), or successive rank-one updating (SR1U), a form of alternating least squares (ALS)—is simply formulated in Algorithm 2.

Algorithm 2 Basic greedy rank-one updating

```
$u^1 := 0$;
for $R := 1, \ldots$ to sufficient accuracy do
    $\phi_0^R := 1$;
    for $k := 1, \ldots$ to convergence do
        Solve Eq. (25) for $\boldsymbol{w}_k^R$ resp. $(w_n^R)_k$; minimising $\Pi_R$ w.r.t. $\boldsymbol{w}_k^R$.
        Solve Eq. (26) for $\phi_k^R$ resp. $(\phi_\beta^R)_k$; minimising $\Pi_R$ w.r.t. $\phi_k^R$.
    end for
    $\boldsymbol{w}^R := \boldsymbol{w}_k^R = \sum_n w_{n,k}^R \boldsymbol{v}_n$;
    $\phi^R := \phi_k^R := \sum_\beta \phi_{\beta,k}^R X_\beta$;
    $u^{R+1} := u^R + \boldsymbol{w}^R \otimes \phi^R$;
end for
```

The innermost loop can again be seen as a block-Gauss-Seidel method to solve the coupled system Eqs. (23) and (24), and may certainly be accelerated. Further improvements are cummulatively possible. First one may note that the $X_\beta(\boldsymbol{\xi}) = X_\beta(\xi_1, \ldots, \xi_M)$ are multivariate functions $X_\beta : \mathbb{R}^M \to \mathbb{R}$, and may be further split up into tensor products, e.g. we make $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_M)$ a multi-index and write $\phi^R(\boldsymbol{\xi}) = \phi^R(\xi_1, \ldots, \xi_M) = \sum_{\boldsymbol{\beta}} \phi_{\boldsymbol{\beta}}^R \left(\prod_{m=1}^M \hat{X}_{\beta_m}(\xi_m)\right)$ with some univariate functions $\hat{X}_j$; this would allow for even 'finer' computations in Algorithm 2, minimising in the direction of each $\hat{X}_j$ separately.

Using the notation at the end of Sect. 3.2 for tensor representations, the incremental potential in Eq. (22) may also be written as

$$\Pi_R(\boldsymbol{w}^R, \phi^R) := \Pi(u^R + F_1(\boldsymbol{w}^R, \phi^R)), \tag{27}$$

where $F_1(\boldsymbol{w}^R, \phi^R) = \boldsymbol{w}^R \otimes \phi^R$; the Algorithm 2 in the innermost loop then minimises Π_R in Eq. (27) over the space $\mathcal{U}_N \times \mathcal{S}_B$. An extension of the above algorithm—although at some additional cost—is to use the maps $F_R : \mathcal{U}_N^R \times \mathcal{S}_B^R \to \mathcal{U}_N \otimes \mathcal{S}_B$ defined by the decomposition Eq. (21)

$$F_R : (\boldsymbol{w}^1, \ldots, \boldsymbol{w}^R, \phi^1, \ldots, \phi^R) \mapsto \sum_{r=1}^R \boldsymbol{w}^r \otimes \phi^r,$$

and then—after each increase of R—to optimise in the innermost loop the functional

$$\hat{\Pi}_R(\boldsymbol{w}^1, \ldots, \boldsymbol{w}^R, \phi^1, \ldots, \phi^R) := \Pi(F_R(\boldsymbol{w}^1, \ldots, \boldsymbol{w}^R, \phi^1, \ldots, \phi^R)) = \Pi\left(\sum_{r=1}^R \boldsymbol{w}^r \otimes \phi^r\right)$$

over the vector space $\mathcal{U}_N^R \times \mathcal{S}_B^R$ instead of the functional Π_R from Eq. (27). The example to be shown in Sect. 5 was computed in this way. For the sake of brevity we will not spell out this algorithm, more details may be found in [12].

