Juergen Geiser

# Multicomponent and Multiscale Systems

## Theory, Methods, and Applications in Engineering

Springer

Multicomponent and Multiscale Systems

Juergen Geiser

# Multicomponent and Multiscale Systems

## Theory, Methods, and Applications in Engineering

Springer

Juergen Geiser
Department of Electrical Engineering
Ruhr University of Bochum
Bochum
Germany

ISBN 978-3-319-15116-8 ISBN 978-3-319-15117-5 (eBook)
DOI 10.1007/978-3-319-15117-5

Library of Congress Control Number: 2015947793

Springer Cham Heidelberg New York Dordrecht London


Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

*First, I am grateful to my colleagues at the Ernst-Moritz Arndt University of Greifswald, Germany, and Ruhr-University of Bochum, Germany, for their support and ideas on modelling and computational sciences. Next, I have to thank my supporters and mentors in all modelling problems and splitting ideas. They helped me to be open and to be sensitive to rigorous analysis of the numerical methods, modelling of engineering problems and their applications.*

*My special thanks go to my wife Andrea and my daughter Lilli who have always supported and encouraged me.*

# Preface

I am glad to introduce *Multicomponent and Multiscale Systems: Theory, Methods and Applications in Engineering*.

When I started this book project, I proposed to write a book about my recent advances in mathematical modelling problems to multicomponent and multiscale systems. I considered the upcoming areas in material modelling, which include transport and reaction flow simulations and also electronic applications with electromagnetic fields.

I organized this book in combining theoretical and also application to practical problems. While multicomponent and multiscale systems are very new problems, the early stage of such a field needs such a book to explain in a theoretical and also a practical manner the tools and methods to solve such problems.

I have tried to fill the gap between numerical methods and the applications to real problems. I present rigorously the fundamental aspects of the numerical methods with their underlying analysis and applying such schemes to real-life.

This monograph is in the field of technical and physical simulation problems in engineering and sciences. Based on the theoretical framework in methods and structures of applied mathematics, it concludes with numerical approximations of multi-component and multi-scale problem. A main motivation of the book came from students and researchers in different lectures and research projects.

In this monograph, we describe the theoretical and practical aspects of solving complicated and multi-component and multi-scale systems, which are applied in engineering models and problems.

In the book, we are motivated to describe numerical receipts, based on different multi-scale and multi-component methods, that allow to apply truly working multi-scale and multi-component approaches. Nowadays, one of the main problems in multi-scale and multi-component systems is the gap between several models based on different time- and spatial-scales. Often the drawback of applying standard numerical methods, e.g. explicit time-discretization schemes, instead of working multi-scale approaches, e.g. multi-scale expansion methods, is, that we have a dramatic limiting factor, e.g. very small time- or spatial steps (due to resolving the

finest scale). Such limiting factors did not allow to solve engineering complexity and industrial advancement is impossible to obtain. Here, we fill the gap between numerical methods and their applications to engineering complexities of real-life problems.

Such engineering complexities are delicate and need extraordinary treatment with special solver and tools to overcome the difficulties and restrictions of time- and spatial steps.

Therefore, we discuss the ideas of solving such multi-component and multi-scale systems with the help of non-iterative and iterative methods. Often such methods can be related to splitting multi-scale methods to be taken into account to decompose such problems to simpler ones. Such decomposition allows to treat the complex systems in simpler ones and skip the restriction of the finest scale to the solver methods, while we can apply individual scale to the decomposed system.

We discuss analytical and numerical methods in time and space for evolution equations and also nonlinear evolution equations with respect to their linearization and relaxation schemes.

All problems are related to engineering problems and their applications. I have started from reactive flow and transport models, which are related to bioremediation, combustion and various CFD applications, to delicate electronic models, which are related to plasma transport and flow processes in technical apparatus.

The main motivation is to embed novel multiscale approaches to complex engineering problems such that it is possible to apply a model-reduction. Thus, it is possible that parts of the model can be reduced or for those based on multiscale or multicomponent approaches, the data-transfer between fine and coarse grid is done, in a way that each scale is considered.

The outline of the monograph is given as:

1. Introduction (outline of the book)
2. General principles for multi-component and multiscale systems
    a. Multi-component Analysis (separating of components)
    b. Multiscale analysis (separating of scales)
    c. Mathematical methods
3. Theoretical part: functional splitting:
    a. Decomposition of a global multi-component problem
    b. Decomposition of a global multiscale problem
4. Algorithmic part
    a. Iterative methods
    b. Additive methods
    c. Parallelization

5. Models and applications
   a. Multicomponent applications
      i. Application of multicomponent fluids
      ii. Application of multicomponent kinetics
      iii. Analytical methods for a multicomponent transport model
   b. Multiscale applications
      i. Additive splitting method for Maxwell-equations
      ii. Nonuniform grids for particle in cell methods
6. Engineering applications (real-life models)
   a. Multicomponent applications
      i. Application of a multicomponent model in a plasma-mixture problem
      ii. Application of a multicomponent model in a biological problem (glycolysis)
   b. Multiscale applications
      i. Application of a multiscale model in a stochastic problem
      ii. Application of a multiscale model in a code-coupling problem
      iii. Application of a multiscale model in a dynamical problem
      iv. Application of a multiscale model in a particle transport problem
      v. Application of a multiscale model in plasma applications
      vi. Application of a multiscale model in complex fluids
7. Conclusion (fields of application and future ideas)

Based on the outline of the book, we hope that we could increase the attention of both industry and scientists; theoretical and practical aspects are illustrated and considered in an equal way.

Dallgow-Doeberitz Juergen Geiser
June 2015

# Acknowledgments

I would like to thank Th. Zacher for programming MULTI-OPERA software and his help in the numerical experiments. I would also like to acknowledge my colleagues and students who helped me to write such a book and gave me hints with numerical and experimental results.

