4

Application of the Discrete Empirical Interpolation Method to Reduced Order Modeling of Nonlinear and Parametric Systems*

Harbir Antil, Matthias Heinkenschloss and Danny C. Sorensen

Abstract Projection based methods lead to reduced order models (ROMs) with dramatically reduced numbers of equations and unknowns. However, for nonlinear or parametrically varying problems the cost of evaluating these ROMs still depends on the size of the full order model and therefore is still expensive. The Discrete Empirical Interpolation Method (DEIM) further approximates the nonlinearity in the projection based ROM. The resulting DEIM ROM nonlinearity depends only on a few components of the original nonlinearity. If each component of the original nonlinearity depends only on a few components of the argument, the resulting DEIM ROM can be evaluated efficiently at a cost that is independent of the size of the original problem. For systems obtained from finite difference approximations, the ith component of the original nonlinearity often depends only on the ith component of the argument. This is different for systems obtained using finite element methods, where the dependence is determined by the mesh and by the polynomial degree of the finite element subspaces. This paper describes two approaches of applying DEIM in the finite element context, one applied to the assembled and the other to the unassembled form of the nonlinearity. We carefully examine how the DEIM is applied in each case, and the substantial efficiency gains obtained by the DEIM. In addition, we demonstrate how to apply DEIM to obtain ROMs for a class of parameterized system that arises, e.g., in shape optimization. The evaluations of the DEIM ROMs are substantially faster than those of the standard projection based ROMs. Additional

H. Antil (✉)
Department of Mathematical Sciences, George Mason University Fairfax, VA 22030
e-mail: hantil@gmu.edu

M. Heinkenschloss · D.C. Sorensen
Department of Computational and Applied Mathematics, Rice University, Houston, TX 77005-1892
e-mail: heinken@rice.edu, sorensen@rice.edu

* This research was supported in part by AFOSR grant FA9550-12-1-0155 and NSF grants CCF-1017401, DMS-0915238, DMS-1115345.

A. Quarteroni, G. Rozza (eds.): *Reduced Order Methods for Modeling and Computational Reduction*, MS&A 9. DOI 10.1007/978-3-319-02090-7_4, © Springer International Publishing Switzerland 2014

gains are obtained with the DEIM ROMs when one has to compute derivatives of the model with respect to the parameter.

4.1 Introduction

Projection based reduced order models systematically extract the features of very large-scale systems to approximate these systems by substantially smaller ones. However, if the original system is parameter dependent or is semilinear, then, although the new small system involves substantially fewer equations and unknowns than the original one, the computational cost of its numerical solution can be essentially the same as that of the original large-scale system. The discrete empirical interpolation method (DEIM) of [7] further approximates projection based reduced order models to obtain small systems that capture the solution of the original large-scale system and that can also be solved at a computational cost that depends only on the size of the small system, provided each component of the original semilinear function depends only on a few components of its argument. So far, the DEIM has been primarily applied to finite difference discretizations of semilinear PDEs where the ith component of the nonlinearity depends only on the ith component of the argument. This is different in finite element discretizations, where the dependence of the nonlinear function is determined by the mesh as well as by the polynomial degree used to construct the finite element spaces. Therefore results from DEIM applied to finite difference approximations of PDEs do not necessarily carry over to DEIM applied to finite element approximations of PDEs. One purpose of this paper is demonstrate two approaches to apply DEIM to finite element discretizations of semilinear PDEs and numerically study their computational cost. The two approaches apply DEIM at different stages of the finite element assembly process. The size of the nonlinear function as well as its dependence on the argument are different at each stage of the assembly process, which impacts the computational efficiency of the resulting DEIM reduced models. The second purpose of this paper is to demonstrate how to apply DEIM to a class of parameter dependent systems that arise, e.g., in shape optimization.

Discretizations of parameterized semilinear elliptic partial differential equations (PDEs) lead to large scale nonlinear algebraic systems of the form

$$\mathbf{A}(\theta)\mathbf{y} + \mathbf{F}(\mathbf{y};\theta) = \mathbf{b}(\theta), \tag{4.1}$$

where the parameters $\theta \in \Theta \subset \mathbb{R}^p$ and for each parameter θ the matrix $\mathbf{A}(\theta) \in \mathbb{R}^{N\times N}$ and the vectors $\mathbf{F}(\mathbf{y};\theta)$ and $\mathbf{b}(\theta) \in \mathbb{R}^N$. Projection based model reduction techniques [1,19,24,29] generate matrices $\mathbf{V}_\ell$ and $\mathbf{V}_r \in \mathbb{R}^{N\times n}$ with $n \ll N$ and replace (4.1) with the reduced system

$$\mathbf{V}_\ell^T\mathbf{A}(\theta)(\overline{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}) + \mathbf{V}_\ell^T\mathbf{F}(\overline{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}};\theta) = \mathbf{V}_\ell^T\mathbf{b}(\theta). \tag{4.2}$$

While the reduced order system (4.2) is much smaller than the original one, the cost of computation of $\theta \mapsto \mathbf{V}_\ell^T\mathbf{A}(\theta)\mathbf{V}_r$, $\theta \mapsto \mathbf{V}_\ell^T\mathbf{b}(\theta)$, and $(\widehat{\mathbf{y}},\theta) \mapsto \mathbf{V}_\ell^T\mathbf{F}(\overline{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}};\theta)$

still depends on N. Therefore, additional approximations are needed to obtain reduced order models that capture the original system as well as evaluate with a computational complexity that depends only on the reduced order system size n but is independent of the full order model size $N \gg n$.

The empirical interpolation method (EIM) of [2] and the DEIM of [7] generate reduced order models from (4.2) that approximate the full order model within desired error bounds and that can be numerically solved at a cost that essentially depends only on the reduced order system size. While the EIM is applied to the variational formulation that leads to the nonlinear algebraic system (4.1), its derivative, the DEIM is applied directly to discrete systems. Applications of EIM and DEIM to nonlinear finite element computations are also discussed, e.g., in [9, 15, 17, 22]. We will focus on discrete systems (4.1) and therefore consider the DEIM. Especially, we carefully expose the dependency of the computational complexity of the DEIM on the polynomial degree of the finite element method.

One purpose of this paper is the study of DEIM to nonlinear systems $\mathbf{A}\mathbf{y} + \mathbf{F}(\mathbf{y}) = \mathbf{b}$ obtained from finite element discretizations. The DEIM reduced order model is of the form $\mathbf{V}_\ell^T \mathbf{A}(\bar{\mathbf{y}} + \mathbf{V}_r \widehat{\mathbf{y}}) + \widehat{\mathbf{F}}(\widehat{\mathbf{y}}) = \mathbf{V}_\ell^T \mathbf{b}$, where $\widehat{\mathbf{F}}$ depends only on m components of the original nonlinearity $\mathbf{F}$. As we have mentioned before, the efficiency with which the DEIM reduced order model can be applied depends on how many components of the argument are needed to evaluate m components of the original nonlinearity $\mathbf{F}$. For systems obtained from finite element discretizations the dependence of $\mathbf{F}$ on its argument is determined by the mesh, as well as by the polynomial degree used to construct the finite element spaces. One can apply DEIM at different stages of the finite element assembly process. This effects the structure of the nonlinearity. We demonstrate how to apply DEIM to finite element discretizations of nonlinear PDEs in the assembled and in the unassembled form, and we numerically study the computational cost of the resulting reduced order models. Either version of the DEIM is preferable over the naive application of projection based model reduction as in (4.2). For large systems, the application of the DEIM to the so-called unassembled form of the nonlinearity leads to additional gains in the on-line cost of the reduced order models.

A second focus of this paper is the application of DEIM to generate reduced order models for parametrically dependent PDEs $\mathbf{A}(\theta)\mathbf{y} = \mathbf{b}(\theta)$, where $\mathbf{A}(\theta) = \sum_{i=1}^{M} \mathbf{g}_i(\theta)\mathbf{A}_i$ and $\mathbf{b}(\theta) = \sum_{i=1}^{M} \mathbf{l}_i(\theta)\mathbf{b}_i$. For large M the complexity of evaluating the reduced order matrix $\mathbf{V}_\ell^T \mathbf{A}(\theta)\mathbf{V}_r = \sum_{i=1}^{M} \mathbf{g}_i(\theta)\mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r$ is still high. The DEIM can be used to obtain an approximation that allows more pre-computation of matrices and that can be evaluated more efficiently in the on-line phase. Additional benefits arise when derivatives of the matrix with respect to the parameter θ have to be computed, and we illustrate these gains in the context of shape optimization.

The next section describes two model problems, a semilinear elliptic advection reaction diffusion equation and the Stokes equations on a parameterized domain, and their finite element discretizations. These problems will be used to demonstrate the application of the DEIM, and to numerically evaluate the computational costs required to solve the full and the reduced order models. Section 4.3 reviews approaches to construct the reduced order subspaces spanned by the columns of the matrices $\mathbf{V}_\ell$

and $\mathbf{V}_r \in \mathbb{R}^{N\times n}$, and it reviews the DEIM. The main contributions of this paper are presented in Sects. 4.4 and 4.5.

In Sect. 4.4 we discuss the application of the DEIM to finite element discretizations of semilinear PDEs. We illustrate how ith component of the nonlinearity depends on the components of its arguments for piecewise linear and piecewise quadratic elements, and we demonstrate how this dependence impacts the efficiency of the DEIM. In addition, we discuss the application of DEIM to the fully assembled system, as well as the unassembled form of the nonlinearity. The latter was originally suggested by [9, 33]. The nonlinear vectors are larger, but each component depends on fewer components of the argument. We describe both version of the DEIM and computationally compare them on the semilinear elliptic advection reaction diffusion model equation of Sect. 4.2.1. As we have mentioned before, either version of the DEIM is preferable over the naive application of projection based model reduction as in (4.2). For large systems, the application of the DEIM to the unassembled form of the nonlinearity is more expensive in the off-line cost, but leads to additional gains in the on-line cost of the reduced order models.

The application of the DEIM to obtain efficient reduced order models for systems with parameterized matrices $\mathbf{A}(\theta) = \sum_{i=1}^{M} \mathbf{g}_i(\theta)\mathbf{A}_i$ and vectors $\mathbf{b}(\theta) = \sum_{i=1}^{M} \mathbf{l}_i(\theta)\mathbf{b}_i$ is demonstrated in Sect. 4.5. We numerically illustrate the efficiency gains achieved by the DEIM reduced order model using the Stokes equation on parameterized domains introduced in Sect. 4.2.2. The DEIM not only leads to reduced order models that can be evaluated efficiently, but in addition it also leads to reduced order models where derivatives with respect to the parameter θ can be computed efficiently. Both efficiency gains are crucial, e.g., for shape optimization.

4.2 Model Problems

4.2.1 Semilinear Advection-Diffusion-Reaction PDE

Our first model problem is a semilinear advection diffusion reaction equation. Let $\Omega \subset \mathbb{R}^d$, $d \in \{2,3\}$ be an open, bounded Lipschitz domain with boundary $\partial\Omega = \Gamma_D \cup \Gamma_N$, where Γ_D and Γ_N corresponds to Dirichlet and Neumann parts. Given a diffusion coefficient $\nu > 0$, an advection vector $\beta \in \mathbb{R}^d$, a nonlinear function $f : \mathbb{R} \times \mathbb{R}^p \to \mathbb{R}$, and Dirichlet data h, the semilinear advection diffusion reaction equation is given by

$$-\nabla \cdot (\nu \nabla y) + \beta \cdot \nabla y + f(y,\theta) = 0, \qquad \text{in } \Omega, \tag{4.3a}$$

$$y = h, \qquad \text{on } \Gamma_D, \tag{4.3b}$$

$$\nabla y \cdot n = 0, \qquad \text{on } \Gamma_N. \tag{4.3c}$$

We consider the specific nonlinearity

$$f(y,\theta) = Ay(C-y)e^{-E/(D-y)} \tag{4.4}$$

used e.g., in [11]. Here C, D are known constants and $\theta = (\ln(A), E)$ are system parameters that can vary within the parameter domain $\Theta = [5.00, 7.25] \times [0.05, 0.15] \subset \mathbb{R}^2$.

The weak form of (4.3) is given as follows. Find $y \in H^1(\Omega)$ with $y = h$ on Γ_D such that

$$\int_\Omega \nu \nabla y \cdot \nabla v dx + \int_\Omega \beta \cdot \nabla y v dx + \int_\Omega f(y, \theta) v dx = 0 \tag{4.5}$$

for all $v \in H^1(\Omega)$ with $v = 0$ on Γ_D. Existence results for linear and nonlinear advection diffusion equations can be found, e.g., in [26, 32] and [23], [27, Sec. 6.3]

We discretize the equations using an SUPG (streamline upwind/Petrov-Galerkin) stabilized FEM [5, 10, 25]. The Dirichlet boundary conditions are implemented via interpolation. Let $\{\Omega_e\}_{e=1}^{n_e}$ be a conforming triangulation of the domain Ω. Furthermore, let $\{\phi_j\}_{j=1}^{N}$ be the piecewise polynomial nodal basis functions. To simplify the presentation, we assume that nodes with indices $1, \ldots, N_F$ are in $\overline{\Omega} \setminus \Gamma_D$ and that the nodes with indices $N_F + 1, \ldots, N_F + N_D$ are in Γ_D. We define

$$\begin{aligned} a_h(y_h, \phi) = & \int_\Omega \nu \nabla y_h(x) \cdot \nabla \phi(x) + \beta \cdot \nabla y_h(x) \phi(x) dx \\ & + \sum_{e=1}^{n_e} \int_{\Omega_e} \tau_e \, \beta \cdot \nabla \phi(x) \, (-\nabla \cdot (\nu \nabla y_h(x)) + \beta \cdot \nabla y_h(x)) dx, \end{aligned} \tag{4.6a}$$

$$F_h(y_h, \phi; \theta) = \int_\Omega f(y_h(x), \theta) \phi \, dx + \sum_{e=1}^{n_e} \int_{\Omega_e} \tau_e \, \beta \cdot \nabla \phi(x) \, f(y_h(x), \theta) dx. \tag{4.6b}$$

If we let h_e denote the length of largest side of each element Ω_e and $P_e = h_e \|\beta\| / (2\nu)$ the mesh Péclet number, then the SUPG stabilization parameter is defined as

$$\tau_e = \frac{h_e}{2\|\beta\|} \left(1 - \frac{1}{P_e}\right).$$

The solution y of (4.5) is approximated by

$$y_h(x) = \sum_{j=1}^{N_F + N_D} y_j \phi_j(x) \tag{4.7}$$

where y_h satisfies

$$a_h(y_h, \phi_i) + F_h(y_h, \phi_i; \theta) = 0, \qquad i = 1, \ldots, N_F, \tag{4.8a}$$

$$y_h(x_{N_F + i}) = h(x_{N_F + i}), \qquad i = 1, \ldots, N_D. \tag{4.8b}$$

To state the nonlinear algebraic system corresponding to (4.8), we define

$$\mathbf{y}_F = (y_1, \ldots, y_{N_F})^T, \quad \mathbf{y}_D = (y_{N_F+1}, \ldots, y_{N_F+N_D})^T,$$

$$\mathbf{h} = (h(x_{N_F+1}), \ldots, h(x_{N_F+N_D}))^T,$$

and partition the matrices and vectors into submatrices and subvectors corresponding to the free variables $\mathbf{y}_F$ and those determined by the Dirichlet boundary conditions,

(4.8) leads to a system of algebraic equations of the type

$$\begin{pmatrix} \mathbf{A}_{FF} & \mathbf{A}_{FD} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{y}_F \\ \mathbf{y}_D \end{pmatrix} + \begin{pmatrix} \mathbf{F}_F(\mathbf{y}_F, \mathbf{y}_D; \theta) \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{h} \end{pmatrix}. \tag{4.9}$$