4 Stochastic Coupled Problems

After all this preparation, let us return to Eqs. (7) and (8), the type of system we are interested in. We assume that analogous to Sect. 3.2 the uncertainties q_I in subsystem I have been expressed or approximated by a set of independent RVs $\boldsymbol{\xi}_I = (\xi_{1,I}, \ldots, \xi_{M_I,I})$, i.e. $q_I = q_I(\boldsymbol{\xi}_I)$, and similarly the uncertainties q_{II} in subsystem II by $\boldsymbol{\xi}_{II} = (\xi_{1,II}, \ldots, \xi_{M_{II},II})$, i.e. $q_{II} = q_{II}(\boldsymbol{\xi}_{II})$.

The quantities describing the solution will then be functions of both $\boldsymbol{\xi}_I$ and $\boldsymbol{\xi}_{II}$, defined on an $M_I + M_{II}$ dimensional space, i.e. the stochastic modelling leads to additional coupling. This is the difficulty, that with each additional coupled system the dimension of the underlying variable space grows. This can of course not be avoided, but mitigated through a low-rank separated or tensor representation. Taking the solution $u_I(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II})$ of system I in Eq. (7) as an example, we know that—similarly to Eq. (21)—it is representable as

$$u_I(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}) \approx \sum_{r=1}^{R} \phi_I^r(\boldsymbol{\xi}_I)\phi_{II}^r(\boldsymbol{\xi}_{II})\boldsymbol{w}_I^r = \sum_{r=1}^{R} \boldsymbol{w}_I^r \otimes \phi_I^r \otimes \phi_{II}^r, \tag{28}$$

as it is an element of $\mathcal{U}_I \otimes \mathcal{S}_I \otimes \mathcal{S}_{II}$. Considering all three components (u_I, u_{II}, λ), they are elements of the tensor product space $(\mathcal{U}_I \times \mathcal{U}_{II} \times \mathcal{M}) \otimes (\mathcal{S}_{B_I} \otimes \mathcal{S}_{B_{II}})$, where $\mathcal{M}$ is the spatial space of Lagrange multipliers λ. Again we assume that finite dimensional subspaces / bases

$$(\text{span}\{\boldsymbol{v}_{I,i}\}_{i=1}^{N^I} \times \text{span}\{\boldsymbol{v}_{II,j}\}_{i=1}^{N^{II}} \times \text{span}\{\boldsymbol{v}_\ell\}_{\ell=1}^{L}) = (\mathcal{U}_{N^I} \times \mathcal{U}_{N^{II}} \times \mathcal{M}_L) \subset (\mathcal{U}_I \times \mathcal{U}_{II} \times \mathcal{M})$$

and

$$(\text{span}\{X_{I,\alpha}\}_{\alpha=1}^{B^I} \times \text{span}\{X_{II,\beta}\}_{\beta=1}^{B^{II}}) = (\mathcal{S}_{B^I} \times \mathcal{S}_{B^{II}}) \subset (\mathcal{S}_I \times \mathcal{S}_{II})$$

have been chosen.

4.1 Stochastic Separated Representation

We turn right away to the linear coupled problem Eq. (9), which are the conditions for a saddle-point of the Lagranrian Eq. (10). Here we make a low-rank ‘ansatz’ as in Eq. (28) for all solution quantities:

$$\begin{bmatrix} u_I(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}) \\ u_{II}(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}) \\ \lambda(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}) \end{bmatrix} \approx \sum_{r=1}^{R} \phi_I^r(\boldsymbol{\xi}_I)\phi_{II}^r(\boldsymbol{\xi}_{II}) \begin{bmatrix} \boldsymbol{w}_I^r \\ \boldsymbol{w}_{II}^r \\ \boldsymbol{\mu}^r \end{bmatrix}. \tag{29}$$

The sum will again be computed term-by-term as in Algorithm 2. Following the general case Eq. (15), first define the corresponding stochastic Lagrangian as expected value of the deterministic Lagranian Π in Eq. (10):

$$\begin{aligned}\Pi(q_I, q_{II}; u_I, u_{II}, \lambda) := \Pi_I(q_I, ; u_I) + \Pi_{II}(q_{II}, ; u_{II}) + \lambda^T(\hat{C}_I u_I - \hat{C}_{II} u_{II})\\ = \mathbb{E}\left(\Pi_I(q_I(\boldsymbol{\xi}_I); u_I(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}))\right) + \mathbb{E}\left(\Pi_{II}(q_{II}(\boldsymbol{\xi}_{II}); u_{II}(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}))\right)\\ + \mathbb{E}\left(\lambda(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II})^T(C_I u_I(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}) - C_{II} u_{II}(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}))\right) \qquad (30)\end{aligned}$$