# Contents

# Acronyms

| | |
|---|---|
| BCH | Baker-Campbell-Hausdorff formula |
| BDF | Backward differentiation formula |
| CFD | Computational Fluid Dynamics |
| CFL | Courant-Friedrichs-Lewy condition |
| CIC | Cloud-in-Cell function (see [1] and [2]) |
| CVD | Chemical vapor deposition |
| CSP | Computational Singular Perturbation Method (see [3]) |
| DD | Domain decomposition methods |
| FDTD | Finite Difference Finite Time method (see [4]) |
| EFM | Equation Free Method (see [5]) |
| HIPIMS | High Power Impulse Magnetron Sputtering |
| HMM | Heterogeneous Multiscale Method (see [6]) |
| HPM | Homotopy Perturbation Method (see [7]) |
| ICE-PIC | Invariant Constrained-equilibrium Edge Pre-Image Curve (see [8]) |
| ILDM | Intrinsic Low Dimensional Manifold approach (see [9]) |
| IOS | Iterative operator splitting methods |
| MAX-phase | Special material with metallic and ceramic behavior; see [10] |
| MD | Molecular dynamics |
| MIG | Method of Invariant Grid (see [11]) |
| MIM | Method of Invariant Manifold |
| MISM | Multiscale Iterative Splitting Method (see [12]) |
| MQ | Multiquadric bases functions (see [13]) |

| | |
|---|---|
| MULTI-OPERA | Software package based on MATLAB, which solves multiscale problems with splitting methods |
| ODE | Ordinary differential equation |
| OFELI | Object finite element library |
| PDE | Partial differential equation |
| PECVD | Plasma-enhanced chemical vapor deposition |
| PIC | Particle in Cell (see [1]) |
| PID | Proportional integral derivative controller |
| PM | Particle Method (see [1]) |
| PVD | Physical vapor deposition |
| RBF | Radial basis function |
| REDIMs | Reaction–Diffusion Manifolds approach (see [14]) |
| RRM | Relaxation Redistribution Method (see [15]) |
| $R^3T$ | Radioactive-reaction-retardation-transport software toolbox, done with the software package UG (unstructured grids) |
| SiC | Silicon carbide |
| $Ti_3SiC_2$ | Special material used for thin-layer deposition; see [10] |
| SIM | Slow Invariant Manifold |
| SDE | Stochastic Differential Equation |
| SODE | Stochastic Ordinary Differential Equation |
| SPDE | Stochastic Partial differential equation |
| UG | Unstructured grid (software package; see [16]) |

## Symbols

| | |
|---|---|
| $\lambda$ | Eigenvalue |
| $A$ | In the following $A$ is a matrix in $\mathbb{R}^m \times \mathbb{R}^m$, $m \in \mathbb{N}^+$ is the rank |
| $\lambda_i$ | $i$-th eigenvalue of $A$ |
| $\rho(A)$ | Spectral radius of $A$ |
| $e_i$ | $i$-th eigenvector of matrix $A$ |
| $\sigma(A)$ | Spectrum of $A$ |
| $Re(\lambda_i)$ | $i$-th real eigenvalue of $\lambda$ |
| $u_t = \frac{\partial u}{\partial t}$ | First-order partial time derivative of $c$ |
| $u_{tt} = \frac{\partial^2 u}{\partial t^2}$ | Second-order partial time derivative of $c$ |
| $u_{ttt} = \frac{\partial^3 u}{\partial t^3}$ | Third-order partial time derivative of $u$ |
| $u_{tttt} = \frac{\partial^4 u}{\partial t^4}$ | Fourth-order partial time derivative of $u$ |
| $u' = \frac{du}{dt}$ | First-order time derivative of $u$ |
| $u'' = \frac{d^2 u}{dt^2}$ | Second-order time derivative of $u$ |
| $\tau = \tau_n = t^{n+1} - t^n$ | Time step |

| | |
|---|---|
| $u^n$ | Approximated solution of $u$ at time $t^n$ |
| $\partial_t^+ u = \frac{u^{n+1}-u^n}{\tau_n}$ | Forward finite difference of $u$ in time |
| $\partial_t^- u = \frac{u^n-u^{n-1}}{\tau_n}$ | Backward finite difference of $u$ in time |
| $\partial_t^0 u = \frac{u^{n+1}-u^{n-1}}{2\tau_n}$ | Central finite difference of $u$ in time |
| $\partial_t^2 u = \partial_t^+ \partial_t^- u$ | Second-order finite difference of $u$ in time |
| $\nabla u$ | Gradient of $u$ |
| $\Delta u(x,t)$ | Laplace operator of $u$ |
| $\nabla \cdot \mathbf{u}$ | Divergence of $\mathbf{u}$ (where $\mathbf{u}$ is a vector function) |
| $n_m$ | Outer normal vector to $\Omega_m$ |
| $\partial_x^+ u$ | Forward finite difference of $u$ in space dimension $x$ |
| $\partial_x^- u$ | Backward finite difference of $u$ in space dimension $x$ |
| $\partial_x^0 u$ | Central finite difference of $u$ in space dimension $x$ |
| $\partial_x^2 u$ | Second-order finite difference of $u$ in space dimension $x$ |
| $\partial_y^+ u$ | Forward finite difference of $u$ in space dimension $y$ |
| $\partial_y^- u$ | Backward finite difference of $u$ in space dimension $y$ |
| $\partial_y^0 u$ | Central finite difference of $u$ in space dimension $y$ |
| $\partial_y^2 u$ | Second-order finite difference of $u$ in space dimension $y$ |
| $e_i(t) := u(t) - u_i(t)$ | Local error function with approximated solution $u_i(t)$ |
| $err_{\text{local}}$ | Local error |
| $err_{\text{global}}$ | Global error |
| $[A,B] = AB - BA$ | Commutator of operators $A$ and $B$ |