Of course, this is equivalent to $\mathbf{A}_{FF}\mathbf{y}_F + \mathbf{F}_F(\mathbf{y}_F, \mathbf{h}; \theta) + \mathbf{A}_{FD}\mathbf{h} = \mathbf{0}$. If we set $\mathbf{b} = -\mathbf{A}_{FD}\mathbf{h}$, $N = N_F$, if we drop the subscript F, and if we drop the constant $\mathbf{h}$ from the arguments of the nonlinearity $\mathbf{F}$, then we arrive at the $N \times N$ system

$$\mathbf{A}\mathbf{y} + \mathbf{F}(\mathbf{y}; \theta) = \mathbf{b}, \tag{4.10}$$

which is a special case of (4.2). In this model problem, the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ do not depend on the parameter θ. For later reference, we note that the matrix $\mathbf{A}$, the function $\mathbf{F}$, and the vector $\mathbf{b}$ are given by

$$\mathbf{A}_{ij} = a_h(\phi_j, \phi_i) \qquad i, j = 1, \ldots N, \tag{4.11a}$$

$$\mathbf{F}_i(\mathbf{y}; \theta) = F_h(\sum_{j=1}^{N} y_j \phi_j + \sum_{j=N+1}^{N+N_D} h(x_j)\phi_j, \phi_i; \theta), \qquad i = 1, \ldots N, \tag{4.11b}$$

$$\mathbf{b}_i = b_h(\phi_i) := a_h(\sum_{j=N+1}^{N+N_D} h(x_j)\phi_j, \phi_i), \qquad i = 1, \ldots N. \tag{4.11c}$$

4.2.2 The Stokes Equations on Parameterized Domains

As our second model problem we consider the Stokes equations posed on a family of parameterized domains $\Omega(\theta) \subset \mathbb{R}^2$, where $\theta \in \Theta \subset \mathbb{R}^p$. Since our numerical examples are 2D problems we describe the approach for parameterized domains in $\mathbb{R}^2$. However, everything can be easily generalized to the Stokes equations on parameterized domains in $\mathbb{R}^3$. The boundary $\partial\Omega = \Gamma_D \cup \Gamma_{\text{out}}$ is decomposed into an outflow boundary Γ_{out} and $\Gamma_D = \partial\Omega \setminus \Gamma_{\text{out}}$. We assume that the parameterized domains $\Omega(\theta)$ can be mapped onto a reference domain $\widetilde{\Omega} \subset \mathbb{R}^2$. That is we assume that for each $\theta \in \Theta$ there exists a diffeomorphism $\Phi(\cdot; \theta)$ with

$$\Omega(\theta) = \Phi(\widetilde{\Omega}; \theta). \tag{4.12}$$

The Stokes equations for the velocity u and the pressure p are

$$-\nu\Delta u(x) + \nabla p(x) = f(x), \qquad \text{in } \Omega(\theta) \tag{4.13a}$$

$$\nabla \cdot u(x) = 0, \qquad \text{in } \Omega(\theta) \tag{4.13b}$$

$$u(x) = h(x), \qquad \text{on } \Gamma_D(\theta) \tag{4.13c}$$

$$(\nu\nabla u(x) - p(x)) \cdot n(x) = 0, \qquad \text{on } \Gamma_{\text{out}}(\theta), \tag{4.13d}$$

where $f \in (L^2(\Omega(\theta)))^2$. The weak form of (4.13) is given as follows: Find $u \in (H^1(\Omega(\theta)))^2$ with $u = h$ on $\Gamma_D(\theta)$ and $p \in L^2(\Omega(\theta))$ such that

$$\int_{\Omega(\theta)} \nu \nabla u(x) : \nabla v(x) - \int_{\Omega(\theta)} \nabla \cdot v(x) p(x) = \int_{\Omega(\theta)} f(x) v(x), \tag{4.14a}$$

$$- \int_{\Omega(\theta)} \nabla \cdot u(x) q(x) = 0, \tag{4.14b}$$

for all $v \in \{\phi \in (H^1(\Omega(\theta)))^2 \ : \ \phi = 0 \text{ on } \Gamma_D(\theta)\}$ and $q \in L^2(\Omega(\theta))$. Existence results for the Stokes equations can be found, e.g., in [12, 13].

We approximate (4.14) using Taylor-Hood P2-P1 finite elements [10]. We triangulate the reference domain $\widetilde{\Omega}$ and use (4.12). Let N_v be the number of velocity nodes in $\widetilde{\Omega} \cup \widetilde{\Gamma}_{\text{out}}$ and let N_p be the number of pressure nodes in $\widetilde{\Omega}$. If the piecewise quadratic basis functions for the velocities on the reference domain are $\widetilde{\phi}_j$, $j = 1, \ldots, N_v$, and the piecewise linear basis functions for the pressure on the reference domain are $\widetilde{\psi}_j$, $j = 1, \ldots, N_p$, then the basis functions for velocities and pressure on the domain $\Omega(\theta)$ are

$$\begin{aligned} \phi_j(\cdot;\theta) &= \widetilde{\phi}_j \circ \Phi^{-1}(\cdot;\theta), \quad j = 1, \ldots, N_v, \\ \psi_j(\cdot;\theta) &= \widetilde{\psi}_j \circ \Phi^{-1}(\cdot;\theta), \quad j = 1, \ldots, N_p. \end{aligned} \tag{4.15}$$

The Taylor-Hood P2-P1 finite element discretization of (4.14) leads to

$$\mathbf{S}(\theta)\mathbf{y} = \mathbf{b}(\theta), \tag{4.16}$$

where

$$\mathbf{S}(\theta) = \begin{pmatrix} \mathbf{A}(\theta) & 0 & \mathbf{B}^{(1)}(\theta)^T \\ 0 & \mathbf{A}(\theta) & \mathbf{B}^{(2)}(\theta)^T \\ \mathbf{B}^{(1)}(\theta) & \mathbf{B}^{(2)}(\theta) & 0 \end{pmatrix}, \ \mathbf{y} = \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{p} \end{pmatrix}, \ \mathbf{b}(\theta) = \begin{pmatrix} \mathbf{b}^{(1)}(\theta) \\ \mathbf{b}^{(2)}(\theta) \\ \mathbf{b}^{(3)}(\theta) \end{pmatrix}, \tag{4.17}$$

with

$$\mathbf{A}(\theta)_{ij} = \int_{\Omega(\theta)} \nu \nabla \phi_i^T \nabla \phi_j \, dx, \qquad 1 \le i, j \le N_v,$$

and

$$\mathbf{B}^{(1)}(\theta)_{ij} = - \int_{\Omega(\theta)} \frac{\partial \phi_j}{\partial x_1} \psi_i \, dx, \quad \mathbf{B}^{(2)}(\theta)_{ij} = - \int_{\Omega(\theta)} \frac{\partial \phi_j}{\partial x_2} \psi_i \, dx,$$

$1 \le j \le N_v$, $1 \le i \le N_p$.

We use the integral transformation as well as the structure (4.12) of the basis functions to compute

$$\mathbf{A}(\theta)_{ij} = \int_{\widetilde{\Omega}} \nu \, \widetilde{\nabla} \widetilde{\phi}_i(\widetilde{x})^T (D\Phi(\widetilde{x};\theta))^{-1} (D\Phi(\widetilde{x};\theta))^{-T} \widetilde{\nabla} \widetilde{\phi}_j(\widetilde{x}) \, |\det(D\Phi(\widetilde{x};\theta))| \, d\widetilde{x}$$

for $1 \le i, j \le N_v$, and

$$\begin{pmatrix} \mathbf{B}^{(1)}(\theta)_{ij} \\ \mathbf{B}^{(2)}(\theta)_{ij} \end{pmatrix} = - \int_{\widetilde{\Omega}} (D\Phi(\widetilde{x};\theta))^{-T} \widetilde{\nabla} \widetilde{\phi}_j(\widetilde{x}) \, \widetilde{\psi}_i(\widetilde{x}) \, |\det(D\Phi(\widetilde{x};\theta))| \, d\widetilde{x},$$

for $1 \le j \le N_v$, $1 \le i \le N_p$.

Finally, we approximate the integrals by a quadrature rule with nodes $\widetilde{x}_i$ and weights ω_i, $i = 1, \ldots, M$. To keep the presentation simple, we assume that the same quadrature rule is used for all integrals. If we define functions $\mathbf{g}_k : \Theta \to \mathbb{R}^M$, $k = 1, \ldots, 7$, component-wise as follows

$$\begin{pmatrix} (\mathbf{g}_1(\theta))_\ell & (\mathbf{g}_2(\theta))_\ell \\ (\mathbf{g}_2(\theta))_\ell & (\mathbf{g}_3(\theta))_\ell \end{pmatrix} = \nu\, \omega_\ell\, (D\Phi(\widetilde{x}_\ell;\theta))^{-1} (D\Phi(\widetilde{x}_\ell;\theta))^{-T}\, |\det(D\Phi(\widetilde{x}_\ell;\theta))|,$$
$$\begin{pmatrix} (\mathbf{g}_4(\theta))_\ell & (\mathbf{g}_5(\theta))_\ell \\ (\mathbf{g}_6(\theta))_\ell & (\mathbf{g}_7(\theta))_\ell \end{pmatrix} = \omega_\ell\, (D\Phi(\widetilde{x}_\ell;\theta))^{-T}\, |\det(D\Phi(\widetilde{x}_\ell;\theta))|,$$

then, if the integrals are replaced by quadrature, the matrices in the Stokes system can be written as

$$\mathbf{A}(\theta)_{ij} = \sum_{\ell=1}^{M} \widetilde{\nabla}\widetilde{\phi}_i(\widetilde{x}_\ell)^T \begin{pmatrix} (\mathbf{g}_1(\theta))_\ell & (\mathbf{g}_2(\theta))_\ell \\ (\mathbf{g}_2(\theta))_\ell & (\mathbf{g}_3(\theta))_\ell \end{pmatrix} \widetilde{\nabla}\widetilde{\phi}_j(\widetilde{x}_\ell), \quad 1 \le i, j \le N_v$$
$$\begin{pmatrix} \mathbf{B}^{(1)}(\theta)_{ij} \\ \mathbf{B}^{(2)}(\theta)_{ij} \end{pmatrix} = \sum_{\ell=1}^{M} \widetilde{\psi}_i(\widetilde{x}_\ell) \begin{pmatrix} (\mathbf{g}_4(\theta))_\ell & (\mathbf{g}_5(\theta))_\ell \\ (\mathbf{g}_6(\theta))_\ell & (\mathbf{g}_7(\theta))_\ell \end{pmatrix} \widetilde{\nabla}\widetilde{\phi}_j(\widetilde{x}_\ell), \quad 1 \le j \le N_v,\ 1 \le i \le N_p.$$

If we insert this representation into (4.17), then

$$\mathbf{S}(\theta) = \sum_{\ell=1}^{M} \sum_{k=1}^{7} (\mathbf{g}_k)_\ell(\theta) \mathbf{S}_{\ell k}. \tag{4.18}$$

Similarly, if we replace the integrals in the right hand side vectors

$$\mathbf{b}^{(k)}(\theta)_i = \int_{\Omega(\theta)} f_k(x)\phi_i(x)\, dx = \int_{\widetilde{\Omega}} f_k(\Phi(\widetilde{x};\theta))\, \widetilde{\phi}_i(\widetilde{x})\, |\det(D\Phi(\widetilde{x};\theta))|\, d\widetilde{x}, \quad k = 1, 2,$$

by quadrature rules, then

$$\mathbf{b}^{(k)}(\theta)_i = \sum_{\ell=1}^{M} \widetilde{\phi}_i(\widetilde{x}_\ell)\, (\mathbf{g}_{7+k}(\theta))_\ell, \quad k = 1, 2,$$

where $(\mathbf{g}_{7+k}(\theta))_\ell = \omega_\ell\, f_k(\Phi(\widetilde{x}_\ell;\theta))\, |\det(D\Phi(\widetilde{x}_\ell;\theta))|$, $k = 1, 2$.

4.3 Projection Based Reduced Order Models

4.3.1 Generating the Reduced Order Model Subspaces

The computation of the matrices $\mathbf{V}_\ell, \mathbf{V}_r \in \mathbb{R}^{N \times n}$ is crucial for the accuracy of the resulting reduced order model and involves some sort of sampling of the solutions to the full order model. Commonly used methods to generate these matrices include the greedy algorithm (see, e.g., [4, 6, 24, 29]), proper orthogonal decomposition (POD) (see, e.g., [19]), and, for time dependent linear problems, balanced POD (see, e.g., [1, 21, 28]). Since emphasis of this paper is the efficient evaluation of the reduced order model (4.2) using DEIM, it does not matter how $\mathbf{V}_\ell, \mathbf{V}_r \in \mathbb{R}^{N \times n}$ have been generated. We assume these matrices have been generated by a suitable method. In

our numerical examples, we generate $\mathbf{V} = \mathbf{V}_\ell = \mathbf{V}_r \in \mathbb{R}^{N\times n}$ using a simple sampling strategy and proper orthogonal decomposition. This often results in good reduced order models, although more sophisticated sampling strategies might have provided equally good reduced order models using fewer samples.

Since we will refer to the proper orthogonal decomposition (POD) later, we provide a few details on this method. First by POD we mean the construction of a k dimensional subspace that best approximates given samples $\mathbf{s}_1, \ldots, \mathbf{s}_K$. Thus, selection of these samples is not part of POD. We assume $\mathbf{s}_1, \ldots, \mathbf{s}_K \in \mathbb{R}^N$, but in general these samples could be vectors in a Hilbert space. See, e.g., [19]. Given the samples $\mathbf{s}_1, \ldots, \mathbf{s}_K$ the POD successively computes vectors $\mathbf{v}_1, \ldots, \mathbf{v}_k$ as the solution of

$$\text{minimize} \sum_{j=1}^{K} \|\mathbf{s}_j - \sum_{i=1}^{\ell} \mathbf{v}_i \, \mathbf{v}_i^T \mathbf{s}_j\|_2^2 \tag{4.19a}$$

$$\text{subject to } \mathbf{v}_i^T \mathbf{v}_j = \delta_{ij}, \quad i, j = 1, \ldots, k, \tag{4.19b}$$

where δ_{ij} is the Kronecker delta, or in matrix notation

$$\text{minimize } \|\mathbf{S} - \mathbf{V}_k \, \mathbf{V}_k^T \mathbf{S}\|_F^2 \tag{4.20a}$$

$$\text{subject to } \mathbf{V}_k^T \mathbf{V}_k = \mathbf{I}_k, \tag{4.20b}$$

where $\mathbf{I}_k \in \mathbb{R}^{k\times k}$ is the identity. Is is well known that the solution can be computed via the singular value decomposition (SVD) of $\mathbf{S}$, $\mathbf{S} = \mathbf{V}\Sigma\mathbf{W}^T$. In fact, since $\mathbf{W}$ is orthogonal, $\|\mathbf{S} - \mathbf{V}_k \, \mathbf{V}_k^T \mathbf{S}\|_F^2 = \|\mathbf{V}\Sigma - \mathbf{V}_k \, \mathbf{V}_k^T \mathbf{V}\Sigma\|_F^2$. If $\mathbf{V}_k \in \mathbb{R}^{N\times k}$ is submatrix consisting of the first k columns of $\mathbf{V} \in \mathbb{R}^{N\times N}$, and if $\Sigma_k \in \mathbb{R}^{N\times K}$ is obtained by replacing the singular values $\sigma_{k+1}, \sigma_{k+2}, \ldots$ in $\Sigma \in \mathbb{R}^{N\times K}$ by zero, then

$$\begin{aligned} \|\mathbf{S} - \mathbf{V}_k \, \mathbf{V}_k^T \mathbf{S}\|_F^2 &= \|\mathbf{V}\Sigma - \mathbf{V}_k \, \mathbf{V}_k^T \mathbf{V}\Sigma\|_F^2 = \|\mathbf{V}\Sigma - \mathbf{V}\Sigma_k\|_F^2 = \|\Sigma - \Sigma_k\|_F^2 \\ &= \sum_{j=k+1}^{\min\{K,N\}} \sigma_j^2. \end{aligned} \tag{4.21}$$

Algorithm 4.1 (POD)

Input: Samples $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_K) \in \mathbb{R}^{N\times K}$ and tolerance $\tau > 0$.
Output: $\mathbf{V}_k = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{N\times k}$.