Following the general prescription in Sect. 3.3, as in Eq. (22), assume that a low-rank representation like Eq. (29) up to terms of $R-1$ has already been computed, define the abbreviations

$$\begin{bmatrix} u_I^R(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}) \\ u_{II}^R(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}) \\ \lambda^R(\boldsymbol{\xi}_I, \boldsymbol{\xi}_{II}) \end{bmatrix} := \sum_{r=1}^{R-1} \begin{bmatrix} \boldsymbol{w}_I^r \\ \boldsymbol{w}_{II}^r \\ \boldsymbol{\mu}^r \end{bmatrix} \phi_I^r(\boldsymbol{\xi}_I)\phi_{II}^r(\boldsymbol{\xi}_{II}),$$

and look at the incremental Lagrangian (cf. Eq. (22)) corresponding to Eq. (30):

$$\Pi_R(\boldsymbol{w}_I^R, \boldsymbol{w}_{II}^R, \boldsymbol{\mu}^R, \phi_I^R, \phi_{II}^R) = \Pi(q_I, q_{II}; u_I^R + \boldsymbol{w}_I^R\phi_I^R\phi_{II}^R, u_{II}^R + \boldsymbol{w}_{II}^R\phi_I^R\phi_{II}^R, \lambda^R + \boldsymbol{\mu}^R\phi_I^R\phi_{II}^R). \qquad (31)$$

4.2 Partitioned Greedy Algorithm

The conditions for stationarity of the incremental Lagrangian Π_R from Eq. (31)—as before in Eqs. (23) and (24) —are

$$\forall\, \boldsymbol{v}_{I,i}, \boldsymbol{v}_{II,j}, \boldsymbol{v}_\ell: \quad 0 = \begin{bmatrix} \delta_{w_I^R}\Pi_R \\ \delta_{w_{II}^R}\Pi_R \\ \delta_{\mu^R}\Pi_R \end{bmatrix} = \begin{bmatrix} \langle\langle \delta_{u_I}\Pi, \boldsymbol{v}_{I,i} \otimes \phi_I \otimes \phi_{II} \rangle\rangle \\ \langle\langle \delta_{u_{II}}\Pi, \boldsymbol{v}_{II,i} \otimes \phi_I \otimes \phi_{II} \rangle\rangle \\ \langle\langle \delta_\lambda\Pi, \boldsymbol{v}_\ell \otimes \phi_I \otimes \phi_{II} \rangle\rangle \end{bmatrix}, \qquad (32)$$

and on the stochastic variables

$$\forall\, X_{I,\alpha}, X_{II,\beta}: \quad 0 = \begin{bmatrix} \delta_{\phi_I}\Pi_R \\ \delta_{\phi_{II}}\Pi_R \end{bmatrix} =$$
$$\begin{bmatrix} \langle\langle \delta_{u_I}\Pi, \boldsymbol{w}_I^R \otimes X_{I,\alpha} \otimes \phi_{II} \rangle\rangle + \langle\langle \delta_{u_{II}}\Pi, \boldsymbol{w}_{II}^R \otimes X_{I,\alpha} \otimes \phi_{II} \rangle\rangle + \langle\langle \delta_\lambda\Pi, \boldsymbol{\mu}^R \otimes X_{I,\alpha} \otimes \phi_{II} \rangle\rangle \\ \langle\langle \delta_{u_I}\Pi, \boldsymbol{w}_I^R \otimes \phi_I \otimes X_{II,\beta} \rangle\rangle + \langle\langle \delta_{u_{II}}\Pi, \boldsymbol{w}_{II}^R \otimes \phi_I \otimes X_{II,\beta} \rangle\rangle + \langle\langle \delta_\lambda\Pi, \boldsymbol{\mu}^R \otimes \phi_I \otimes X_{II,\beta} \rangle\rangle \end{bmatrix}. \qquad (33)$$