## References


1. R. Hockney, J. Eastwood, *Computer Simulation Using Particles* (CRC Press, Boca Raton, 1985)
2. M.E. Innocenti, G. Lapenta, S. Markidis, A. Beck, A. Vapirev, A multi level multi domain method for particle in cell plasma simulations. J. Comput. Phys. **238**, 115–140 (2013)
3. S.H. Lam, D.A. Goussis, The CSP method for simplifying kinetics. Int. J. Chem. Kinet. **26**, 461–486 (1994)
4. A. Taflove, *Computational Electrodynamics: The Finite Difference Time Domain Method* (Artech House Inc., Boston, 1995)
5. I.G. Kevrekidis, G. Samaey, Equation-free multiscale computation: algorithms and applications. Annu. Rev. Phys. Chem. **60**, 321–344 (2009)
6. E. Weiman *Principle of Multiscale Modelling* (Cambridge University Press, Cambridge, 2010)
7. S.J. Liao, On the homotopy analysis method for nonlinear problems. Appl. Math. Comput. **147**, 499–513 (2004)
8. Z. Ren, St. B. Pope, A. Vladimirsky, J.M. Guckenheimer, Application of the ICE-PIC method for the dimension reduction of chemical kinetics coupled with transport. Proc. Combust. Inst. **31**, 473–481 (2007)

9. U. Maas, S.B. Pope, Simplifying chemical kinetics: intrinsic low-dimensional manifolds in composition space. Combust. Flame **88**, 239–264 (1992)
10. M.W. Barsoum, T. El-Raghy, Synthesis and characterization of a remarkable ceramic: $Ti_3SiC_2$. J. Am. Ceram. Soc. **79**(1), 1953–1956 (1996)
11. E. Chiavazzo, I.V. Karlin, K. Boulouchos, Method of invariant grid for model reduction of hydrogen combustion. Proc. Combust. Inst. **32**(1), 519–526 (2009)
12. J. Geiser, Recent advances in splitting methods for multiphysics and multiscale: theory and applications. J. Algorithms Comput. Technol., Multi-Sci., Brentwood, Essex, UK, accepted August 2014 (to be published second issue 2015)
13. R.L. Hardy, Multiquadric equations of topography and other irregular surfaces. J. Geophys. Res. **76**(8), 1905–1915 (1971)
14. V. Bykov, U. Maas, Problem adapted reduced models based on reaction diffusion manifolds (*REDIMs*). Institut für Technische Thermodynamik, Karlsruhe University, Kaiserstraße 12, D-76128 Karlsruhe, Germany, Proceedings of the Combustion Institute 01/2009 (2009)
15. E. Chiavazzo, I. Karlin, Adaptive simplification of complex multiscale systems. Phys. Rev. E **83**, 036706 (2011)
16. P. Bastian, K. Birken, K. Eckstein, K. Johannsen, S. Lang, N. Neuss, H. Rentz-Reichert, UG—a flexible software toolbox for solving partial differential equations. Comput. Vis. Sci. **1**(1), 27–40 (1997)

# Introduction

While engineering applications are becoming increasingly complicate, the underlying modelling problems are becoming more related with multi-modelling aspects. Such complexities arise due to multiscale and multicomponent approaches in the modelling-equations, which need rigorous numerical analysis for the underlying schemes. The main problems are the disparate time- and spatial scales, which have to be included into the models and their underlying numerical approaches.

In the next chapters, we like to solve such delicate problems with numerical schemes, which are improved multi-scale and multi-component methods.

We discuss the following items:

- General principles for multi-component and multiscale systems,
- Multi-component analysis (separating of components),
- Multiscale analysis (separating of scales),
- Mathematical and numerical methods.

While we start with classical multicomponent and multiscale methods, e.g. homogenization and asymptotic matching, we discuss their limits and application background. Such limits allow us to take into account the design of structure and algorithmical methods, which overcome the restriction of disparate scales and modify such methods to apply engineering problems with delicate complexities.

Here the main topic is related to splitting methods, which are nowadays applied to multi-scale and multi-component problems, while they are flexible in coupling different spatial and time scales.

We discuss additive and iterative methods, which can be embedded to standard discretization and solver schemes, such that the multiscales are respected in their modelling structures. Practical and theoretical tools are extended with scientific simulations of their underlying models, which allows revealing the deeper structures, for example multi-component and multi-scale structures, which are coupled in different time and spatial scales.

Based on the upcoming areas of multi-scale approaches for material modelling, see the framework of Horizon 2020,[1] it is important to link different models, e.g.,

- Multi-scaling: Different time- or spatial scales of the phenomena are modelled in different entities (e.g., micro- or macro model) and their results are transferred from one model to another.
- Multi-Modelling: Different physics and chemistry are coupled at the same scale, which means the models are applied to the same time- and spatial scale.

Such new modelling areas are nowadays important and we take into account the modification of our proposed multi-scale and multi-component methods to customize for practical engineering problems. One of the key motivation is to bridge the gap between the engineering application and the development of multi-scale methods for theoretical test applications, such that it is possible to adapt the theoretical tested schemes to real problems.

Here in our book, we concentrate on multi-scaling, which means, we can apply our models and methods to such problems, that different models (microscopic model or macroscopic model) are coupled via a method and therefore, we transfer the results from one to the other model.

We consider the following multi-scaling:

- Multi-scaling: Different time- or spatial scales of the phenomena are modelled in different entities (e.g., micro- or macro model) and their results are transferred from one model to another.
- Multi-Modelling: Different physics and chemistry is coupled at the same scale, which means the models are applied to the same time- and spatial scale.

Based on the first interpretation, we have in the book the following examples:

- Langevin-like equations (micro- and macro-time scale), see Sect. 5.1,
- Levitron Problem (micro- and macro-spatial scale), see Sect. 5.3,
- Glycolysis Problem (micro- and macro-time scale), see Sect. 5.6.