1. Compute the singular value decomposition $\mathbf{S} = \mathbf{V}\Sigma\mathbf{W}^T$.
2. Find smallest index k such that the singular values satisfy $\sigma_{k+1} < \tau\sigma_1$.
3. Return the first k columns $\mathbf{V}_k = (\mathbf{v}_1, \ldots, \mathbf{v}_k) \in \mathbb{R}^{N\times k}$ of $\mathbf{V}$.

Given the bound (4.21), the index k is often chosen to be the smallest index such that $\sum_{j=k+1}^{\min\{K,N\}} \sigma_j^2 < \tau$. This requires computation of all singular values, which can be expensive. Therefore, we use the smallest index k such that $\sigma_{k+1} < \tau\sigma_1$. This alternative provides a bound on the relative error in the two-norm: $\|\mathbf{S} - \mathbf{V}_k \, \mathbf{V}_k^T \mathbf{S}\|_2 \leq$

$\tau\|\mathbf{S}\|_2$. In our examples, the matrix of samples $\mathbf{S} \in \mathbb{R}^{N\times K}$ satisfies $K \ll N$ and we compute the so-called economy-sized SVD. In the large scale setting, we can use an iterative method (e.g. ARPACK) to compute just the largest k singular values without computing all of them.

We note that often the snapshots do not have to be approximated in the Euclidean norm sense as in (4.19), but instead using a weighted dot product $\mathbf{v}_i^T\mathbf{M}\mathbf{s}_j$ and corresponding norm $\|\mathbf{s}\|_{\mathbf{M}}^2 = \mathbf{s}^T\mathbf{M}\mathbf{s}$, respectively, where $\mathbf{M} \in \mathbb{R}^{N\times N}$ is a symmetric positive definite matrix. This is for example the case when the snapshots $s_j(x) = \sum_{i=1}^N \mathbf{s}_{ij}\phi_i(x)$ belong to the Hilbert space $H_0^1(\Omega)$. In this case, $\mathbf{M}$ is the stiffness matrix. See, e.g., [19]. This can be accomplished by modifying the SVD.

4.3.2 The DEIM

In this section we review the DEIM to approximate a function $\mathbf{G} : \mathbb{R}^k \to \mathbb{R}^N$. We require a subspace with basis $\{\mathbf{u}_1, \ldots, \mathbf{u}_m\}$ such that $\mathbf{G}(\mathbf{z})$ is approximately contained in span$\{\mathbf{u}_1, \ldots, \mathbf{u}_m\}$ for the arguments $\mathbf{z}$ of interest. Typically, one samples $\mathbf{G}$ and then applies the POD to the samples to obtain an orthonormal basis $\{\mathbf{u}_1, \ldots, \mathbf{u}_m\}$. To obtain a computationally efficient DEIM approximation of $\mathbf{G}$ one needs that $m \ll N$.

The DEIM [7] can be viewed as variant of the empirical interpolation method of [2] (see also [14]) applied to large scale finite dimensional systems.

The DEIM computes indices $p_1, \ldots, p_m$ in $\{1, \ldots, N\}$ and an approximation $\widehat{\mathbf{G}} : \mathbb{R}^k \to \mathbb{R}^N$ of the function $\mathbf{G}$ which satisfies

$$\widehat{\mathbf{G}}_{p_i}(\mathbf{z}) = \mathbf{G}_{p_i}(\mathbf{z}) \quad \text{for } i = 1, \ldots, m \tag{4.22}$$

moreover, for each $\mathbf{z}$ the computation of $\widehat{\mathbf{G}}(\mathbf{z})$ only requires the m components $\mathbf{G}_{p_1}(\mathbf{z}), \ldots, \mathbf{G}_{p_m}(\mathbf{z})$ of the original function $\mathbf{G}$. More specifically, if $\mathbf{e}_i$ is the ith unit vector in $\mathbb{R}^N$, $\mathbf{P} = [\mathbf{e}_{p_1}, \ldots, \mathbf{e}_{p_m}] \in \mathbb{R}^{N\times m}$ is the submatrix of the identity obtained by extracting the columns $p_1, \ldots, p_m$, and $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$, then the DEIM approximation of $\mathbf{G}$ is

$$\widehat{\mathbf{G}} = \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{G} : \mathbb{R}^k \to \mathbb{R}^N.$$

Clearly, $\mathbf{P}^T\widehat{\mathbf{G}} = \mathbf{P}^T\mathbf{G}$, which verifies the interpolation property (4.22), and $\mathbf{P}^T\mathbf{G} = (\mathbf{G}_{p_1}, \ldots, \mathbf{G}_{p_m})^T$, which means that only the components $p_1, \ldots, p_m$ of $\mathbf{G}$ are needed to compute the approximation $\widehat{\mathbf{G}}$. This is the source of the complexity reduction provided by the DEIM.

Before we review how DEIM computes the indices $p_1, \ldots, p_m$ and the DEIM error bounds, we discuss when the DEIM approximation is useful. For example, in model reduction we have to evaluate the nonlinearity $(\widehat{\mathbf{y}}; \theta) \mapsto \mathbf{V}_\ell^T\mathbf{F}(\overline{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}; \theta)$, where $\mathbf{F} : \mathbb{R}^N \times \mathbb{R}^p \to \mathbb{R}^N$ and $\mathbf{V}_\ell$ and $\mathbf{V}_r \in \mathbb{R}^{N\times n}$ with $n \ll N$. As we have mentioned, this requires the computation of $\overline{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}$, the evaluation of the nonlinearity $\mathbf{F}(\overline{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}; \theta)$ and the projection $\mathbf{V}_\ell^T\mathbf{F}(\overline{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}; \theta)$. All of these operations depend on the size N of the full system and, therefore, the evaluation of the reduced order model is almost expensive as that of the full order model. The complexity of the reduced order model can be made independent of the full order problem size N using the

DEIM approximation. If we compute a DEIM approximation

$$\widehat{\mathbf{F}} = \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{F},$$

then we can approximate the nonlinearity $\mathbf{V}_\ell^T\mathbf{F}(\overline{\mathbf{y}}+\mathbf{V}_r\widehat{\mathbf{y}};\theta)$ by

$$\mathbf{V}_\ell^T\widehat{\mathbf{F}}(\overline{\mathbf{y}}+\mathbf{V}_r\widehat{\mathbf{y}};\theta) = \left(\mathbf{V}_\ell^T\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\right)\mathbf{P}^T\mathbf{F}(\overline{\mathbf{y}}+\mathbf{V}_r\widehat{\mathbf{y}};\theta). \tag{4.23}$$

Typically in problems arising from spatial discretization of a PDE, the ith component of $\mathbf{F}$ depends only on a few components of $\mathbf{y}$. Hence, the evaluation of the m components $\mathbf{P}^T\mathbf{F}(\overline{\mathbf{y}}+\mathbf{V}_r\widehat{\mathbf{y}};\theta)$ of the nonlinearity requires only a few, say $O(m)$ components of $\overline{\mathbf{y}}+\mathbf{V}_r\widehat{\mathbf{y}}$. Hence we do not need to compute $\overline{\mathbf{y}}+\mathbf{V}_r\widehat{\mathbf{y}}$ at a cost of $2Nn+N$ flops (we count multiplication and addition as a flop), but only some components of this vector at a cost of $O(mn)$. Furthermore, the matrix $\mathbf{V}_\ell^T\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1} \in \mathbb{R}^{n\times m}$ can be precomputed so that afterwards the evaluation of $(\widehat{\mathbf{y}};\theta) \mapsto \mathbf{V}_\ell^T\widehat{\mathbf{F}}(\overline{\mathbf{y}}+\mathbf{V}_r\widehat{\mathbf{y}};\theta)$ defined in (4.23) requires only $O(mn)$ operations. In finite difference approximations, the ith component of the nonlinearity $\mathbf{F}$ typically depends only on the ith component of the argument $\mathbf{y}$. Finite difference approximations are used, e.g., in the examples in [7,8]. If finite element methods are used, the ith component of the nonlinearity $\mathbf{F}$ depends on more than the ith component of the argument. The dependency of the ith component of $\mathbf{F}$ on the components of the argument depends on the polynomial order used in the finite element method, on the mesh, and also in what stage of the finite element assembly process the DEIM is applied. We will explore this in Sec. 4.4.

Algorithm 4.2 (DEIM)

Input: Linearly independent vectors $\mathbf{u}_1,\ldots,\mathbf{u}_m$.
Output: Indices $p_1,\ldots,p_m$.

1. $[\rho, p_1] = \max\{|\mathbf{u}_1|\}$
2. Set $\mathbf{U} = [\mathbf{u}_1]$, $\mathbf{P} = [\mathbf{e}_{p_1}]$, $\mathbf{p} = [p_1]$
3. For $i = 2,\ldots,m$ do
   a. Solve $(\mathbf{P}^T\mathbf{U})\mathbf{c} = \mathbf{P}^T\mathbf{u}_i$ for $\mathbf{c}$
   b. $\mathbf{r}_i = \mathbf{u}_i - \mathbf{U}\mathbf{c}$
   c. $[\rho, p_i] = \max\{|\mathbf{r}_i|\}$
   d. Update $\mathbf{U} = [\mathbf{U} \quad \mathbf{u}_i]$, $\mathbf{P} = [\mathbf{P} \quad \mathbf{e}_{p_i}]$, $\mathbf{p} = [\mathbf{p}^T \quad p_i]^T$

We next state an error estimate from [7] for the DEIM approximation

$$\widehat{\mathbf{G}} = \mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{G}$$

to $\mathbf{G}$. If $\mathbf{U} \in \mathbb{R}^{N\times m}$ has ortho-normal columns, then

$$\|\mathbf{G} - \widehat{\mathbf{G}}\|_2 \le \|(\mathbf{P}^T\mathbf{U})^{-1}\|_2\|(\mathbf{I}-\mathbf{U}\mathbf{U}^T)\mathbf{G}\|_2\,. \tag{4.24}$$

This result indicates that very little accuracy is lost when the orthogonal projection of POD is replaced by the DEIM interpolatory projection so long as $\|(\mathbf{P}^T\mathbf{U})^{-1}\|_2$ is of modest size. In practice, we simply compute this quantity and use it as an a-

posteriori estimate. The greedy DEIM index selection actually limits the growth of $\|(\mathbf{P}^T\mathbf{U})^{-1}\|_2$ and typically it has remained on the order of 100 or less in all of the examples we have considered. Finally, we must emphasize that the DEIM does not improve the accuracy of the POD reduced model. The sole benefit of the DEIM is to greatly reduce the complexity of evaluating the reduced model.

4.4 Evaluation of Nonlinear Functions Arising in Finite Element Methods Using DEIM

We study the application of the DEIM for the evaluation of nonlinear terms in finite element models. As noted in Sect. 4.3.2, the main issue here is the computational complexity of the DEIM reduced model. It depends on how may components of the argument influence a component of the nonlinearity, and it is determined by the finite elements used. We present two ways of applying the DEIM. One approach applies DEIM to the assembled form of the nonlinear term, the other approach, originally suggested by Dedden et al. [9, 33], to the unassembled form.

We use the semilinear advection diffusion reaction equation from Sect. 4.2.1 and continuous finite element approximations. However, the approaches can easily be extended to other equations and discontinuous Galerkin methods.

4.4.1 The Reduced Order Model

We consider the finite element discretization of the semilinear advection diffusion reaction equation discussed in Sect. 4.2.1. To simplify our notation, we assume that the boundary data $h(x) = 0$ in (4.3). The finite element discretization of (4.3) leads to the $N \times N$ system of nonlinear equations

$$\mathbf{A}\mathbf{y} + \mathbf{F}(\mathbf{y}; \theta) = \mathbf{b}, \tag{4.25}$$

where $\mathbf{A} \in \mathbb{R}^{N\times N}$ and $\mathbf{F} : \mathbb{R}^N \times \mathbb{R}^p \to \mathbb{R}^N$ are given by (4.11). Note that since $h(x) = 0$, the vector $\mathbf{b} = \mathbf{0} \in \mathbb{R}^N$.

Assume we have generated $\mathbf{V}_\ell$ and $\mathbf{V}_r \in \mathbb{R}^{N\times n}$ with $n \ll N$. Then the reduced order model of (4.25) is

$$\mathbf{V}_\ell^T \mathbf{A}(\bar{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}) + \mathbf{V}_\ell^T \mathbf{F}(\bar{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}; \theta) = \mathbf{V}_\ell^T \mathbf{b}. \tag{4.26}$$

As we have mentioned before, $\mathbf{V}_\ell^T\mathbf{A}\mathbf{V}_r$, $\mathbf{V}_\ell^T\mathbf{A}\bar{\mathbf{y}}$ and $\mathbf{V}_\ell^T\mathbf{b}$ can be precomputed, but since the nonlinearity depends on $\widehat{\mathbf{y}}$ and θ the term $\mathbf{V}_\ell^T\mathbf{F}(\bar{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}; \theta)$ needs to be evaluated whenever $\widehat{\mathbf{y}}$ or θ changes, and the cost of evaluating this nonlinearity still depends on the size N of the full order model.

To reduce the complexity of the nonlinear term, we apply the DEIM. The DEIM reduced order model is given by

$$\mathbf{V}_\ell^T \mathbf{A}(\bar{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}) + \left(\mathbf{V}_\ell^T\mathbf{U}(\mathbf{P}^T\mathbf{U})^{-1}\right)\mathbf{P}^T\mathbf{F}(\bar{\mathbf{y}} + \mathbf{V}_r\widehat{\mathbf{y}}; \theta) = \mathbf{V}_\ell^T\mathbf{b}. \tag{4.27}$$

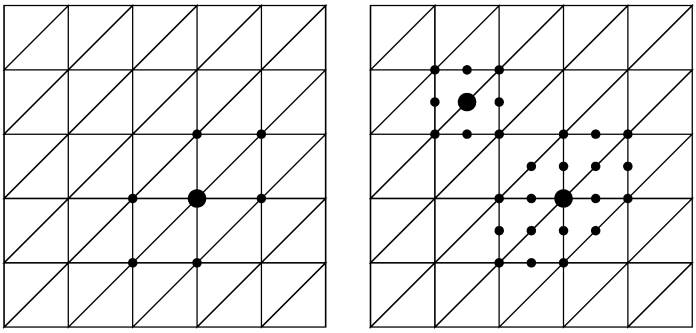

Fig. 4.1. Left plot: Piecewise linear finite elements on triangles. If the DEIM index p_i corresponds to the vertex indicated by the large dot, then the p_ith component of the nonlinear function depends on the seven adjacent vertices indicated by dots. Right plot: Piecewise quadratic finite elements on triangles. If the DEIM index p_i corresponds to the vertex indicated by the large dot, then the p_ith component of the nonlinear function depends on nineteen adjacent nodes indicated by dots. If the DEIM index p_i corresponds to the midpoint indicated by the large dot, then the p_ith component of the nonlinear function depends on nine adjacent nodes indicated by dots

The $n \times m$ matrix $\mathbf{V}_\ell^T \mathbf{U}(\mathbf{P}^T \mathbf{U})^{-1}$ can be precomputed once. We still need to study the complexity of the evaluation of the nonlinearity

$$\mathbf{P}^T \mathbf{F}(\bar{\mathbf{y}} + \mathbf{V}_r \widehat{\mathbf{y}}; \theta) = \Big(\mathbf{F}_{p_1}(\bar{\mathbf{y}} + \mathbf{V}_r \widehat{\mathbf{y}}; \theta), \ldots, \mathbf{F}_{p_m}(\bar{\mathbf{y}} + \mathbf{V}_r \widehat{\mathbf{y}}; \theta) \Big)^T \in \mathbb{R}^m$$

in the DEIM reduced model (4.27). The ith component $\mathbf{F}_i$ of the nonlinearity depends on all components $\mathbf{y}_j$ for which the intersection of the support of basis functions ϕ_i and ϕ_j does not have measure zero. See (4.11b).