From this and Eqs. (10) and (30) one obtains after a short calculation for Eq. (32)

$$\begin{bmatrix} \hat{K}_I & 0 & \hat{C}_I^T \\ 0 & \hat{K}_{II} & -\hat{C}_{II}^T \\ \hat{C}_I & -\hat{C}_{II} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_I^R \\ \boldsymbol{w}_{II}^R \\ \boldsymbol{\mu}^R \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{f}}_I \\ \hat{\boldsymbol{f}}_{II} \\ \hat{\boldsymbol{g}} \end{bmatrix}, \tag{34}$$

which has exactly the same size and structure as Eq. (9). Hence it can be solved with the *deterministic* FETI algorithm. The averaged deterministic-size terms in Eq. (34) are

$$\begin{aligned}
\hat{K}_I &= \mathbb{E}\left(\phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\ K_I(\boldsymbol{\xi}_I)\ \phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\right) = \mathbb{E}\left(\phi_{II}^R(\boldsymbol{\xi}_{II})^2\right)\ \mathbb{E}\left(\phi_I^R(\boldsymbol{\xi}_I)^2\ K_I(\boldsymbol{\xi}_I)\right),\\
\hat{K}_{II} &= \mathbb{E}\left(\phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\ K_{II}(\boldsymbol{\xi}_{II})\ \phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\right) = \mathbb{E}\left(\phi_I^R(\boldsymbol{\xi}_I)^2\right)\ \mathbb{E}\left(\phi_{II}^R(\boldsymbol{\xi}_{II})^2\ K_{II}(\boldsymbol{\xi}_{II})\right),\\
\hat{C}_I &= C_I\ \mathbb{E}\left(\phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\right), \quad \hat{C}_{II} = C_{II}\ \mathbb{E}\left(\phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\right),\\
\hat{\boldsymbol{f}}_I &= \mathbb{E}\left((f_I(\boldsymbol{\xi}_I) - K_I(\boldsymbol{\xi}_I)u_I^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II}))\ \phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\right),\\
\hat{\boldsymbol{f}}_{II} &= \mathbb{E}\left((f_{II}(\boldsymbol{\xi}_{II}) - K_{II}(\boldsymbol{\xi}_{II})u_{II}^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II}))\ \phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\right),\\
\hat{\boldsymbol{g}} &= \mathbb{E}\left((C_I^T u_I^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II}) - C_{II}^T u_{II}^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II}))\ \phi_I^R(\boldsymbol{\xi}_I)\phi_{II}^R(\boldsymbol{\xi}_{II})\right).
\end{aligned}$$

For Eq. (33) one obtains in the first relation for $\phi_I^R = \sum_\gamma \phi_{I,\gamma}^R X_{I,\gamma}$ after some computation $\forall X_{I,\alpha}$:

$$\sum_\gamma \mathbb{E}\left(X_{I,\alpha}(\boldsymbol{\xi}_I)\,\phi_{II}^R(\boldsymbol{\xi}_{II})^2\,\hat{k}(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II})\,X_{I,\gamma}(\boldsymbol{\xi}_I)\right)\ \phi_{I,\gamma}^R = \mathbb{E}\left(\phi_{II}^R(\boldsymbol{\xi}_{II})\,\varrho(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II})\,X_{I,\alpha}(\boldsymbol{\xi}_I)\right), \tag{35}$$

with the abbreviations $\varrho(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II}) := u_I^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II})^T f_I(\boldsymbol{\xi}_I) + u_{II}^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II})^T f_{II}(\boldsymbol{\xi}_{II})$ and

$$\hat{k}(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II}) := u_I^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II})^T\ K_I(\boldsymbol{\xi}_I)\ u_I^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II}) + u_{II}^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II})^T\ K_{II}(\boldsymbol{\xi}_{II})\ u_{II}^R(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II}).$$