In Fig. 1, we present the first interpretation (same physics model to different spatial and time scales).

In Fig. 2, we present the second interpretation (linking models with different physics). Based on the second interpretation, we have in the book the following examples:

- Code-coupling (fluid dynamical and heat-transfer model), see Sect. 5.2,
- Adaptive Particle in Cell (molecular dynamical model and continuum model), see Sect. 5.4,
- Multicomponent Plasma (kinetic model and continuum model), see Sect. 5.5.

[1]European Commission, Research & Innovation-Key Enabling Technologies, Modelling Material, http://ec.europa.eu/research/industrial_technologies/modelling-materials_en.html.

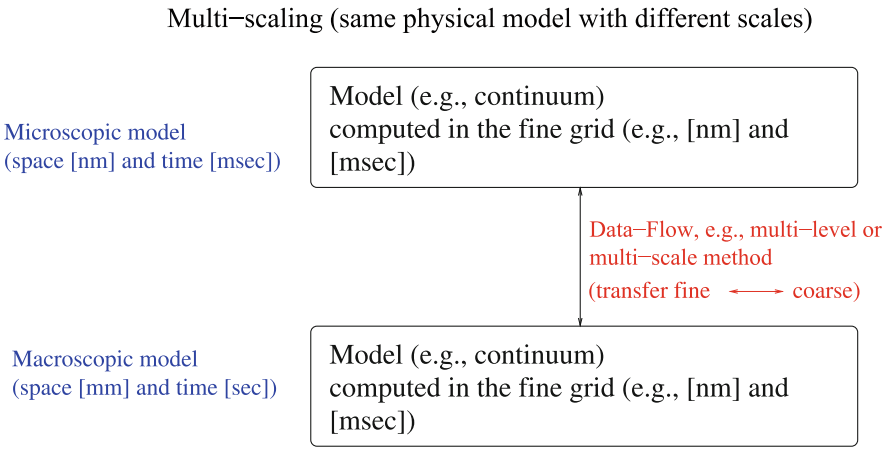



**Fig. 1** First interpretation of multi-scaling (often in classical material engineering applied)

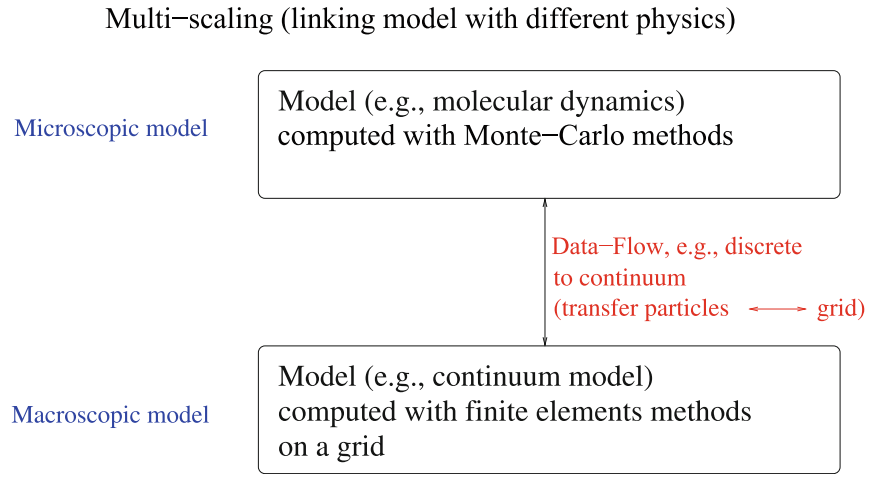



**Fig. 2** Second interpretation of multi-scaling (in modern material engineering applied)

In the book, we try to close the gap between several available models, e.g. in material modelling, due to disparate time and spatial scales, and the possibility to apply multi-scale and multi-component methods to couple such scales.

The use of such truly working multi-scale approaches is important in the case of engineering complexity; in the book, we present such approaches. Nowadays, if such methods are not considered or well-studied in the applications, it is a dramatic limiting factor for today's industrial advancement, see [1].

## Reference

1. L. Rosso, A.F. de Baas, Review of Materials Modelling: What makes a material function? Let me compute the ways … European Commission, General for Research and Innovation Directorate, Industrial Technologies, Unit G3 Materials (2014). http://ec.europa.eu/research/industrial_technologies/modelling-materials_en.html

# Chapter 1
# General Principles

**Abstract** In the general principle, we give an overview of the recently used methods and schemes to solve multicomponent and multiscale systems. While multicomponent systems are evolution equations based on each single species, which are coupled with the other species, e.g. with reaction-, diffusion-processes, multiscale systems are evolution equations based on different scales for each species, e.g. macroscopic- or microscopic scale. We give the general criteria for practically performing the different splitting and multiscale methods, such that a modification to practical applications of the splitting schemes to a real-life problem can be done.

## 1.1 Multicomponent Systems

In the following, we deal with multicomponent systems. Multicomponent systems concentrate on disparate components in the underlying multiscale models, while they can be coupled by linear or nonlinear functions or differential systems, e.g. reactions or transport phenomena. Often, also different scales are important to resolve to understand the interactions of the different components. Here, we have to apply multicomponent schemes that are also related to disparate spatial and timescales to resolve such complexities, see algorithmic ideas of multicomponent problems in [1].

In the following sections, we concentrate on modelling or algorithmical aspects of multicomponent systems for the following applications:

- Multicomponent flows, see [2, 3].
- Multicomponent transport, see [1, 4].

We simulate the flow and transport systems based on their interactions with the different components. Hence, we allow to study weakly or strongly coupled components in the underlying modelling equation systems.