This is illustrated in Fig. 4.1 for piecewise linear (left plot) and piecewise quadratic (right plot) basis functions ϕ_i on triangles. In the case of piecewise linear basis functions, there are $N = 36$ degrees of freedom, which correspond to the vertices. If the DEIM index p_i corresponds to the vertex indicated by the large dot, then the p_ith component of $\mathbf{F}$ depends on seven components of $\mathbf{y}$, which corresponds to the vertices indicated by dots. If piecewise quadratic basis functions are used, then there are $N = 121$ degrees of freedom, which correspond to the vertices and edge midpoints. If the DEIM index p_i corresponds to the vertex indicated by the large dot, then the p_ith component of $\mathbf{F}$ depends on nineteen components of $\mathbf{y}$, which corresponds to the vertices and edge midpoints indicated by dots in the bottom right part of the right plot in Fig. 4.1. On the other hand, if the DEIM index p_i corresponds to a midpoint, then this midpoint is shared by only two triangles, and the p_ith component of $\mathbf{F}$ depends on nine components of $\mathbf{y}$, which corresponds to the vertices and edge midpoints indicated by dots in the top left part of the right plot in Fig. 4.1.

An alternative DEIM reduced order model is obtained when we consider the unassembled nonlinearity. As we have mentioned earlier, this was first suggested and

explored by [9, 33]. Since $\int_\Omega = \sum_{e=1}^{n_e} \int_{\Omega_e}$, we can write (4.11b) as

$$\mathbf{F}_i(\mathbf{y};\theta) = \sum_{e=1}^{n_e} \int_{\Omega_e} f\big(\sum_{j=1}^{N} y_j\phi_j;\theta\big)\phi_i + \tau_e\,(\beta\cdot\nabla\phi_i)\,f\big(\sum_{j=1}^{N} y_j\phi_j;\theta\big)\,dx.$$

When the intersection of the supports of the basis functions ϕ_i and ϕ_j and of the element Ω_e has measure zero, the integral $\int_{\Omega_e} f\big(\sum_{j=1}^{N} y_j\phi_j;\theta\big)\phi_i + \tau_e\,(\beta\cdot\nabla\phi_i)\,f\big(\sum_{j=1}^{N} y_j\phi_j;\theta\big)\,dx$ is zero. Therefore for nodal basis functions, this integral can only be nonzero when the indices i and j correspond to nodes in $\overline{\Omega_e}$. For each of the n_e elements Ω_e we can compute n_p integrals

$$\mathbf{F}_i^e(\mathbf{y};\theta) = \int_{\Omega_e} f\big(\sum_{j=1}^{N} y_j\phi_j;\theta\big)\phi_i + \tau_e\,(\beta\cdot\nabla\phi_i)\,f\big(\sum_{j=1}^{N} y_j\phi_j;\theta\big)\,dx, \tag{4.28a}$$

where n_p is the number of degrees of freedom per element and the indices i corresponds to nodes in the element $\overline{\Omega_e}$. This gives a function $\mathbf{F}^e(\mathbf{y};\theta) : \mathbb{R}^{n_e n_p} \times \mathbb{R}^p \to \mathbb{R}^{n_e n_p}$. Then we can assemble the element information into the global vector of unknowns $\mathbf{F}$. This can be expressed as

$$\mathbf{F}(\mathbf{y};\theta) = \mathbf{Q}\,\mathbf{F}^e(\mathbf{y};\theta) \tag{4.28b}$$

where $\mathbf{Q} \in \mathbb{R}^{N\times(n_e n_p)}$. The size of the unassembled nonlinearity $\mathbf{F}^e$ is larger than that of the assembled one $\mathbf{F}$. If the ith component of the unassembled nonlinearity belongs to element Ω_e, then $\mathbf{F}_i^e$ only depends on the unknowns y_j with indices j corresponding to nodes in the element Ω_e, see Fig. 4.2. Consequently, a component

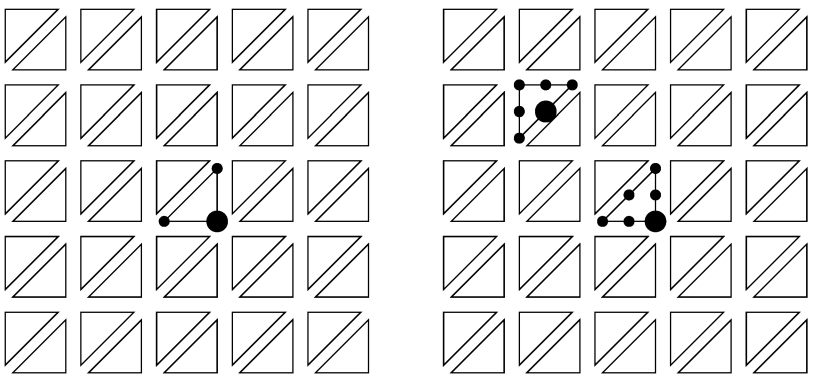

Fig. 4.2. If DEIM is applied to unassembled piecewise linear elements, then the p_i-th component of the unassembled nonlinearity only depends on values at nodes in the element that contains the node p_i. Left plot: For piecewise linear elements on triangles, the p_i-th component of the unassembled nonlinearity only depends on the values at the three vertices, indicated by dots, of one triangle. Right plot: For piecewise quadratic elements on triangles, the p_i-th component of the unassembled nonlinearity only depends on the values at the vertices and edge midpoints, indicated by dots, of one triangle

of unassembled nonlinearity depends on fewer components than a component of assembled nonlinearity does.

The reduced order model (4.26) can now be written as

$$\mathbf{V}_\ell^T \mathbf{A}(\bar{\mathbf{y}} + \mathbf{V}_r \widehat{\mathbf{y}}) + \mathbf{V}_\ell^T \mathbf{Q}\mathbf{F}^e(\bar{\mathbf{y}} + \mathbf{V}_r \widehat{\mathbf{y}}; \theta) = \mathbf{V}_\ell^T \mathbf{b}. \tag{4.29}$$

We can apply DEIM to the unassembled nonlinearity. Let the columns of $\mathbf{U}^e = [\mathbf{u}_1^e, \ldots, \mathbf{u}_{m^e}^e]$ be a basis of a subspace that approximately contains $\mathbf{F}^e(\mathbf{y}; \theta)$ for the arguments $\mathbf{y}$ and θ of interest. The DEIM approximation of the unassembled nonlinearity is given by

$$\widehat{\mathbf{F}}^e(\mathbf{y}; \theta) = \mathbf{U}^e \left((\mathbf{P}^e)^T (\mathbf{U}^e) \right)^{-1} (\mathbf{P}^e)^T \mathbf{F}^e(\mathbf{y}; \theta).$$

Here $\mathbf{P}^e$ is the sub matrix of the identity generated using the indices $p_1^e, \ldots, p_{m^e}^e$ generated by the DEIM applied to $\mathbf{u}_1^e, \ldots, \mathbf{u}_{m^e}^e$.

If we insert this into (4.29) we arrive at the DEIM reduced order model

$$\mathbf{V}_\ell^T \mathbf{A}\mathbf{V}_r(\bar{\mathbf{y}} + \mathbf{V}_r \widehat{\mathbf{y}}) + \left(\mathbf{V}_\ell^T \mathbf{Q}\mathbf{U}^e \left((\mathbf{P}^e)^T (\mathbf{U}^e) \right)^{-1} \right) (\mathbf{P}^e)^T \mathbf{F}^e(\bar{\mathbf{y}} + \mathbf{V}_r \widehat{\mathbf{y}}; \theta) = \mathbf{V}_\ell^T \mathbf{b}. \tag{4.30}$$

The $n \times m^e$ matrix $\mathbf{V}_\ell^T \mathbf{Q}\mathbf{U}^e \left((\mathbf{P}^e)^T (\mathbf{U}^e) \right)^{-1}$ can be precomputed.

The advantage of the DEIM reduced order model (4.30) over (4.27) is that each component of the unassembled nonlinearity in (4.30) depends on fewer components of the argument than the nonlinearity in (4.27) does. Hence, if the dimension of the subspace $\mathscr{R}(\mathbf{U})$ containing the image of $\mathbf{F}$ is roughly equal to dimension of the subspace $\mathscr{R}(\mathbf{U}^e)$ containing the image of $\mathbf{F}^e$, i.e., if $m \approx m^e$, then the evaluation of (4.30) is computationally less expensive than that of (4.27). This is illustrated in Fig. 4.2. If a DEM point p_i corresponds to a node in a triangle, the the p_ith nonlinearity depends on all components of the argument that correspond to nodes in the triangle. The left plot in Fig. 4.2 illustrates this for one point when piecewise linear elements are used, whereas the right plot in Fig. 4.2 illustrates this when piecewise quadratic elements are used. Note, that if the unassembled form of the nonlinearity is used, the connectivity is the same no matter whether the DEIM point corresponds to an vertex or an edge midpoint.

The disadvantage of the DEIM reduced order model (4.30) compared to (4.27) is that the size of the unassembled nonlinearity $\mathbf{F}^e(\mathbf{y}; \theta)$ is significantly larger than the size N of the nonlinearity $\mathbf{F}(\mathbf{y}; \theta)$. The size $n_e n_p$ of the unassembled nonlinearity $\mathbf{F}^e(\mathbf{y}; \theta)$ now depends on the number n_e of elements and the number n_p of degrees of freedom n_p per element. For example, if we use piecewise linear basis functions on the mesh in the left plot in Fig. 4.1, there are $N = 36$ vertices, where as $n_e n_p = 150$. If we use piecewise quadratic basis functions on the mesh in the right plot in Fig. 4.1, then there are $N = 121$ degrees of freedom, whereas $n_e n_p = 300$. Since the vectors $\mathbf{u}_1, \ldots, \mathbf{u}_m$ and $\mathbf{u}_1^e, \ldots, \mathbf{u}_{m^e}^e$ are typically computed from a POD of samples of the nonlinearities $\mathbf{F}$ and $\mathbf{F}^e$, respectively, the computation of the vectors $\mathbf{u}_1^e, \ldots, \mathbf{u}_{m^e}^e$ is more expensive than the computation of $\mathbf{u}_1, \ldots, \mathbf{u}_m$. However, this computation is done in the off-line phase.

4.4.2 Numerical Examples

We apply DEIM reduced order models to approximate the semilinear advection diffusion reaction equation (4.3) with nonlinearity (4.4). The full order model is obtained using the SUPG stabilized finite elements reviewed in Sect. 4.2.1. The diffusivity is $\nu = 5 \cdot 10^{-6}$, and the parameters $C = 0.2$ and $D = 0.4$ in (4.4) are fixed and $\theta = (\ln(A), E)$ vary within $\Theta \equiv [5.00, 7.25] \times [0.05, 0.15] \subset \mathbb{R}^2$.

To construct the reduced basis matrices $\mathbf{V} = \mathbf{V}_\ell = \mathbf{V}_r$, we sample the finite element solution of (4.4) at 25 parameters. We denote these solutions by $\mathbf{y}(\theta_1), \ldots, \mathbf{y}(\theta_{25})$. We compute the mean $\overline{\mathbf{y}} = \frac{1}{25} \sum_{i=1}^{25} \mathbf{y}(\theta_i)$, and generate the reduced basis matrices $\mathbf{V} = \mathbf{V}_\ell = \mathbf{V}_r$ by applying the POD, Algorithm 4.1 to the samples $\mathbf{y}(\theta_1) - \overline{\mathbf{y}}, \ldots, \mathbf{y}(\theta_{25}) - \overline{\mathbf{y}}$ with tolerance $\tau = 10^{-4}$. To construct $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$ and $\mathbf{U}^e = [\mathbf{u}_1^e, \ldots, \mathbf{u}_{m^e}^e]$ for the DEIM approximation, we sample the nonlinearities $\mathbf{F}(\mathbf{y}(\theta); \theta)$ and $\mathbf{F}^e(\mathbf{y}(\theta); \theta)$, respectively, at the same parameters used to construct $\mathbf{V}$, and then we apply the POD with tolerance $\tau = 10^{-4}$ to obtain $\mathbf{U}$ and $\mathbf{U}^e$, respectively.

All computations in this subsections were done using Matlab on MacBook Air with 8GB of memory and 1.8 GHz Intel Core i5 processor. The nonlinear full order or reduced order models are solved using Newton's method. The linear systems in Newton's method are solved using the Matlab backslash command.

4.4.2.1 2D Example

We consider the domain $\Omega \subset \mathbb{R}^2$ shown in Fig. 4.3, taken from [3]. The Dirichlet boundary segments are $\Gamma_D = \{(0, x_2) : x_2 \in (0,2) \cup (2.75, 4.25) \cup (5,7)\}$ and the Dirichlet data h is specified in Fig, 4.3.

To study the computational cost of applying DEIM reduced order models we use three meshes, referred to as Mesh 1 to Mesh 3, of different sizes, and we use piecewise linear and quadratic elements. We compute an approximate solution of (4.3) at the parameter $(\ln(A), E) = (6.4, 0.11)$ not contained in the parameter sample. Figure 4.4 shows the triangulation corresponding to Mesh 2, of medium size, as well as the full order model solution of (4.3). (The reduced order model solutions are indistinguishable from the full order model solution.)

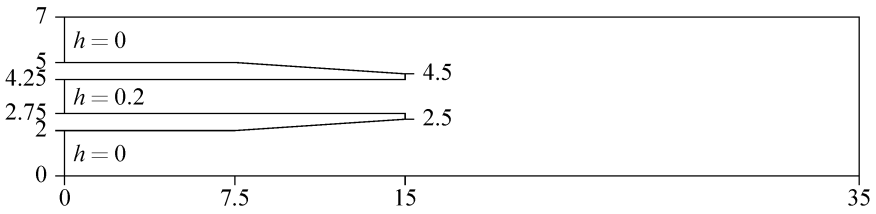

Fig. 4.3. 2D Example: The domain Ω with Dirichlet boundary segments $\Gamma_D = \{(0, x_2) : x_2 \in (0,2) \cup (2.75, 4.25) \cup (5,7)\}$ and Dirichlet data h

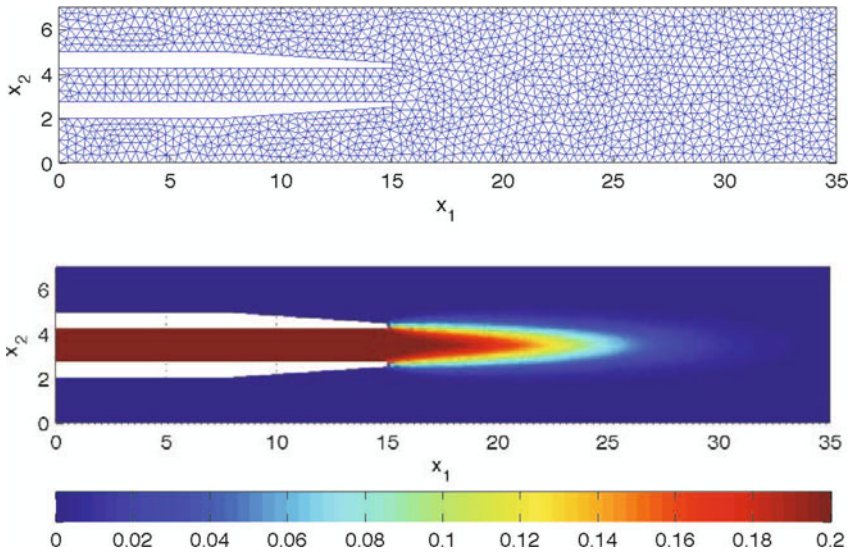

Fig. 4.4. 2D Example: A triangulation of the domain Ω (top plot) and solution of the advection diffusion reaction equation (4.3) with parameter $(\ln(A), E) = (6.4, 0.11)$ (bottom plot)

As we have described in the previous section, the complexity of evaluating DEIM reduced order models depends on the connectivity of the nodes in the finite element mesh. We illustrate this in Fig. 4.5 using Mesh 2. For four different configurations, we plot the triangles that are involved in the evaluation of the DEIM nonlinear term. More precisely, the degrees of freedom corresponding to all nodes in the red solid triangles are needed to evaluate the DEIM nonlinear term. The top two plots correspond to piecewise linear finite elements using the assembled (top plot) and unassembled (second from top plot) form of the nonlinearity. The top two plots in Fig. 4.5 correspond to the schematic plots on the left in Figs. 4.1 and 4.2, respectively.