The Eq. (35) is a linear symmetric positive definite system of size $B^I \times B^I$, as it is equivalent to the minimisation of Π_R. Analogously, for $\phi_{II}^R = \sum_\gamma \phi_{II,\gamma}^R X_{II,\gamma}$, the second relation in Eq. (33) yields a similar linear system of size $B^{II} \times B^{II}$ such that $\forall X_{II,\beta}$:

$$\sum_\gamma \mathbb{E}\left(X_{II,\beta}(\boldsymbol{\xi}_{II})\,\phi_I^R(\boldsymbol{\xi}_I)^2\,\hat{k}(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II})\,X_{II,\gamma}(\boldsymbol{\xi}_{II})\right)\ \phi_{II,\gamma}^R = \mathbb{E}\left(\phi_I^R(\boldsymbol{\xi}_I)\,\varrho(\boldsymbol{\xi}_I,\boldsymbol{\xi}_{II})\,X_{II,\beta}(\boldsymbol{\xi}_{II})\right). \tag{36}$$

The procedure to compute the *separated* solution in a *partitioned* way is given by Algorithm 3.

Of course it is possible to extend this algorithm as alluded to at the end of Sect. 3.3, namely to not only solve saddle point problems on $\mathcal{U}_{N^I} \times \mathcal{U}_{N^{II}} \times \mathcal{M}_L \times \mathcal{S}_{B^I} \times \mathcal{S}_{B^{II}}$ as it is done in Algorithm 3, but on $\mathcal{U}_{N^I}^R \times \mathcal{U}_{N^{II}}^R \times \mathcal{M}_L^R \times \mathcal{S}_{B^I}^R \times \mathcal{S}_{B^{II}}^R$, details

Algorithm 3 Partitioned greedy rank-one updating

```
$u_I^1 := 0; u_{II}^1 := 0; \lambda^1 := 0;$
**for** $R := 1, \ldots$ **to** sufficient accuracy **do**
  $\phi_{I,0}^R := 1; \phi_{II,0}^R := 1;$
  **for** $k := 1, \ldots$ **to** convergence **do**
    Solve linear saddle point system Eq. (34) for $(\boldsymbol{w}_{I,k}^R, \boldsymbol{w}_{II,k}^R, \boldsymbol{\mu}_k^R)$ with FETI using $(\phi_{I,k-1}^R, \phi_{II,k-1}^R)$.
    Solve linear s.p.d. system Eq. (35) for $\phi_{I,k}^R$ using $(\boldsymbol{w}_{I,k}^R, \boldsymbol{w}_{II,k}^R, \boldsymbol{\mu}_k^R, \phi_{II,k-1}^R)$, minimising $\Pi_R$ w.r.t. $\phi_{I,k}^R$.
    Solve linear s.p.d. system Eq. (36) for $\phi_{II,k}^R$ using $(\boldsymbol{w}_{I,k}^R, \boldsymbol{w}_{II,k}^R, \boldsymbol{\mu}_k^R, \phi_{I,k}^R)$, minimising $\Pi_R$ w.r.t. $\phi_{II,k}^R$.
  **end for**
  $\boldsymbol{w}_I^R := \boldsymbol{w}_{I,k}^R; \boldsymbol{w}_{II}^R := \boldsymbol{w}_{II,k}^R; \boldsymbol{\mu}^R := \boldsymbol{\mu}_k^R; \phi_I^R := \phi_{I,k}^R; \phi_{II}^R := \phi_{II,k}^R;$
  $u_I^{R+1} := u_I^R + \boldsymbol{w}_I^R \otimes \phi_I^R \otimes \phi_{II}^R; u_{II}^{R+1} := u_{II}^R + \boldsymbol{w}_I^R \otimes \phi_I^R \otimes \phi_{II}^R; \lambda^{R+1} := \lambda^R + \boldsymbol{\mu}_I^R \otimes \phi_I^R \otimes \phi_{II}^R;$
**end for**
```