### *1.1.1 Multicomponent Flows*

The class of multicomponent flows can be defined as a mixture of different chemical species on a molecular level, which are flown with the same velocity and temperature,



J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5_1

see [5]. The chemical species are interacting by chemical reactions and such a result is a multicomponent reactive flow.

The modelling of such behaviours is important in engineering, e.g. reactor design (Chemical vapour deposition reactors, see [6]) in chemical engineering. Such processes are very complex, while different physical processes occur, e.g. injection, heating, mixturing, homogeneous and heterogeneous chemistry and further, see [7].

In the following, we present some typical problems in multicomponent flow problems:

- Ionized Species, e.g. plasma problems, see [4].
- Combustion of oil, coal or natural gas, see [8].
- Chemical reaction processes in chemical engineering, see [6, 9, 10].
- Atmospheric pollution, see [11].

*Remark 1.1* In the different applications, the term multiphase flow is often used. Here, we define multiphase flows where the phases are immiscible and not chemically related, see [2, 12]. So each phase has a separately defined volume fraction and velocity field. Therefore, also the conservation equations for the flow of each species and their interchange between the phases are different from the multicomponent flow. Here, one is taken into account to define a common pressure field, while each phase is related to the gradient of this field and its volume fraction, see [13]. The applications are two-phase flow problems, Buckley Leverett problems and multiphase heat transfer, see [2].

### *1.1.2 Multicomponent Transport*

We define the multicomponent transport in the direction of a computational aspect. If we deal in our description with multicomponent flow model, e.g. multicomponent plasma, multicomponent fluid, the interest is related in this item to the transport properties, e.g. of the chemical mixture.

The algorithmical aspect is important to deal with multicomponent systems; here often the idea of splitting into simpler and faster equation parts is important, e.g.:

- Multicomponent splitting of multicomponent flow problem, e.g. ocean modelling [14].
- Multicomponent transport algorithms, e.g. fluid modelling [4].
- Multicomponent transport modelling, e.g. plasma modelling [15].

All ideas are related to decompose the multicomponent model into simpler single-component models and solve them separately.

### 1.1.3 Application of Operator Splitting Methods to Multicomponent Flow and Transport Problems

The general criteria for a practical performing of the operator splitting methods to multicomponent systems are motivated by decomposing into simpler systems, which can be solved independently or with less computational amount, see [16, 17].

One of the main advantages of decoupling operators in multicomponent systems is the computational efficiency, while we decompose the operators in their different temporal and spatial scales. Such a decomposition allows to apply the most accurate discretization and solver methods.

A classical receipt of such a decomposition for multicomponent systems is given in Fig. 1.1.

*Example 1.1* As an example, we deal with a multicomponent transport problem, a multi-diffusion-reaction equation. Such a multi-diffusion operator is decomposed into simpler diffusion operators. Each operator part can be solved with its accurate method (e.g. implicit time discretization for the fastest diffusion part to allow large time steps and higher order explicit Runge–Kutta methods for slowest diffusion part to obtain accurate results), see [18].

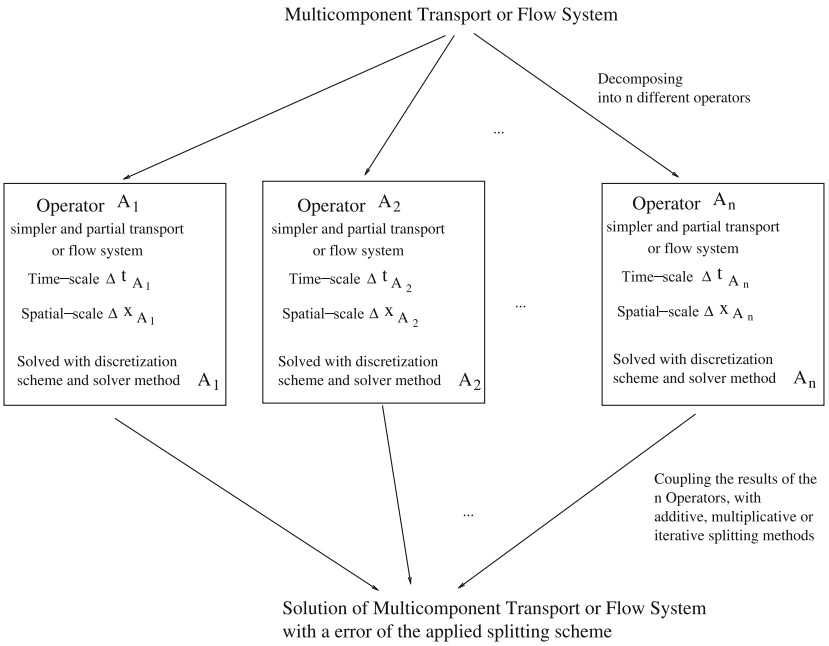



**Fig. 1.1** Multicomponent systems and an operator splitting approach

## 1.2 Multiscale Systems

Multiscale systems deal with modelling equations of different time- and spatial behaviours. While different scale lengths, e.g. microscopic or macroscopic scales, in time and space are important, also the different models of the problem, e.g. microscopic or macroscopic model, are important to understand the complexity of the underlying system. Different hierarchies of models are important to see in each different time- or spatial scale the influence to the model problem or the influence to a lower or upper hierarchical model, see [19, 20].

We have to understand the related multiscale models, e.g. different scales or different models, and also the application of different multiscale methods, e.g. top–down or bottom–up methods, see [21].

In the following, we deal with multiscale systems and their modelling and algorithmical aspects.

We discuss the following topics:

- Multiscale Modelling, see [2, 3].
- Multiscale Methods, see [1, 4].

We take into account the different modelling types, type A and type B, and the practical implementation to adequate multiscale methods to close the gap between the influence of disparate time- and spatial schemes. We can accelerate the computational time while we solve upscaled microscopic equation into fast performing macroscopic equations or concentrate on embedding microscopic performance to important time- or spatial windows of the macroscopic equations, e.g. updating macroscopic parameters with microscopic computations.