The bottom two plots in Fig. 4.5 correspond to quadratic finite elements. If we look at the third plot from the top, which colors the triangles involved in the evaluation of the DEIM nonlinear functions (assembled form), then at most two triangles are connected. This means that all DEIM points in this case correspond to edge midpoints (see the right plot in Fig. 4.2). We observed the same for the computations on Mesh 1 and Mesh 3. The bottom plot in Fig. 4.5 corresponds to the unassembled form of the DEIM using quadratic finite elements. In this plot a few adjacent triangles are colored red, which simply means that the DEIM selected points that happen to correspond to nodes in adjacent triangles.

Table 4.1 summarizes the problem size for the different models for the three meshes and piecewise linear and quadratic finite elements. In Tables 4.1 to 4.3, DEIM refers to the DEIM reduced order model (4.27) obtained using the assembled form of the nonlinearity, whereas DEIM-u refers to the DEIM reduced order model (4.30) obtained using the unassembled form of the nonlinearity.

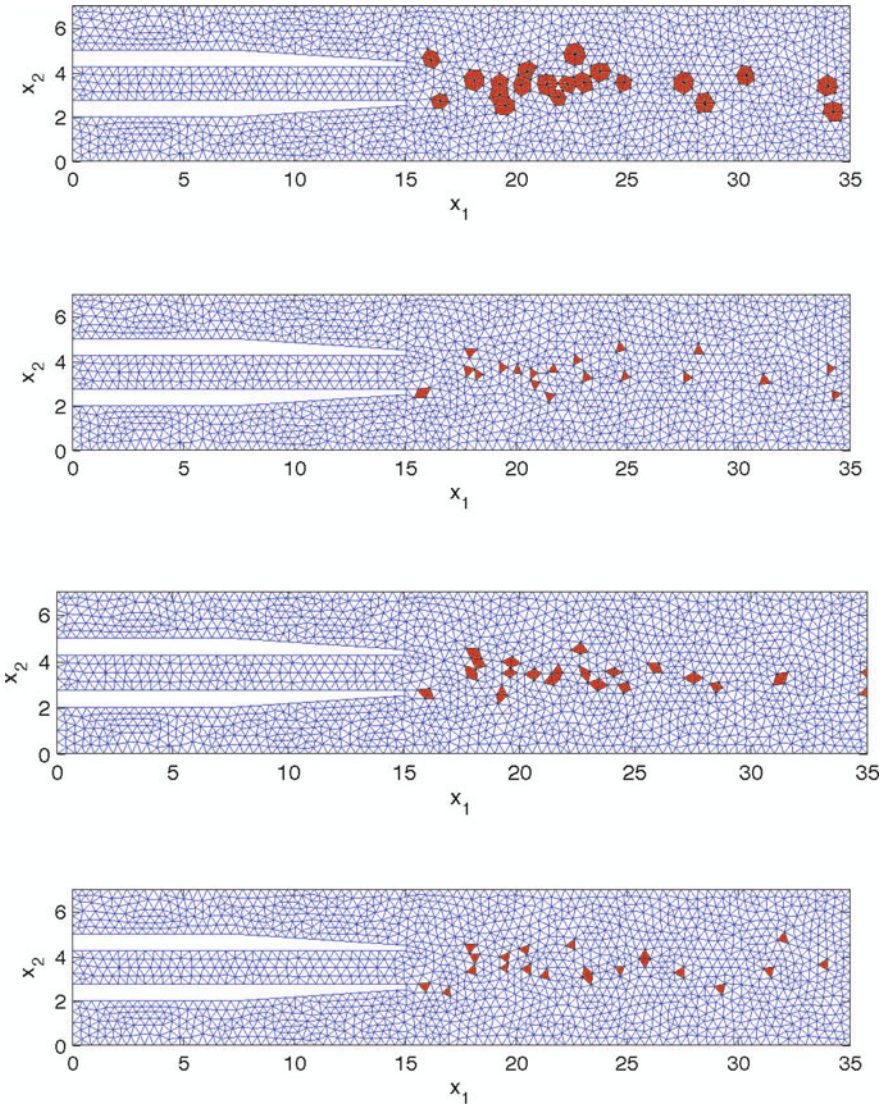

Fig. 4.5. 2D Example: The triangles that contain DEIM points are shown in solid red. The different plots correspond to different polynomial degree used in the FEM and application of the DEIM to the assembled or unassembled form of the nonlinearity

The computing times to evaluate the full and the various reduced order models are shown in Table 4.2. The nonlinear systems are solved using Newton's method and the computing times listed are for the Newton solve (and not for one Newton iteration). The number of Newton iterations required are shown in parenthesis. The

Table 4.1. 2D Example: The size N of the full order finite element system, the number of POD basis vectors n, the number of DEIM points m the number of nodes adjacent to DEIM points, the number of DEIM points m^e when the unassembled (DEIM-u) nonlinearity is used, and the number of nodes adjacent to DEIM points for piecewise linear and quadratic finite elements on three grids. The mesh in Fig. 4.5 correspond to grid number 2

Polynomial degree	$p=1$			$p=2$		
Mesh number	1	2	3	1	2	3
number of triangles	1,437	3,213	12,976	1,437	3,213	12,976
number of nodes N	825	1,768	6,813	3,089	6,751	26,604
number of POD basis vectors n	17	17	17	17	17	17
number of DEIM points m	20	20	21	21	21	21
number of nodes adjacent to DEIM pts.	107	139	166	165	174	186
number of DEIM-u points m^e	20	20	21	20	21	21
number of nodes adjacent to DEIM-u pts.	48	56	63	111	117	126

Table 4.2. 2D Example: The computing times (in sec) and the number of Newton iterations (in parenthesis) needed to solve the full order model, the POD reduced order model, the POD-DEIM reduced order model, and the POD-DEIM-u (unassembled) reduced order model for different grid levels and linear and quadratic finite elements

Polynomial degree	$p=1$			$p=2$		
Mesh number	1	2	3	1	2	3
Full	0.55 (4)	0.41 (4)	1.49 (4)	0.85 (4)	1.36 (4)	5.75 (4)
POD	0.17 (4)	0.29 (4)	1.24 (4)	0.51 (4)	1.03 (4)	3.77 (4)
POD-DEIM	0.04 (4)	0.04 (4)	0.02 (4)	0.08 (4)	0.07 (4)	0.12 (4)
POD-DEIM-u	0.11 (8)	0.05 (5)	0.04 (5)	0.13 (5)	0.07 (4)	0.08 (4)

computing times do not include the time needed to compute the matrices $\mathbf{V}$, $\mathbf{U}$, or $\mathbf{U}^e$ via POD.

In this application, the solution of the POD-DEIM-u reduced order model required more Newton iterations in several cases, offsetting the gain in computational complexity of the POD-DEIM-u reduced order model nonlinearity. Another issue that makes computing time comparisons difficult using Matlab is that the computing time is often not determined by how many floating point operations are executed, but instead by how well the code is vectorized. We have made a great effort to vectorize the code for all models as much as possible, this is more effective for the full order and the POD reduced order models because by design the POD-DEIM and POD-DEIM-u reduced order models work with shorter vectors. Therefore the Matlab timings for the smaller problems likely do not accurately reflect what would be observed with, say, C code. However, from the Table 4.2 we can infer that POD reduced order models are only slightly more computationally efficient than the full order model. Applying DEIM for the assembled or unassembled form of the nonlin-

Table 4.3. 2D Example: Errors between the full order model solution and the POD reduced order model solution, the POD-DEIM reduced order model solution, and the POD-DEIM-u (unassembled) reduced order model solution

Polynomial degree	$p=1$			$p=2$		
Mesh number	1	2	3	1	2	3
POD	7.8e-5	1.5e-4	3.5e-4	1.3e-4	2.5e-4	6.9e-4
POD-DEIM	7.8e-5	9.4e-5	4.8e-4	2.5e-4	2.6e-4	7.9e-4
POD-DEIM-u	1.2e-4	1.5e-4	2.2e-4	1.4e-4	1.8e-4	6.3e-4

earity results in significant computational savings compared to both the full and the POD reduced order models when applied to larger problems. For larger problems the POD-DEIM-u reduced order model nonlinearities can be evaluated more efficiently than the POD-DEIM reduced order model nonlinearities. Different reduced order models may require different numbers of Newton iterations. In this example, the number of Newton iterations needed to solve the POD-DEIM-u reduced order model was at least as large as the number of Newton iterations needed to solve the POD-DEIM reduced order model. If the Newton iterations needed to solve the POD-DEIM-u reduced order model is larger, then the gains in efficiency of evaluating the nonlinearity is offset by the larger number of Newton iterations.

The errors between the full order model solution and the reduced order model solutions shown in Table 4.3 are of the order of the tolerance $\tau = 10^{-4}$ used to construct the bases with the POD.

4.4.2.2 3D Example

The domain is the cube $\Omega = (0,18) \times (0,9) \times (0,9)$ (in [mm]). The left face $\partial\Omega_D = \{0\} \times [0,9] \times [0,9]$ is the Dirichlet boundary, all other faces corresponds to Neumann boundaries $\partial\Omega_N$. On the part $\{0\} \times [3,6] \times \times [3,6]$ of the Dirichlet boundary we impose the Dirichlet conditions $y = 0.2$ and on the remainder of $\partial\Omega_D$ impose $y = 0$. This is the problem setup used in [11].

For the numerical solution, we use SUPG stabilized piecewise linear FEM on tetrahedra. To discretize the domain, Ω is divided into cubes of size $h \times h \times h$ and then each cube is divided into six tetrahedra. We use three meshes, Mesh 1 to Mesh 3, with $h = 1.125$, $h = 0.5625$, and $h = 0.375$, respectively. Mesh 2 is shown in the left plot in Fig. 4.6. The full order model solution of (4.3) parameter $(\ln(A), E) = (6.4, 0.11)$ is shown in the right plot in Fig. 4.6. (The reduced order model solutions are indistinguishable from the full order model solution.) For reasons explained below, we only apply piecewise quadratic finite elements on Meshes 1 and 2, but not on Mesh 3.

Figure 4.7 shows the tetrahedra in Mesh 2 that contain a node corresponding to a DEIM point. The plots in the left column correspond to the DEIM applied to the assembled form of the nonlinearity. In case of quadratic elements, the nodes are ei-

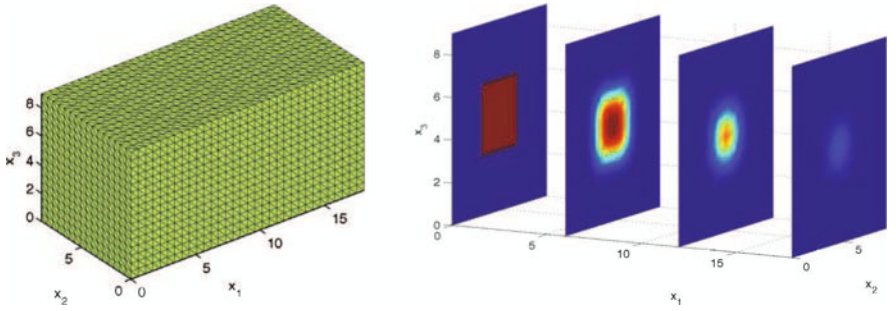

Fig. 4.6. 3D Example: Partitioning of the domain Ω into tetrahedra (left plot) and solution of the advection diffusion reaction equation (4.3) with parameter $(\ln(A), E) = (6.4, 0.11)$ (right plot)

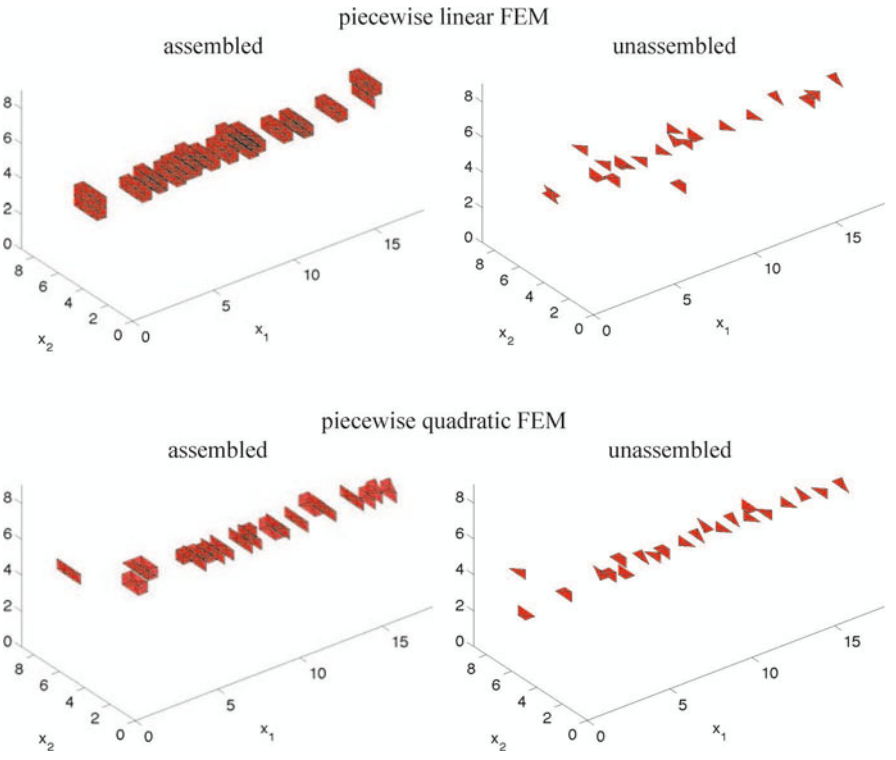

Fig. 4.7. 3D Example: The tetrahedra that contain DEIM points are shown. The different plots correspond to different polynomial degree used in the FEM and application of the DEIM to the assembled or unassembled form of the nonlinearity.

ther vertices or are edge midpoints. If we use Mesh 1, only one of the 21 DEIM points corresponds to a vertex. If we use Mesh 2, then none of the 21 DEIM points corresponds to a vertex. Since vertices are shared by more tetrahedra than edge midpoints, this means that DEIM points corresponding to edge midpoints lead to DEIM reduced order nonlinearities that can be evaluated more efficiently.

Table 4.4 summarizes the problem size for the different models for the three meshes and piecewise linear and quadratic finite elements. As before, in Tables 4.4 to 4.6, DEIM refers to the DEIM reduced order model (4.27) obtained using the assembled form of the nonlinearity, whereas DEIM-u refers to the DEIM reduced order model (4.30) obtained using the unassembled form of the nonlinearity.