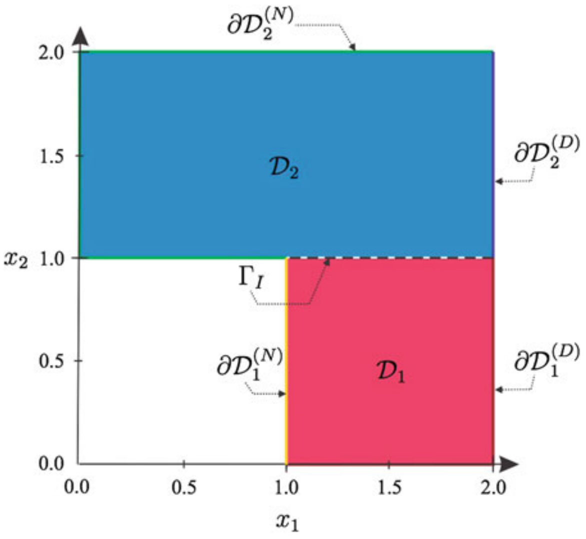

$\partial\mathcal{D}_2^{(N)}$
$\partial\mathcal{D}_2^{(D)}$
$\mathcal{D}_2$
Γ_I
$\partial\mathcal{D}_1^{(N)}$
$\mathcal{D}_1$
$\partial\mathcal{D}_1^{(D)}$
x_2
x_1
0.0
0.5
1.0
1.5
2.0

Fig. 1 L-shaped domain: coupled diffusion problem

may again be found in [12]. Of course, in that case each subproblem in the innermost loop is R times larger, but the algorithm can find a better approximation with smaller R. This is how the example in the following Sect. 5 was computed.

5 Computational Example

The example of a coupled problem—taken from [12] and shown here in Fig. 1—can equally well be viewed as a problem which has been partitioned. This exemplifies another way in which the methods presented here can be used, namely to partition

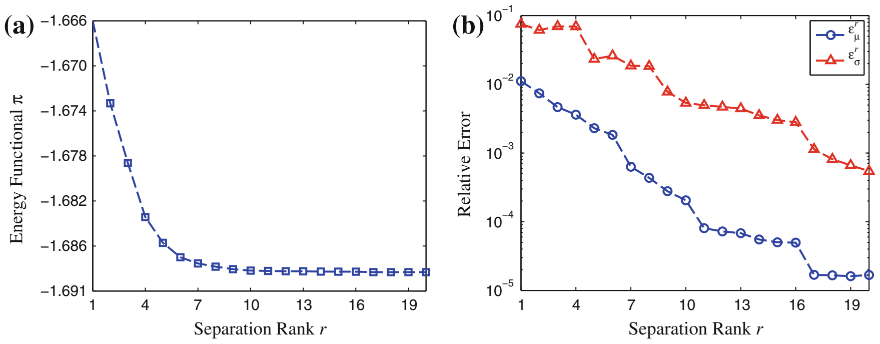


Fig. 2 Separated Approximation: **a** Value of the Lagrangian, **b** Errors: mean and std. deviation

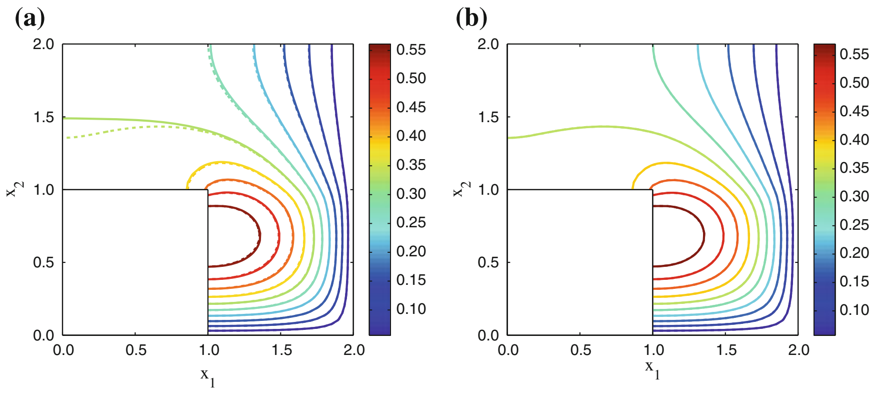


Fig. 3 Separated Approximation—Mean: **a** Contours for $R = 1$, **b** Contours for $R = 20$

'large' problems to break them up into manageable pieces. The L-shaped domain in Fig. 1 has been partitioned as indicated, and thus is a *coupled* problem. It is a diffusion problem with a random diffusion coefficient, where we assume that the uncertainty in this coefficient can be modelled by independent RVs belonging to the respective subdomains. This means that the diffusion coefficient in both subdomains is not correlated.