### *1.2.1 Multiscale Modelling*

In the past years, in many engineering applications, multiscale systems are important tools for solving engineering models which have multiple scales, e.g. spatial and/or temporal scales of different, see [22].

The modelling aspects concentrate on resolving the properties or system behaviour on each important level (e.g. microscale, mesoscale, macroscale) by using additional information from the other levels or scales (e.g. lower or higher levels).

While each level has its own specific behaviour, e.g. conservation or constraints, such a behaviour is important for a detailed description of the full system with all the levels.

Multiscale modelling is therefore important to understand the detailed information of an engineering model, e.g. the material and system behaviours. Such a detailed analysis allows to forecast the material or system behaviours.

### 1.2.2 Multiscale Methods

There are several different methods to solve multiscale problems.

Often, it is sufficient to deal with analytical methods, e.g. method of multiple-timescales (MMTS) see [23]. The next larger group is numerical methods, where we distinguish between different types of algorithms:

- Top–down.
- Bottom–up.

Further, we distinguish between classical and modern numerical schemes, while the classical schemes, e.g. multiscale methods, multiresolution methods, have linear scaling and modern schemes, e.g. heterogeneous multiscale method (HMM), equation free method (EFM), iterative multiscale methods (IMSM) have sublinear scaling of the computational time, see [22].

### 1.2.3 Application of Different Multiscale Methods to Multiscale Problems

In the following, we discuss three recipes of practical application of the following multiscale methods:

- HMM (Heterogeneous Multiscale Method), see [24].
- EFM (Equation free method), see [25].
- MISM (Multiscale Iterative Splitting Method), see [26].

For multiscale problems, a main motivation is also to reduce the computational amount, see [22].

To have a general criteria to apply to a real model, we can classify a multiscale problem into two types and apply each type to the appropriate method, see [20, 26].

- Multiscale Problem of Type A (top–down)
  The micro-model is only used for the regions where the microscopic laws are important, e.g. local defects or singularities, boundary-layers or interfaces. For all other regions it is sufficient to apply the macro-model (i.e. the macroscopic equations). Here, the examples are fluid–solid models or plasma-boundary models.
- Multiscale Problem of Type B (bottom–up)
  The microscale model is necessary for all the regions (e.g. globally) to derive the parameters for the macroscopic laws. The macroscopic model is extrapolated by the microscopic model, i.e. we reconstruct the macroscopic equations.

The methods for the different types are given in the following:

**HMM: Top–down-method**

Here, we have the HMM (Heterogeneous Multiscale Method), which embeds the microscale model into the macroscopic model. That is, the macroscopic model is extended by the information of the microscopic model.

We have the following steps of the HMM algorithm, see Algorithm 1.1.

We deal with the following multiscale equation:

$$\frac{dy}{dt} = \frac{1}{\varepsilon}(y - \phi(x)), \tag{1.1}$$

$$\frac{dx}{dt} = f(x, y), \tag{1.2}$$

where $y$ is the fast and $x$ is the slow variable. Further, we assume $\varepsilon \ll 1$ and $f, \phi$ are nonlinear functions and $t \in [0, T]$, where $T$ is the end-time point.

**Algorithm 1.1** • We solve the microscopic equation:

$$y^{n,m+1} = y^{n,m} - \frac{\delta t}{\varepsilon}(y^{n,m} - \phi(x^n)), \tag{1.3}$$

with $m = 0, 1, \ldots, M-1$, e.g. $\delta t \leq \Delta t/M$ as microscopic time-steps.

- We reconstruct or equilibrate the microscale operator:

$$F^n = \frac{1}{M}\sum_{m=1}^{M} f(x^n, y^{n,m}). \tag{1.4}$$

- We solve the macroscopic equation, with respect to the improved operator $F^n$:

$$x^{n+1} = x^n - \Delta t F^n. \tag{1.5}$$

with the macroscopic time-step $\Delta t$.

*Remark 1.2* We solve only a few microscopic time-steps around a macroscopic time-point, such that we do not resolve the full macroscopic time-step, therefore we can reduce the microscopic computations.

**MISM: Top–down-method**

Here, we have a next top–down method, which deals with an underlying macroscopic equation with $A$ as the macroscopic operator and the coarse timescale $\tau$. Further, we assume a microscopic equation (e.g. material law) with the microscopic operator $B$ and the fine timescale $\delta\tau$.

The different scales and the method is illustrated in Fig. 1.2.

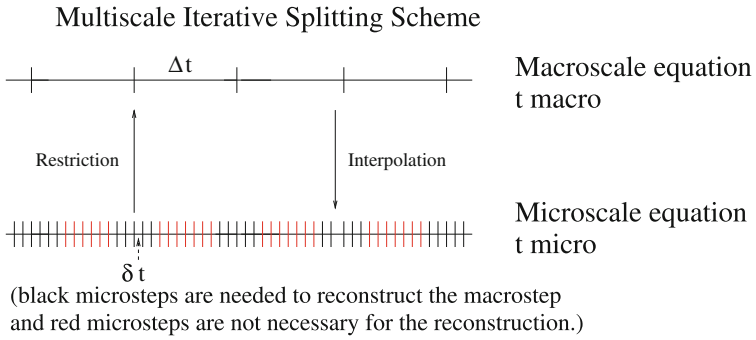



**Fig. 1.2** Illustration of the MISM method

The idea is to embed the results of the microscopic equations around the macroscopic time-points, i.e. smaller time frame of microscopic steps as the full macroscopic time-step, such that we can save computational time. The finer scale is embedded into the coarser scale and it is sufficient to update a smaller time interval around the coarser time-steps to concentrate on the macroscopic equations, which can be solved much more efficiently than the microscopic equation.

The Algorithm 1.2 is given in the following.