The computing times to evaluate the full and the various reduced order models are shown in Table 4.5. Again, the computing times listed are for the entire Newton solve (and not for one Newton iteration). The number of Newton iterations required

Table 4.4. 3D Example: The size N of the full order finite element system, the number of POD basis vectors n, the number of DEIM points m the number of nodes adjacent to DEIM points, the number of DEIM points m^e when the unassembled nonlinearity is used, and the number of nodes adjacent to DEIM points for piecewise linear and quadratic finite elements on three grids. The mesh in Fig. 4.7 corresponds to grid number 2

Polynomial degree	$p=1$			$p=2$	
Mesh number	1	2	3	1	2
number of tetrahedra	6,144	49,152	165,888	6,144	49,152
number of nodes N	1,296	9,248	30,000	9,248	69,696
number of POD basis vectors n	19	18	19	18	19
number of DEIM points m	21	21	22	21	22
number of nodes adjacent to DEIM pts.	183	271	320	445	559
number of DEIM points m^e	21	21	22	21	22
number of nodes adjacent to DEIM pts.	67	80	88	193	220

Table 4.5. 3D Example: The computing times (in sec) and the number of Newton iterations (in parenthesis) needed to solve the full order model, the POD reduced order model, the POD-DEIM reduced order model, and the POD-DEIM-u (unassembled) reduced order model for different grid levels and linear and quadratic finite elements

Polynomial degree	$p=1$			$p=2$	
Mesh number	1	2	3	1	2
Full	1.78 (4)	10.60 (3)	43.30 (3)	7.80 (3)	185.00 (3)
POD	1.28 (4)	8.04 (3)	23.80 (3)	4.12 (3)	38.80 (3)
POD-DEIM	0.15 (4)	0.10 (3)	0.21 (4)	0.21 (3)	0.40 (3)
POD-DEIM-u	0.16 (9)	0.07 (4)	0.10 (4)	0.01 (4)	0.18 (4)

Table 4.6. 3D Example: Errors between the full order model solution and the POD reduced order model solution, the POD-DEIM reduced order model solution, and the POD-DEIM-u (unassembled) reduced order model solution

Polynomial degree	$p=1$			$p=2$	
Mesh number	1	2	3	1	2
POD	4.7e-5	1.7e-4	5.2e-4	1.1e-4	7.3e-4
POD-DEIM	3.4e-4	4.0e-4	4.5e-3	4.8e-4	2.4e-3
POD-DEIM-u	4.4e-4	1.7e-3	5.7e-3	1.4e-3	4.5e-3

are shown in (in parenthesis). The computing times do not include the time needed to compute the matrices $\mathbf{V}$, $\mathbf{U}$, or $\mathbf{U}^e$ via POD.

Table 4.4 shows that in the 3D case the DEIM applied to the unassembled form leads to nonlinear terms in the reduced order models which depend on significantly fewer components of the arguments than the nonlinear terms resulting from the DEIM applied to the assembled form. Table 4.5 shows that the POD-DEIM-u reduced order models are computationally more efficient than the POD-DEIM reduced order models, even if their solution required one more Newton iteration. The POD reduced order model leads to greater computational savings over the full order model in the 3D case compared to the 2D case (see Table 4.2). This is due to the computing time needed to solve the sparse linear systems in Newton's method. As before, significant reductions in computing times can only be achieved after DEIM is applied (either to the assembled or the unassembled form of the nonlinearity).

For 3D problems, the cost of solving the large sparse linear systems arising in Newton's method using the sparse LU decomposition is significant, especially for finer meshes and for piecewise quadratic elements. For the larger problems, it is likely beneficial to replace the direct solvers by iterative solvers. For this reason we have not included results for quadratic elements on the fine Mesh 3. The solution of the full order model using the sparse LU decomposition would have made the full order model solution artificially costly. Switching to iterative solvers for some discretizations would have raised the question what the 'best' iterative solver is. Therefore, we have limited our computational tests, to cases where the use of direct solvers still seems to be justifiable.

As in the 2D case, the errors between the full order model solution and the reduced order model solutions shown in Table 4.6 are of the order of the tolerance $\tau = 10^{-4}$ used to construct the bases with POD.

4.5 Evaluation of Parameterized Matrices and Vectors in Reduced Order Models Using DEIM

In this section we describe the use of the DEIM for the generation of efficient reduced order models that involve parameterized matrices. We first describe the approach

applied to a generic matrix $\mathbf{A}(\theta)$ and afterwards we apply it to the solution of Stokes equation in parameterized domains.

4.5.1 The Reduced Order Matrix

We consider a parametrically dependent matrix $\mathbf{A}(\theta)$ that has the representation

$$\mathbf{A}(\theta) = \sum_{i=1}^{M} \mathbf{g}_i(\theta)\mathbf{A}_i \tag{4.31}$$

with functions $\mathbf{g} = (\mathbf{g}_1, \ldots, \mathbf{g}_M)^T : \Theta \to \mathbb{R}^M$ and matrices $\mathbf{A}_i \in \mathbb{R}^{N\times N}$, $i = 1, \ldots, M$. As we have seen in Sect. 4.2.2 this is, e.g., the case when $\mathbf{A}(\theta)$ is the stiffness matrix of a parametrically varying linear PDE. In this subsection $\mathbf{A}(\theta)$ is a generic matrix. In the next subsection we will apply the reduction technique to the parametrically dependent Stokes system (4.16).

If we have computed the matrices $\mathbf{V}_\ell, \mathbf{V}_r \in \mathbb{R}^{N\times n}$, then the system matrix for the reduced order model is given by

$$\mathbf{V}_\ell^T \mathbf{A}(\theta)\mathbf{V}_r = \sum_{i=1}^{M} \mathbf{g}_i(\theta)\ \mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r. \tag{4.32}$$

If M is small, we can precompute the matrices $\mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r$ and for each θ we can use (4.32) to compute $\mathbf{V}_\ell^T \mathbf{A}(\theta)\mathbf{V}_r$ in $n^2 M$ operations. However, if M is large, which is the case, e.g., in the example in Sect. 4.2.2 an additional approximation is needed to allow for a fast computation of an approximation of $\mathbf{V}_\ell^T \mathbf{A}(\theta)\mathbf{V}_r$. We can apply the DEIM.

The DEIM computes a matrix $\mathbf{U} \in \mathbb{R}^{M\times m}$ of rank m and a function

$$\widetilde{\mathbf{g}} = (\widetilde{\mathbf{g}}_1, \ldots, \widetilde{\mathbf{g}}_m)^T = (\mathbf{P}^T\mathbf{U})^{-1}\mathbf{P}^T\mathbf{g} : \Theta \to \mathbb{R}^m \tag{4.33a}$$

such that

$$\mathbf{g}(\theta) \approx \widehat{\mathbf{g}}(\theta) = \mathbf{U}\widetilde{\mathbf{g}}(\theta). \tag{4.33b}$$

We assume $m \ll M$.

If we insert (4.33b) into (4.32), we obtain

$$\begin{aligned}
\mathbf{V}_\ell^T \mathbf{A}(\theta)\mathbf{V}_r &= \sum_{i=1}^{M} \mathbf{g}_i(\theta)\ \mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r \\
&\approx \sum_{i=1}^{M} \widehat{\mathbf{g}}_i(\theta)\ \mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r = \sum_{i=1}^{M}\sum_{j=1}^{m} \mathbf{U}_{ij}\widetilde{\mathbf{g}}_j(\theta)\ \mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r \\
&= \sum_{j=1}^{m} \left(\sum_{i=1}^{M} \mathbf{U}_{ij}\ \mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r \right) \widetilde{\mathbf{g}}_j(\theta).
\end{aligned} \tag{4.34}$$

The matrices $\sum_{i=1}^{M} \mathbf{U}_{ij}\, \mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r \in \mathbb{R}^{n\times n}$, $j = 1,\ldots,m$, can be precomputed. Afterwards, for each θ the reduced order matrix

$$\widehat{\mathbf{A}}(\theta) = \sum_{j=1}^{m} \left(\sum_{i=1}^{M} \mathbf{U}_{ij}\, \mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r \right) \widetilde{\mathbf{g}}_j(\theta) \tag{4.35}$$

can be computed at a cost of $n^2 m \ll n^2 M$ operations. Under the assumption that $\mathbf{b}(\theta)$ has a decomposition similar to (4.31), we can easily extend the ideas from reduction to $\mathbf{M}(\theta)$ in (4.35) to $\mathbf{b}(\theta)$. We have omitted those details to avoid repetition.

We note that the approximation presented previously can be generalized if $\mathbf{A}(\theta)$ is of the form

$$\mathbf{A}(\theta) = \sum_{i=1}^{M} \sum_{j=1}^{K} (\mathbf{g}_j)_i(\theta) \mathbf{A}_{ij} \tag{4.36}$$

by applying the previous techniques to each of the functions $\mathbf{g}_j = (\mathbf{g}_{1j},\ldots,\mathbf{g}_{Mj})^T$, $j = 1,\ldots,K$.

In many applications we also need to compute the derivative of the matrix $\mathbf{A}(\theta)$ with respect to θ. If the function $\mathbf{g}$ is differentiable, then the derivative of $\mathbf{A}(\theta)$ is given by

$$D_\theta \mathbf{A}(\theta) = \sum_{i=1}^{M} D_\theta \mathbf{g}_i(\theta) \mathbf{A}_i \tag{4.37}$$

and requires the evaluation of the derivative of all M functions $\mathbf{g}_1,\ldots,\mathbf{g}_M$. The same is true for the derivative of (4.32). The derivative of the DEIM reduced matrix,

$$D_\theta \widehat{\mathbf{A}}(\theta) = \sum_{j=1}^{m} \left(\sum_{i=1}^{M} \mathbf{U}_{ij}\, \mathbf{V}_\ell^T \mathbf{A}_i \mathbf{V}_r \right) D_\theta \widetilde{\mathbf{g}}_j(\theta) \tag{4.38}$$

only requires the evaluation of the $m \ll M$ functions $\widetilde{\mathbf{g}}_1,\ldots,\widetilde{\mathbf{g}}_m$. From (4.33) we have that

$$D_\theta \widetilde{\mathbf{g}} = \begin{pmatrix} D_\theta \widetilde{\mathbf{g}}_{p_1} \\ \vdots \\ D_\theta \widetilde{\mathbf{g}}_{p_m} \end{pmatrix} = (\mathbf{P}^T\mathbf{U})^{-1} \mathbf{P}^T D_\theta \mathbf{g} = (\mathbf{P}^T\mathbf{U})^{-1} \begin{pmatrix} D_\theta \mathbf{g}_{p_1} \\ \vdots \\ D_\theta \mathbf{g}_{p_m} \end{pmatrix},$$

since $\mathbf{P}^T$ just extracts the m rows from $D_\theta \mathbf{g}$ that corresponding to the DEIM indices $p_1,\ldots,p_m$. Thus evaluating the derivative $D_\theta \widehat{\mathbf{A}}(\theta)$ of the DEIM reduced matrix, requires the derivative of only $m \ll M$ functions $\mathbf{g}_{p_1},\ldots,\mathbf{g}_{p_m}$.

4.5.2 Numerical Example

We illustrate the DEIM approximation of parametrized matrices and vectors on the example of evaluating the objective function and its derivative in shape optimization of Stokes equation.

Suppose we want to minimize the functional

$$J(\theta) = \int_{\Omega(\theta)} l(u(\theta), p(\theta)) dx, \tag{4.39}$$

where the velocities $u(\theta)$ and pressure $p(\theta)$ are the solution of the Stokes equation (4.13). The function l will be specified later.

We assume that the domains $\Omega(\theta)$ are obtained by mapping a reference domain as shown in (4.12). Furthermore, we discretize the Stokes equation using P2-P1 finite elements as described in Sect. 4.2.2. The discretized Stokes system is given by

$$\mathbf{S}(\theta)\begin{pmatrix}\mathbf{u}\\ \mathbf{p}\end{pmatrix} = \mathbf{b}(\theta), \tag{4.40}$$

where $\mathbf{S}(\theta)$ has the form (4.18). In our examples, the forcing function f in (4.13) is zero. Therefore, the right hand side $\mathbf{b}$ is determined by the $\mathbf{S}(\theta)$ and the in homogenous Dirichlet data on the velocity in (4.13). The Stokes matrix $\mathbf{S}(\theta)$ in (4.18) has the same structure as the generic matrix $\mathbf{S}$ in (4.36). Since the forcing function f in (4.13) is zero, no additional parameterization of the right hand side $\mathbf{b}(\theta)$ is needed.

Applying the domain mapping, the P2-P1 finite element discretization, and the quadrature formula from Sect. 4.2.2 to the objective (4.39) gives the discrete objective functional

$$J_h(\theta) = \sum_{\ell=1}^{M} \omega_\ell \, l(u_h(\widetilde{x}_\ell), p_h(\widetilde{x}_\ell)) \, |\det(D\Phi(\widetilde{x}_\ell;\theta))|, \tag{4.41}$$

where u_h is the piecewise quadratic FEM approximation of the velocity and p_h is the piecewise linear FEM approximation of the pressure. The objective (4.41) depends on θ via the function $\mathbf{g}_8 : \Theta \to \mathbb{R}^M$ defined by

$$(\mathbf{g}_8(\theta))_\ell = \omega_\ell |\det(D\Phi(\widetilde{x}_\ell;\theta))|.$$

Since the discretized velocity and pressure u_h and p_h are determined by their coefficients $\mathbf{u}$ and $\mathbf{p}$, we can write the discrete objective functional (4.41) as

$$J_h(\theta) = \sum_{\ell=1}^{M} \mathbf{l}_\ell(\mathbf{u},\mathbf{p}) \, (\mathbf{g}_8(\theta))_\ell. \tag{4.42}$$

Note that its parameter dependence has the same structure as that of the generic matrix and therefore DEIM can be applied to reduce the computational cost. As in Sect. 4.2.2 we set

$$\mathbf{y} = \begin{pmatrix}\mathbf{u}\\ \mathbf{p}\end{pmatrix}.$$

To summarize, we want to minimize

$$J_h(\theta) = \mathbf{l}(\mathbf{y}(\theta))^T \mathbf{g}_8(\theta) = \sum_{\ell=1}^{M} \mathbf{l}_\ell(\mathbf{y}(\theta)) \, (\mathbf{g}_8(\theta))_\ell,$$

where $\mathbf{y}(\theta)$ solves (4.40). In many minimization problems there will be additional constraints on the parameter or on the velocities or pressures. Since we focus on the evaluation of reduced order models, we focus on the evaluation of $J_h(\theta)$ and on its gradient.

The evaluation of the objective function requires the following steps.

$\mathscr{J}_1$: Assemble $\mathbf{S}(\theta)$ and $\mathbf{b}(\theta)$ and solve the state equation $\mathbf{S}(\theta)\mathbf{y} = \mathbf{b}(\theta)$ for $\mathbf{y}(\theta)$.
$\mathscr{J}_2$: Compute $J_h(\theta) = \mathbf{l}(\mathbf{y}(\theta))^T \mathbf{g}_8(\theta)$.

We briefly summarize the computation of the gradient of $J_h(\theta)$ via the adjoint approach, see, e.g., [18, Sec. 1.6]. We define the Lagrangian functional

$$L(\mathbf{y}, \lambda, \theta) = \mathbf{l}(\mathbf{y})^T \mathbf{g}_8(\theta) + \lambda^T (\mathbf{S}(\theta)\mathbf{y} - \mathbf{b}(\theta)).$$

We assume that the objective has already been computed, i.e., that $\mathbf{S}(\theta)$ and $\mathbf{b}(\theta)$ have been assembled and that $\mathbf{y}(\theta)$ has been computed. Then the computation of the gradient requires the following steps.

$\mathscr{G}_1$: Solve the adjoint equation $\mathbf{S}(\theta)^T \lambda = -D_{\mathbf{y}} \mathbf{l}(\mathbf{y}(\theta))^T \mathbf{g}_8(\theta)$ for $\lambda(\theta)$.

$\mathscr{G}_2$: Compute $\nabla J_h(\theta) = \mathbf{l}(\mathbf{y}(\theta))^T D_\theta \mathbf{g}_8(\theta) + \lambda(\theta)^T (D_\theta \mathbf{S}(\theta)\mathbf{y}(\theta) - D_\theta \mathbf{b}(\theta))$.