The coupling conditions are enforced by Lagrange multipliers, and the coupled problem was solved by a FETI method, and the total problem by the extension of Algorithm 3 as alluded to at the end of Sect. 4.2. In Fig. 2a the convergence of the value of the Lagrangian with increasing rank can be observed, where it may be noted that beyond rank $R = 7$ the value does not change much any more. In Fig. 2b we show the decrease of error for the overall mean and standard deviation with increasing rank. In the following two figures we show the contour lines—for the mean in Fig. 3 and for the standard deviation in Fig. 4—for rank $R = 1$ in part a) and for $R = 20$ in part b). The converged contours are shown as dashed lines in all cases. It may be observed that for the mean in Fig. 3a the contour lines are already quite accurate for $R = 1$.

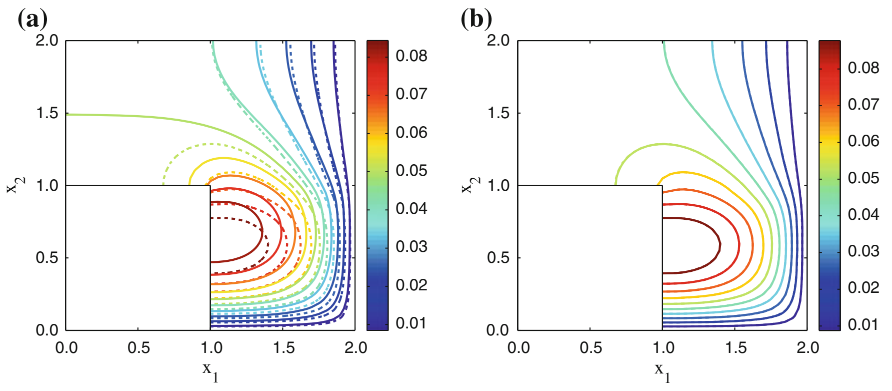


Fig. 4 Separated Approximation—Standard Deviation: **a** Contours for $R = 1$, **b** Contours for $R = 20$

6 Concluding Remarks

We have indicated a *fast* but partitioned computational framework for the propagation of uncertainty through coupled problems, which also saves storage. The proposed approach constructs a solution-adaptive stochastic basis of separated form with respect to the random inputs characterising the uncertainty in each sub-problem, leading to a *partitioned* treatment of the stochastic space and consequently to a higher scalability of the method as compared with standard uncertainty propagation approaches. For situations where the separation rank is small, the proposed approach provides at the same time a reduced order representation of the coupled solution.

The deterministic coefficients associated with each separated stochastic basis capture the spatial variability of the solution and are computed via the standard finite element tearing and interconnecting (FETI) approach. Therefore, the method achieves a high level of parallelism while requiring no intrusion in each domain solver. Although our present formulation of domain coupling is based on the standard FETI approach, we foresee no major technical difficulties in employing more advanced domain coupling schemes.

The proposed framework was demonstrated through its application to a linear elliptic PDE with high-dimensional random inputs. Despite the high-dimensionality of the random inputs, accurate estimates of the solution statistics were achieved with relatively low separation ranks, thus demonstrating the effectiveness of the present approach.

Acknowledgments The authors are indebted for the fruitful discussions they had with Prof. K.C. Park from University of Colorado, Boulder. AD gratefully acknowledges the financial support of the Department of Energy under Advanced Scientific Computing Research Early Career Research Award DE-SC0006402. MH's work was supported by the National Science Foundation grant CMMI-1201207. The work of HGM and RN has been partly supported by the German Research Foundation "Deutsche Forschungsgemeinschaft" (DFG).