**Algorithm 1.2** We have the following parameters:

- The coarse time-step is $\tau$.
- The fine time-step is $\delta\tau \leq \tau/M$, where $M$ is the number of small time-steps around the coarse time-step.
- The macroscopic time interval is given as $[t^n, t^{n+1}]$.

The algorithm is given in the next steps:

- Initialization: $c_0(t^n) = c^n$, $I$ is the number of iteration steps and we have $N$ time intervals.
- We solve the macroscopic equation with one time-step $\tau$:

$$\frac{\partial c_i(t)}{\partial t} = A(c_i(t)) \; + \; R(B(c_{i-1}(t)). \tag{1.6}$$

- Then we apply the interpolation, i.e. operator $A$ is resolved in the finer scale, so we couple into the microscopic equation

$$I(A(c_i)(t)) = A(c(t^n)) + \frac{\partial A(c)}{\partial c}\frac{c_i(t) - c(t^n)}{c_i(t^{n+1}) - c(t^n)}, \; t \in [t^n, t^{n+1}]. \tag{1.7}$$

$$I(A(c_i)(t)) = A(c(t^n)) + \frac{A(c_i(t^{n+1})) - A(c(t^n))}{c_i(t^{n+1}) - c(t^n)}(c_i(t) - c(t^n)), \; t \in [t^n, t^{n+1}]. \tag{1.8}$$

- We apply $M$ small time-steps ($\delta\tau$) in the microscopic equation:

$$\frac{\partial c_{i+1}(t)}{\partial t} = I(A(c_i(t))) + B(c_{i+1}(t)). \tag{1.9}$$

- Then we apply the restriction, i.e. the operator $B$ is coupled to the macroscopic equation.

$$R(B(c_j)(t^{n+1})) = \frac{1}{M}\sum_{k=1}^{M} B(c_{j,k}(t^{n+1})). \tag{1.10}$$

- We apply the next macroscopic step.

*Remark 1.3* Such a method can be applied by the model where we can distinguish the different operators (e.g. coarse timescale of operator $A$ and fine time scales of operator $B$). While the method is very near to a standard operator splitting scheme, the modifications are only to embed the different interpolation and restriction operators, see [26]. The benefit is in reducing the computational amount, which is from the microscopic equation (e.g. molecular dynamical computations for large time frames) to shift the model via embedding of the microscopic results to the macroscopic equation, e.g. updated macroscopic parameters (diffusion or stress tensors), see [24].

**EFM: Bottom–up-method**

Here, we have the EFM (equation free method), which extrapolates the macroscale model. Vice versa, we have given the microscopic model and resolve the unknown macroscopic model.

We have the following steps of the HMM algorithm, see Algorithm 1.3.

**Algorithm 1.3** • We initialize the microscopic equation (Lifting):

$$u(x,t) = \mu(U(x,t)). \tag{1.11}$$

- We solve the microscopic equation (Evolving):

$$u(x, t + \Delta t) = s^M(u(x,t), \delta t), \tag{1.12}$$

where $\Delta t = M\ \delta t$ is the macroscopic time-step.

- We extrapolate the macroscopic equation (Restriction):

$$U(x,t) = \mathscr{M}(u(x,t)). \tag{1.13}$$

- We reconstruct via the operators the macroscopic equation:

$$U(x, t + \Delta t) = S(U(x,t), \Delta t) = \mathscr{M}(s^M(\mu(U(x,t)), \delta t)). \tag{1.14}$$

*Remark 1.4* The EFM has the advantage to model ab initio problems, e.g. in material modelling, it is important to start from a very fine scale and reconstruct the higher macroscopic models, see [27]. Here, we have the benefit that we can reconstruct the macroscopic behaviour, while for example a macroscopic equation is not a priori known. Based on extrapolation ideas we extend the microscopic scales and construct larger time- and spatial scales to transfer the model into the macroscopic scales, see [25, 28]. Such a reconstruction is important to understand ab initio material processes, see [29].

## 1.3 Multicomponent Analysis

In the following section, we deal with the multicomponent analysis, which is in our case related to the ideas of separating the components.

Therefore, we deal with multicomponent methods to separate and decouple the components, where we deal with the following ideas:

- Additive and multiplicative splitting methods.
- Iterative splitting methods.

The different splitting schemes are used as kernel methods and we can extend and modify such basic methods to the underlying engineering complexities, see [26].

### *1.3.1 Additive and Multiplicative Splitting Methods*

For the decomposition of partial differential equations, there exists different splitting techniques, e.g. splitting in different dimensions, splitting in different operators, etc.

We briefly introduce the most common ideas to decompose a multicomponent equation into different components or operators.

We deal with a linear ordinary differential equation with constant coefficients given as

$$\frac{du(t)}{dt} = A_{\text{full}}\, u(t),\ t \in (0, T), \tag{1.15}$$

$$\frac{du(t)}{dt} = (A_1 + \cdots + A_M)\, u(t),\ t \in (0, T), \tag{1.16}$$

where $A_{\text{full}}, A_1, \ldots, A_M : \mathbb{R}^m \to \mathbb{R}^m$ are matrices, $u = (u_1, \ldots, u_m)^T$ is the solution vector, and $m$ is a given positive number, where $M$ is the number of operators. The initial conditions are given as $u(t = 0) = u_0$, while $u_0$ is a given constant vector.

#### 1.3.1.1 Additive Splitting Scheme

The idea of additive splitting schemes are to decompose into a additive series of operators.

We solve $M$ subproblems sequentially on subintervals $[t^n, t^{n+1}]$, where $n = 0, 1, \ldots, N-1$, $t^0 = 0$ and $t^N = T$.