To construct the reduced basis for the state equations (4.40), we sample the solution of the discrete Stokes (4.40) at r samples in the parameter domain Θ. We then apply the POD Algorithm 4.1 with tolerance τ individually to the snapshots for the x_1- and x_2-components of the velocity and the snapshots for the pressures. If N_v are the degrees of freedom for the x_1- and x_2-components of the velocity and N_p are the degrees of freedom for the pressure, the POD generates matrices $\mathbf{V}_{v_1} \in \mathbb{R}^{N_v \times n_{v_1}}$, $\mathbf{V}_{v_2} \in \mathbb{R}^{N_v \times n_{v_2}}$, and $\mathbf{V}_p \in \mathbb{R}^{N_p \times n_p}$. The reduced order Stokes matrix and right hand side are

$$\begin{aligned}
\mathbf{V}^T \mathbf{S}(\theta)\mathbf{V} &= \begin{pmatrix} \mathbf{V}_{v_1} & 0 & 0 \\ 0 & \mathbf{V}_{v_2} & 0 \\ 0 & 0 & \mathbf{V}_p \end{pmatrix}^T \begin{pmatrix} \mathbf{A}(\theta) & 0 & \mathbf{B}^{(1)}(\theta)^T \\ 0 & \mathbf{A}(\theta) & \mathbf{B}^{(2)}(\theta)^T \\ \mathbf{B}^{(1)}(\theta) & \mathbf{B}^{(2)}(\theta) & 0 \end{pmatrix} \begin{pmatrix} \mathbf{V}_{v_1} & 0 & 0 \\ 0 & \mathbf{V}_{v_2} & 0 \\ 0 & 0 & \mathbf{V}_p \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{V}_{v_1}^T \mathbf{A}(\theta)\mathbf{V}_{v_1} & 0 & \mathbf{V}_{v_1}^T \mathbf{B}^{(1)}(\theta)^T \mathbf{V}_p \\ 0 & \mathbf{V}_{v_2}^T \mathbf{A}(\theta)\mathbf{V}_{v_2} & \mathbf{V}_{v_2}^T \mathbf{B}^{(2)}(\theta)^T \mathbf{V}_p \\ \mathbf{V}_p^T \mathbf{B}^{(1)}(\theta)\mathbf{V}_{v_1} & \mathbf{V}_p^T \mathbf{B}^{(2)}(\theta)\mathbf{V}_{v_2} & 0 \end{pmatrix}
\end{aligned} \tag{4.43}$$

and

$$\mathbf{V}^T \mathbf{b}(\theta) = \begin{pmatrix} \mathbf{V}_{v_1} & 0 & 0 \\ 0 & \mathbf{V}_{v_2} & 0 \\ 0 & 0 & \mathbf{V}_p \end{pmatrix}^T \mathbf{b}(\theta).$$

The preliminary version of the reduced order Stokes system is

$$\mathbf{V}^T \mathbf{S}(\theta)\mathbf{V}\widehat{\mathbf{y}} = \mathbf{V}^T \mathbf{b}(\theta). \tag{4.44}$$

Since the $\mathbf{b}_3$ component of the right hand side (4.16) of the discrete Stokes equation is nonzero, the velocity snapshots are not divergence free (in the discrete sense). Therefore, as already noted in [30], there is no guarantee that the reduced Stokes matrix (4.43) satisfies an inf-sup condition. In [30] a procedure is proposed that enriches the velocity subspaces to guarantee the inf-sup condition. Instead we monitor the inf-sup constant corresponding to (4.43) by computing the singular values of the small matrix $\widehat{\mathbf{B}}(\theta) = (\mathbf{V}_p^T \mathbf{B}^{(1)}(\theta)\mathbf{V}_{v_1}\ ,\ \mathbf{V}_p^T \mathbf{B}^{(2)}(\theta)\mathbf{V}_{v_2})^T$ and found that no enrichment of the velocity space was needed in our example.

The matrix $\mathbf{S}(\theta)$ has the structure (4.18). Since in our examples the forcing function f in (4.13) is zero, no additional parameterization of the right hand side $\mathbf{b}(\theta)$ is needed. We apply the DEIM as described in the previous Sect. 4.5.1 to obtain a DEIM reduced order matrix $\widehat{\mathbf{S}}(\theta)$ and right hand side vector $\widehat{\mathbf{b}}(\theta)$. Specifically, the reduced bases, the matrices $\mathbf{U}$ for the nonlinear terms $\mathbf{g}_1, \ldots, \mathbf{g}_8$ are constructed by sampling these nonlinearities at the same parameters used to construct the reduced basis $\mathbf{V}_{v_1}, \mathbf{V}_{v_2}$ and $\mathbf{V}_p$ and then applying POD with tolerance τ to get matrices $\mathbf{U}$ for the DEIM. We apply DEIM to each of the eight functions $\mathbf{g}_1, \ldots, \mathbf{g}_8$ separately, i.e., for each of the eight functions we generate a matrix $\mathbf{U}$. Applying the DEIM approximation of Sect. 4.5.1 we obtain the DEIM reduced order system

$$\widehat{\mathbf{S}}(\theta)\widehat{\mathbf{y}} = \widehat{\mathbf{b}}. \tag{4.45}$$

Furthermore, applying DEIM to the function $\mathbf{g}_8$ in the objective, i.e., approximating

$$\mathbf{g}_8(\theta) \approx \widehat{\mathbf{g}}_8(\theta) = \mathbf{U}_8 \widetilde{\mathbf{g}}_8(\theta).$$

where

$$\widetilde{\mathbf{g}}_8 = ((\widetilde{\mathbf{g}}_8)_1, \ldots, (\widetilde{\mathbf{g}}_8)_m)^T = (\mathbf{P}_8^T \mathbf{U}_8)^{-1} \mathbf{P}_8^T \mathbf{g}_8 : \Theta \to \mathbb{R}^{m_8}$$

leads to the reduced order objective

$$\widehat{J}_h(\theta) = \mathbf{l}(\mathbf{V}\widehat{\mathbf{y}}(\theta))^T \mathbf{U}_8 \widetilde{\mathbf{g}}_8(\theta), \tag{4.46}$$

where $\widehat{\mathbf{y}}(\theta)$ is the solution of (4.45). In our applications, $\mathbf{l}$ is affine linear or quadratic in $\mathbf{y}$, so that fast computation of $\widehat{\mathbf{y}} \mapsto \mathbf{l}(\mathbf{V}\widehat{\mathbf{y}})^T \mathbf{U}_8 \widetilde{\mathbf{g}}_8(\theta)$ is possible.

All computations were done using Matlab on a MacBook Pro with 8GB of memory and a 2.53 GHz Intel Core 2 Duo processor. The nonlinear full order or reduced order models are solved using Newton's method. The linear systems are solved using the Matlab backslash command. The θ derivatives are computed using INTLAB Version 5.5 [31].

4.5.2.1 Evaluation of Drag Generated by Parameterized Airfoil

The drag on the boundary portion $\Gamma_{\text{drag}}(\theta) \subset \partial\Omega(\theta)$ is defined by

$$C_D = -\frac{2}{U_\infty^2 L} \int_{\Gamma_{\text{drag}}(\theta)} ((\nu \nabla u(x) - p(x)I)n(x)) \cdot \widehat{u}_\infty \, ds, \tag{4.47}$$

where $u_\infty = U_\infty \widehat{u}_\infty$ is the velocity of the incoming flow, $\widehat{u}_\infty$ is the unit vector directed as the incoming flow, U_∞ is constant, and L is the characteristic length of the body. See, e.g., [16,20]. As usual, we use the Stokes equations (4.13) to find an equivalent formula for the drag that avoids integration over the boundary. We use a function $v_\infty \in (H^1(\Omega(\theta)))^2$ with $v_\infty = \widehat{u}_\infty$ on $\Gamma_{\text{drag}}(\theta)$ and $v_\infty = 0$ on $\partial\Omega(\theta) \setminus \Gamma_{\text{drag}}(\theta)$ as a

test function in (4.13) to obtain

$$\begin{aligned} 0 &= -\int_{\partial\Omega(\theta)} ((\nu\nabla u - pI)n)\cdot v_\infty + \int_{\Omega(\theta)} (\nu\nabla u - pI) : \nabla v_\infty - \int_{\Omega(\theta)} f\cdot v_\infty, \\ &= -\int_{\Gamma_{\text{drag}}(\theta)} ((\nu\nabla u - pI)n)\cdot \widehat{u}_\infty + \int_{\Omega(\theta)} (\nu\nabla u - pI) : \nabla v_\infty - \int_{\Omega(\theta)} f\cdot v_\infty. \end{aligned}$$

Hence,

$$C_D = -\frac{2}{U_\infty^2 L}\left(\int_{\Omega(\theta)} (\nu\nabla u(x) - p(x)I) : \nabla v_\infty(x)\,dx - \int_{\Omega(\theta)} f(x)\cdot v_\infty(x)\,dx\right). \tag{4.48}$$

We use C_D as our objective functional for this example, i.e., in this example J in (4.39) is given by (4.48).

The domain $\Omega(\theta)$ (for $\theta = 0.5$) is sketched in Fig. 4.8 and has the boundary $\partial\Omega(\theta) = \Gamma_{\text{in}} \cup \Gamma_D \cup \Gamma_{\text{drag}}(\theta) \cup \Gamma_{\text{out}}$, where $\Gamma_{\text{in}} = \{-6\}\times(-3,5)$, $\Gamma_D = ((-6,6)\times\{-3\})\cup((-6,6)\times\{5\})$, $\Gamma_{\text{out}} = \{6\}\times(-3,5)$ and $\Gamma_{\text{drag}}(\theta)$ is the boundary of airfoil. We specify an inflow velocity $h = 1$, on Γ_{in} and a constant viscosity $\nu = 0.1$. The forcing function f in (4.13) is taken to be zero. We assume that the airfoil is of unit length, and the boundary has the following parameterization,

$$\Gamma_{drag} = \{(x_1,x_2) \mid 0 \le x \le 1, x_2 = 1 + \eta(\theta)\}$$

where for $\theta \in [0,2]$

$$\eta(\theta) = \pm\frac{\theta}{0.2}\left(0.2969\sqrt{x_1} - 0.1260x_1 - 0.3520x_1^2 + 0.2832x_1^3 - 0.1021x_1^4\right).$$

The diffeomorphism Φ that is used to map the reference domain $\widetilde{\Omega}$, is given by

$$\Phi((\widetilde{x}_1,\widetilde{x}_2);\theta) = \begin{pmatrix} \widetilde{x}_1 \\ (1+\eta(\theta))\widetilde{x}_2 \end{pmatrix} = (x_1,x_2)^T.$$

for $\widetilde{x}_1 \in [0,1]$ and $\Phi((\widetilde{x}_1,\widetilde{x}_2);\theta) = (\widetilde{x}_1,\widetilde{x}_2)$ else. The reference domain is $\widetilde{\Omega} = \Phi((-6,6)\times(-3,5);0.5)(=\Omega(0.5))$ and is shown in Fig. 4.8.

The problem is discretized using P2-P1 Taylor Hood elements as described in Sect. 4.2.2. We compute 25 snapshots each for both the solution to the state equations and the nonlinear terms. The reduced basis are generated using the POD with a tolerance $\tau = 10^{-6}$.

We evaluate C_D (see (4.47)) and its derivative with respect to θ at an arbitrary point $\theta = \sqrt{2} \in \Theta$, which is not in the snapshot set. Table 4.7 summarizes the size of the full and the reduced order systems for three finite element grids using the full order model, the POD reduced order model and the POD-DEIM reduced order model. The mesh in Fig. 4.8 is the coarse Mesh 1. The DEIM points (quadrature points) chosen are contained in the triangles marked in red.

The computing times to evaluate the objective function (steps $\mathscr{J}_1 + \mathscr{J}_2$) and its gradient (steps $\mathscr{G}_1 + \mathscr{G}_2$) for the full order model and the reduced order models are shown in Table 4.8. For the reduced order models the times do not include off-line cost. Most of the computational cost in computing objective functional occurs in Step

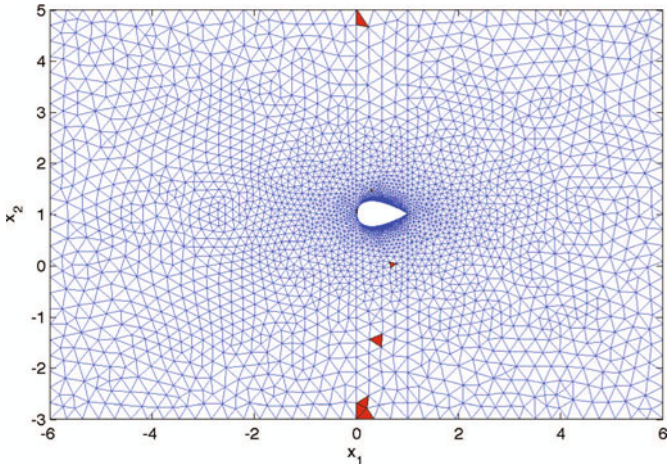

Fig. 4.8. The reference domain $\widetilde{\Omega}$ for the NACA airfoil 4-digit family. The DEIM points (quadrature points) are contained in the triangles marked in solid red

$\mathscr{J}_1$. Computing the gradient requires solving the adjoint equations (Step $\mathscr{G}_1$) and the sensitivities of system matrices and the objective functional (Step $\mathscr{G}_2$) with respect to the shape parameter θ. Since this example only involves a scalar parameter, for the full order model Step $\mathscr{G}_1$ is the most expensive step in the evaluation of the gradient of objective functional. The cost of sensitivities increases with the number of parameters, see Sect. 4.5.2.2.

The errors between the full order model (objective functional C_D) and the reduced order model solutions shown in Table 4.9 are of the order of the tolerance $\tau = 10^{-6}$ used to construct the bases with the POD. The error in gradient computation is slightly higher, due to the fact that adjoint solutions have not been taken into account to generate the reduced bases. This accuracy can be easily improved by enriching the snapshot set.

Table 4.7. The size $N = N_v + N_v + N_p$ of the full order finite element system, the number of POD basis vectors $n = n_{v_1} + n_{v_2} + n_p$, and the number of DEIM points $m = \sum_{\ell=1}^{8} m_\ell$

Mesh number	1	2	3
number of triangles	6,094	8,838	24,990
number of nodes N	27,039	39,343	111,887
number of POD basis vectors n	55	58	63
number of DEIM points $m(=\sum_{\ell=1}^{8} m_\ell)$	26	26	26

4.5.2.2 Channel with Parameterized Top and Bottom Wall

In our second example we consider a channel in which the bottom and top boundaries are parameterized using Bézier curves. The reference domain is $\widetilde{\Omega} = (-1,1)^2$. The bottom and top wall of the channel are parameterized by Bézier curves with p_T control points for the top boundary and p_B control points for the bottom boundary. Thus the physical domain $\Omega(\theta)$ is parameterized by $\theta \in \Theta \subset \mathbb{R}^p$, $p = p_T + p_B$.

The boundary $\partial\Omega(\theta)$ is decomposed into the inflow and outflow boundaries $\Gamma_{\text{in}}(\theta) = \{-1\} \times (-1,1)$, and $\Gamma_{\text{out}}(\theta) = \{1\} \times (-1,1)$, both of which are independent of the parameterization and the top and bottom boundaries $\Gamma_t(\theta)$ and $\Gamma_b(\theta)$. The viscosity is $\nu = 1.0$. On the inflow boundary Γ_{in} we specify an parabolic inflow velocity $h = 8(1+x_2)(1-x_2)$. The velocity $h = 0$ on $\Gamma_t(\theta)$ and $\Gamma_b(\theta)$. The forcing function f in (4.13) is taken to be zero.

We use $p_T = p_B = 2$ Bézier control points to specify the top and the bottom boundary of the variable domain $\Omega(\theta)$. The parameters are in $\Theta = (0.5,3.0) \times (0.5,3.0) \times (-3.0,-0.5) \times (-3.0,-0.5)$

For this example the objective functional is

$$J(\theta) = \int_{\Omega(\theta)} |u - u^d|^2 dx$$

where u are the velocities computed as the solution of the Stokes equations (4.13) on $\Omega(\theta)$. The functions u^d are the desired velocities computed by solving the stokes equation on $\Omega(\theta^d)$ with fixed parameter $\theta^d = (1.0, 0.5, -0.5 - 1.0)^T$.