References

1. Babuška I, Chatzipantelidis P (2002) On solving elliptic stochastic partial differential equations. Comput Methods Appl Mech Eng 191(37–38):4093–4122
2. Babuška I, Tempone R, Zouraris G (2004) Galerkin finite element approximations of stochastic elliptic partial differential equations. SIAM J Numer Anal 42(2):800–825
3. Bieri M, Andreev R, Schwab C (2010) Sparse tensor discretization of elliptic sPDEs. SIAM J Sci Comput 31:4281–4304
4. Doostan A, Iaccarino G (2009) A least-squares approximation of partial differential equations with high-dimensional random inputs. J Comput Phys 228(12):4332–4345
5. Doostan A, Owhadi H (2011) A non-adapted sparse approximation of PDEs with stochastic inputs. J Comput Phys 230:3015–3034
6. Doostan A, Ghanem R, Red-Horse J (2007) Stochastic model reduction for chaos representations. Comput Meth Appl Mech Eng 196(37–40):3951–3966
7. Farhat C, Roux F (1991) A method of finite element tearing and interconnecting and its parallel solution algorithm. Int J Numer Meth Eng 32:1205–1227
8. Farhat C, Lesoinne M, LeTallec P, Pierson K, Rixen D (2001) FETI-DP: a dual-primal unified FETI method-part i: a faster alternative to the two-level FETI method. Int J Num Meth Eng 50:1523–1544
9. Foo J, Karniadakis G (2010) Multi-element probabilistic collocation method in high dimensions. J Comput Phys 229(5):1536–1557
10. Ghosh D, Avery P, Farhat C (2009) A FETI-preconditioned congugate gradient method for large-scale stochastic finite element problems. Int J Numer Meth Eng 80:914–931
11. Giraldi L, Litvinenko A, Liu D, Matthies HG, Nouy A (2013) To be or not to be intrusive? the solution of parametric and stochastic equations–the "plain vanilla" Galerkin case. arXiv:1309.1617v1:[math.NA]. http://arxiv.org/abs/1309.1617v1
12. Hadigol M, Doostan A, Matthies HG, Niekamp R (2013) Partitioned treatment of uncertainty in coupled domain problems: a separated representation approach. arXiv:1305.6818:[math.PR]. http://arxiv.org/abs/1305.6818
13. Khoromskij B, Schwab C (2011) Tensor-structured Galerkin approximation of parametric and stochastic elliptic PDEs. SIAM J Sci Comput 33(1):364–385
14. Le Maître O, Knio O (2010) Spectral methods for uncertainty quantification with applications to computational fluid dynamics. Springer, New York
15. Ma X, Zabaras N (2009) An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations. J Comput Phys 228:3084–3113
16. Matthies HG (2008) Stochastic finite elements: computational approaches to stochastic partial differential equations. Z Angew Math Mech 88:849–873
17. Matthies HG, Keese A (2005) Galerkin methods for linear and nonlinear elliptic stochastic partial differential equations. Comput Meth Appl Mech Eng 4:1295–1331
18. Matthies HG, Niekamp R, Steindorf J (2006) Algorithms for strong coupling procedures. Comput Meth Appl Mech Eng 195(17–18):2028–2049. doi:10.1016/j.cma.2004.11.032
19. Matthies HG, Litvinenko A, Pajonk O, Rosić BV, Zander E (2012) Parametric and uncertainty computations with tensor product representations. In: Dienstfrey A, Boisvert R (eds) Uncertainty quantification in scientific computing. IFIP Advances in Information and Communication Technology, vol 377. Springer, Berlin, pp 139–150. doi:10.1007/978-3-642-32677-6
20. Najm H (2009) Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics. Ann Rev 41(1):35–52
21. Nobile F, Tempone R, Webster C (2008) An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data. SIAM J Numer Anal 46(5):2411–2442
22. Nouy A (2010) Proper generalized decompositions and separated representations for the numerical solution of high dimensional stochastic problems. Arch Comput Meth Eng 17:403–434
23. Park KC, Felippa CA (1998) A variational framework for solution method developments in structural mechanics. J Appl Mech 56(1):242–249

24. Subber W, Sarkar A (2012) Domain decomposition method of stochastic PDEs: a two-level scalable preconditioner. J Phys: Conf Ser 341(1):012033
25. Xiu D (2009) Fast numerical methods for stochastic computations: a review. Commun Comput Phys 5(2–4):242–272
26. Xiu D (2010) Numerical methods for stochastic computations: a spectral method approach. Princeton University Press, Princeton
27. Xiu D, Hesthaven J (2005) High-order collocation methods for differential equations with random inputs. SIAM J Sci Comput 27(3):1118–1139
28. Zhang Z, Choi M, Karniadakis GE (2009) Anchor points matter in ANOVA decomposition. In: Spectral and higher order methods for partial differential equations. Lecture Notes in Computational Science and Engineering, Trondheim, pp 347–355