The problem is discretized using P2-P1 Taylor Hood elements as described in Sect. 4.2.2. To construct the reduced basis, we compute 5^4 snapshots in the parameter domain Θ i.e., we take 5 sample points in each direction. Then we apply Algorithm 4.1 with tolerance $\tau = 10^{-4}$ to construct the reduced basis, as before.

We evaluate J and its derivative with respect to θ at an arbitrary point $\theta = (\sqrt{2}, \sqrt{2}, -\sqrt{2}, -\sqrt{2})^T \in \Theta$, which is not in the snapshot set. Table 4.10 summarizes the size of the full and the reduced order systems for three finite element grids using the full order model, the POD reduced order model and the POD-DEIM reduced order model.

Table 4.8. The computing times (in sec) to evaluate the objective functional (Steps $\mathscr{J}_1 + \mathscr{J}_2$), and the gradient of objective functional (Steps $\mathscr{G}_1 + \mathscr{G}_2$) corresponding to the full order model, the POD reduced order model, and the POD-DEIM reduced order model for different meshes

Mesh number	1		2		3	
	$\mathscr{J}_1 + \mathscr{J}_2$	$\mathscr{G}_1 + \mathscr{G}_2$	$\mathscr{J}_1 + \mathscr{J}_2$	$\mathscr{G}_1 + \mathscr{G}_2$	$\mathscr{J}_1 + \mathscr{J}_2$	$\mathscr{G}_1 + \mathscr{G}_2$
Full	2.04	2.00	3.16	2.94	12.50	12.80
POD	0.83	0.89	1.23	1.07	3.49	2.95
POD-DEIM	0.02	0.02	0.02	0.01	0.05	0.03

Table 4.9. The errors between the full order and the POD reduced order model (objective functional and its gradient) and errors between the full order and the POD-DEIM reduced order model (objective functional and its gradient) for different meshes

Mesh number	1		2		3	
	objective	*gradient*	*objective*	*gradient*	*objective*	*gradient*
POD	4.67e-6	5.08e-5	5.83e-6	2.44e-4	1.12e-5	4.50e-4
POD-DEIM	5.31e-6	1.11e-4	6.05e-6	2.66e-4	1.21e-5	3.67e-4

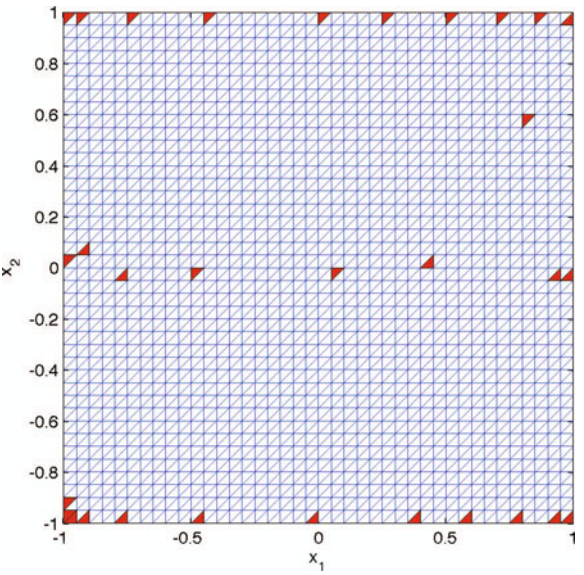

Fig. 4.9. The reference domain $\widetilde{\Omega}$ for the channel example. The top Γ_T and bottom Γ_B boundaries are parameterized by $p_T = 2$ and $p_B = 2$ Bézier control points respectively. The DEIM points (quadrature points) lies in the interior of the triangles marked in solid red

The mesh in Fig. 4.9 is the coarse Mesh 1. The DEIM points (quadrature points) chosen are contained in the triangles marked in red.

The computing times to evaluate the full and the various reduced order objective (Steps $\mathscr{J}_1 + \mathscr{J}_2$) and its gradient (Steps $\mathscr{G}_1 + \mathscr{G}_2$) are shown in Table 4.8. For the reduced order models the times do not include off-line cost. As in the previous example, most of the computing cost for the computation of the objective function occurs in step $\mathscr{J}_1$, the assembly and solution of the state equation. Computing the gradient requires solving the adjoint equations (Step $\mathscr{G}_1$) and the sensitivities of system matrices and the objective functional (Step $\mathscr{G}_2$) with respect to the shape parameter θ. We observe that in this example and for the full order model, Step $\mathscr{G}_2$ is the most expensive step in the evaluation of the gradient of the objective functional. This is due to the fact that we have four parameters.

Table 4.10. The size $N = N_v + N_v + N_p$ of the full order finite element system, the number of POD basis vectors $n = n_{v_1} + n_{v_2} + n_p$, and the number of DEIM points $m = \sum_{\ell=1}^{8} m_\ell$. The mesh in Fig. 4.8 corresponds to grid number 1

Mesh number	1	2	3
number of triangles	800	3,200	7,200
number of nodes N	3,561	14,321	32,281
number of POD basis vectors n	261	264	265
number of DEIM points $m(=\sum_{\ell=1}^{8} m_\ell)$	53	53	53

Table 4.11. The computing times (in sec) to evaluate the objective functional (Steps $\mathscr{J}_1 + \mathscr{J}_2$), and the gradient of objective functional (Steps $\mathscr{G}_1 + \mathscr{G}_2$) corresponding to the full order model, the POD reduced order model, and the POD-DEIM reduced order model for different meshes

Mesh number	1		2		3	
	$\mathscr{J}_1 + \mathscr{J}_2$	$\mathscr{G}_1 + \mathscr{G}_2$	$\mathscr{J}_1 + \mathscr{J}_2$	$\mathscr{G}_1 + \mathscr{G}_2$	$\mathscr{J}_1 + \mathscr{J}_2$	$\mathscr{G}_1 + \mathscr{G}_2$
Full	0.33	0.73	1.50	3.86	5.00	13.30
POD	0.36	1.06	1.26	4.95	4.41	15.90
POD-DEIM	0.05	0.13	0.04	0.14	0.09	0.14

Table 4.12. The errors between the full order and the POD reduced order model (objective functional and its gradient) and errors between the full order and the POD-DEIM reduced order model (objective functional and its gradient) for different meshes

Mesh number	1		2		3	
	objective	gradient	objective	gradient	objective	gradient
POD	2.01e-3	2.53e-3	1.91e-3	1.61e-3	1.83e-3	1.62e-3
POD-DEIM	2.03e-3	2.57e-3	1.93e-3	1.71e-3	1.82e-3	1.63e-3

The errors between the full order model (objective functional C_D and its gradient) and the reduced order model solutions shown in Table 4.12 are of the order of the tolerance $\tau = 10^{-4}$ used to construct the bases with the POD.

4.6 Conclusions

We have demonstrated the application of the DEIM to compute reduced order models for finite element discretizations of seminar elliptic PDEs and for parameterized linear systems that arise, e.g., in shape optimization, and we have studied the computational efficiency of the resulting reduced order models.

The efficiency with which DEIM reduced order models of discretized semilinar elliptic PDEs can be evaluated is determined by how many components of the argument each component of the nonlinearity depends on. For finite element discretizations this dependence is determined by the mesh, the polynomial degree used in the finite element approximation, but also by whether the nonlinearity is defined in its assembled or unassembled form. For nodal based finite element methods, each component of the unassembled form of the nonlinearity depends only on the components associated with the degrees of freedom corresponding to one element. This is different for the assembled form of the nonlinearity. Here a component of the nonlinearity can depend on the degrees of freedom in several adjacent elements. More precisely, if the component of the nonlinearity corresponds to a node on the boundary of an element, then this component of the nonlinearity depends on all degrees of freedom in the elements that share this node. Because of the dependence of the components of the nonlinearity on the components of its argument, the unassembled form is attractive for DEIM. Since DEIM applied to the different forms of the nonlinearity generates different reduced order models, which require different numbers of Newton iterations to solve, the dependency of the nonlinearity on its argument alone cannot be used to decide which form of the DEIM is favorable. Our numerical examples have shown that either version of the DEIM is preferable over the naive application of projection based model reduction. For large systems, the application of the DEIM to the unassembled form of the nonlinearity led to additional gains in the on-line cost of the reduced order models. The off-line cost of DEIM applied to the unassembled form of the nonlinearity is always higher (and can be significantly higher) since the unassembled form results in a nonlinear vector valued function that has significantly more components than the nonlinear vector valued function arising in the assembled form.

A second focus of this paper was to demonstrate the application of the DEIM to compute reduced order models for an important class of parameterized linear systems. The DEIM not only leads to reduced order models that can be evaluated efficiently, but in addition the derivatives of the reduced order models with respect to the parameter can be computed efficiently. Both efficiency gains are crucial, e.g., for shape optimization. We have demonstrated this numerically using the Stokes equations on parameterized domains.

References

1. Antoulas, A.C.: Approximation of large-scale dynamical systems, Advances in Design and Control, vol. 6. Society for Industrial and Applied Mathematics, Philadelphia, PA (2005)
2. Barrault, M., Maday, Y., Nguyen, N.D., Patera, A.T.: An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. C. R. Math. Acad. Sci. Paris **339**(9), 667–672 (2004)
3. Becker, R., Braack, M., Vexler, B.: Numerical parameter estimation for chemical models in multidimensional reactive flows. Combust. Theory Modelling **8**, 6 (2004)

4. Binev, P., Cohen, A., Dahmen, W., DeVore, R., Petrova, G., Wojtaszczyk, P.: Convergence rates for greedy algorithms in reduced basis methods. SIAM J. Math. Anal. **43**(3), 1457–1472 (2011)
5. Brooks, A.N., Hughes, T.J.R.: Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations. Comp. Meth. Appl. Mech. Engng. **32**, 199–259 (1982)
6. Buffa, A., Maday, Y., Patera, A.T., Prud'homme, C., Turinici, G.: A priori convergence of the greedy algorithm for the parametrized reduced basis method. ESAIM Math. Model. Numer. Anal. **46**(3), 595–603 (2012)
7. Chaturantabut, S., Sorensen, D.C.: Nonlinear model reduction via discrete empirical interpolation. SIAM Journal on Scientific Computing **32**(5), 2737–2764 (2010)
8. Chaturantabut, S., Sorensen, D.C.: Application of POD and DEIM on dimension reduction of non-linear miscible viscous fingering in porous media. Math. Comput. Model. Dyn. Syst. **17**(4), 337–353 (2011)
9. Dedden, R.J.: Model order reduction using the discrete empirical interpolation method. Master's thesis, Technical University Delft, Netherlands, 2012. Available from http://repository.tudelft.nl (accessed Dec. 31, 2012)
10. Elman, H.C., Silvester, D. J., Wathen, A. J.: Finite Elements and Fast Iterative Solvers with Applications in Incompressible Fluid Dynamics. Oxford University Press, Oxford (2005)
11. Galbally, D., Fidkowski, K., Willcox, K., Ghattas, O.: Nonlinear model reduction for uncertainty quantification in large-scale inverse problems. Internat. J. Numer. Methods Engrg. **81**(12), 1581–1603 (2010)
12. Galdi, G.P., Simader, C.G., Sohr, H.: On the Stokes problem in Lipschitz domains. Ann. Mat. Pura Appl. (4) **167**, 147–163 (1994)
13. Girault, V., Raviart, P.-A.: Finite element methods for Navier-Stokes equations. Theory and algorithms, volume 5 of Springer Series in Computational Mathematics. Springer-Verlag, Berlin Heidelberg (1986)
14. Grepl, M.A., Maday, Y., Nguyen, N.C., Patera, A.T.: Efficient reduced-basis treatment of nonaffine and nonlinear partial differential equations. M2AN Math. Model. Numer. Anal. **41**(3), 575–605 (2007)
15. Grepl, M.A., Patera, A.T.: A posteriori error bounds for reduced-basis approximations of parametrized parabolic partial differential equations. M2AN Math. Model. Numer. Anal. **39**(1), 157–181 (2005)
16. He, J.-W., Glowinski, R., Metcalfe, R., Nordlander, A., Periaux, J.: Active control and drag optimization for flow past a circular cylinder. I. oscillatory cylinder rotation. Journal of Computational Physics **163**, 83–117 (2000)
17. Hinze, M., Kunkel, M.: Discrete empirical interpolation in pod model order reduction of drift-diffusion equations in electrical networks. In: Michielsen, B., Poirier, J.-R. (eds.) Scientific Computing in Electrical Engineering SCEE 2010, Mathematics in Industry, pp. 423–431. Springer-Verlag, Berlin Heidelberg (2012)
18. Hinze, M., Pinnau, R., Ulbrich, M., Ulbrich, S.: Optimization with Partial Differential Equations. of Mathematical Modelling, Theory and Applications, vol. 23. Springer-Verlag, Berlin Heidelberg New York (2009)
19. Hinze, M., Volkwein, S.: Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: Error estimates and suboptimal control. In: Benner, P., Mehrmann, V., Sorensen, D. C. (eds.) Dimension Reduction of Large-Scale Systems, Lecture Notes in Computational Science and Engineering, vol. 45, pp. 261–306. Springer-Verlag, Berlin Heidelberg (2005)

20. John, V.: Reference values for drag and lift of a two-dimensional time-dependent flow around a cylinder. International Journal for Numerical Methods in Fluids **44**(7), 777–788 (2004)
21. Lall, S., Marsden, J.E., Glavaški, S.: A subspace approach to balanced truncation for model reduction of nonlinear control systems. Internat. J. Robust Nonlinear Control **12**(6), 519–535 (2002)
22. Lass, O., Volkwein, S.: POD Galerkin schemes for nonlinear elliptic-parabolic systems. Konstanzer Schriften in Mathematik No. 301, FB Mathematik & Statistik, Universität Konstanz, D-78457 Konstanz, Germany, 2012. To appear in SIAM J. Scientific Computing
23. Lions, P.-L.: On the existence of positive solutions of semilinear elliptic equations. SIAM Rev. **24**(4), 441–467 (1982)
24. Patera, A.T., Rozza, G.: Reduced Basis Approximation and A Posteriori Error Estimation for Parametrized Partial Differential Equations. MIT Pappalardo Graduate Monographs in Mechanical Engineering. Cambridge, MA (2007). Available from http://augustine.mit.edu/methodology/methodology_book.htm
25. Quarteroni, A., Valli, A.: Numerical Approximation of Partial Differential Equations. Springer-Verlag, Berlin Heidelberg New York (1994)
26. Roos, H.G., Stynes, M., Tobiska, L.: Robust Numerical Methods for Singularly Perturbed Differential Equations. Computational Mathematics, Vol. 24, 2nd ed. Springer-Verlag, Berlin Heidelberg (2008)
27. Roubíček, T.: Nonlinear partial differential equations with applications, volume 153 of International Series of Numerical Mathematics. Birkhäuser, Basel (2005)
28. Rowley, C.W.: Model reduction for fluids, using balanced proper orthogonal decomposition. Int. J. on Bifurcation and Chaos **15**(3), 997–1013 (2005)
29. Rozza, G., Huynh, D.B.P., Patera, A.T.: Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations: application to transport and continuum mechanics. Arch. Comput. Methods Eng. **15**(3), 229–275 (2008)
30. Rozza, G., Veroy, K.: On the stability of the reduced basis method for Stokes equations in parametrized domains. Comput. Methods Appl. Mech. Engrg. **196**(7), 1244–1260 (2007)
31. Rump, S.M.: INTLAB – INTerval LABoratory. In: Csendes, T. (ed.) Developments in Reliable Computing, pp. 77–104. Kluwer Academic Publishers, Dordrecht (1999). http://www.ti3.tu-harburg.de/rump/
32. Stynes, M.: Steady-state convection-diffusion problems. In: Iserles, A. (ed.) Acta Numerica 2005, pp. 445–508. Cambridge University Press, Cambridge, London, New York (2005)
33. Tiso, P., Dedden, R.J., Rixen, D.J.: DEIM for nonlinear structural dynamics model order reduction, 2012. Talk presented at the 10th World Congress on Computational Mechanics, Sao Paulo, Brazil (8–13 July 2012)