So if we deal with a first-order scheme, we can decompose Eq. (3.71) into a series of explicit schemes:

$$u(t^{n+1}) = \left( I + \tau \sum_{i=1}^{M} A_i \right) u(t^n), \tag{1.17}$$

or for the implicit scheme, it is simpler if we choose the approximation:

$$u(t^{n+1}) = \left( I - \tau \sum_{i=1}^{M} A_i \right)^{-1} u(t^n), \tag{1.18}$$

where we modify the implicit additive scheme as follows:

$$u(t^{n+1}) = \frac{1}{M} \sum_{i=1}^{M} (I - M\tau A_i)^{-1} u(t^n). \tag{1.19}$$

The local splitting error of the simple explicit additive splitting method can be derived as

$$\begin{aligned} \mathrm{err}_{\mathrm{local}}(\tau_n) &= \left( \exp\left( \tau_n \sum_{i=1}^{M} A_i \right) - \left( I + \tau \sum_{i=1}^{M} A_i \right) \right) u_{\mathrm{sp}}^n \\ &= \frac{1}{2} \tau_n^2 || (A_1 + \cdots + A_M)^2 \, u_{\mathrm{sp}}^n || + \mathscr{O}(\tau_n^3), \end{aligned} \tag{1.20}$$

where the operators $A_1, \ldots, A_M$ are assumed to be bounded operators.

#### 1.3.1.2 Multiplicative Splitting Scheme

The idea of multiplicative splitting schemes are to decompose into a multiplicative series of operators.

So if we deal with a first-order scheme, we can decompose Eq. (3.71) into a series of

$$u(t) = \Pi_{i=1}^{M} \exp(t A_i) u(0), \ t \in (0, T), \tag{1.21}$$

or simpler, if we choose the approximation:

$$u(t) = \Pi_{i=1}^{M}(I + tA_i)u(0),\ t \in (0, T). \tag{1.22}$$

Algorithmically, we can describe the simplest splitting scheme as follows:

$$\frac{\partial u_1(t)}{\partial t} = A_1 u_1(t),\quad t \in (t^n, t^{n+1}),\quad \text{with } u_1(t^n) = u_{\text{sp}}^n, \tag{1.23}$$

$$\dots \tag{1.24}$$

$$\frac{\partial u_M(t)}{\partial t} = A_M u_M(t),\quad t \in (t^n, t^{n+1}),\quad \text{with } u_M(t^n) = u_{M-1}(t^{n+1}), \tag{1.25}$$

for $n = 0, 1, \dots, N-1$, where $u_{\text{sp}}^0 = u_0$ is given from (5.454). The approximate split solution at the point $t = t^{n+1}$ is defined as $u_{\text{sp}}^{n+1} = u_M(t^{n+1})$.

The local splitting error of the simple multiplicative splitting method can be derived as

$$\begin{aligned} \text{err}_{\text{local}}(\tau_n) &= \left(\exp\left(\tau_n \sum_{i=1}^{M} A_i\right) - \Pi_{i=1}^{M} \exp(\tau_n A_i)\right) u_{\text{sp}}^n, \\ &\le \frac{1}{2}\tau_n^2 \max_{j=1,\dots,M} \max_{\substack{i=1,\dots,M \\ i \neq j}} ||[A_i, A_j]\, u_{\text{sp}}^n|| + \mathcal{O}(\tau_n^3), \end{aligned} \tag{1.26}$$

where the operators $A_1, \dots, A_M$ are assumed to be bounded operators. The splitting time step is defined as $\tau_n = t^{n+1} - t^n$. We define $[A_i, A_j] := A_i A_j - A_j A_i$ as the commutator.

### *1.3.2 Iterative Splitting Methods*

Iterative splitting method underlies the iterative methods used to solve coupled operators using a fixed-point iteration. These algorithms integrate each underlying equation with respect to the last iterated solution. Therefore, the starting solution in each iterative equation is important in order to guarantee fast convergence or a higher order method. The last iterative solution should at least have a local error of $\mathcal{O}(\tau_n^i)$ ($i$th order in time), where $i$ is the number of iteration steps to obtain the next higher order.

One of the main motivations to apply iterative splitting method is to reach higher accuracy based on the iterated solutions, see [30]. Further, such scheme can relax nonlinearities based on the smoothing behaviour of fix-point approaches or successive approximation schemes, see [31, 32].

In the following sections, we concentrate on a basic so-called iterative splitting method, which is discussed in [30].

The algorithm is based on the iteration of a fixed sequential splitting discretization with step size $\tau_n$. On the time interval $[t^n, t^{n+1}]$, we solve the following subproblems consecutively for $i = 1, 3, 5, \cdots 2m + 1$.

$$\begin{aligned} \frac{du_i(t)}{dt} &= Au_i(t) + Bu_{i-1}(t), \\ & t \in (t^n, t^{n+1}), \quad \text{with } u_i(t^n) = u^n_{\text{sp}}, \end{aligned} \tag{1.27}$$

$$\begin{aligned} \frac{du_{i+1}(t)}{dt} &= Au_i(t) + Bu_{i+1}(t), \\ & t \in (t^n, t^{n+1}), \quad \text{with } u_{i+1}(t^n) = u^n_{\text{sp}}, \end{aligned} \tag{1.28}$$

where $u_0(t)$ is any fixed function for each iteration. The initial solution can be given as

- $u_0(t) = 0$ (we initialize with zero),
- $u_0(t) = u(t^n)$ (we initialize with the old solution at time $t^n$).

The iteration (1.27) and (1.28) for $i = 1, 3, \ldots, 2m + 1$ is consistent with an order of consistency $\mathcal{O}(\tau_n^{2m+1})$, see [30].

### 1.3.3 Application of the Operator Splitting Methods to Multiscale Problems

For the practical performing of the operator splitting methods, we have different criteria for a real problem.

In the following, we discuss two recipes of a practical application of different operator splitting methods, based on different assumptions to our real problems.

1. Physical Decomposition, see [16]: The real problem is derived and given as a model-equation based on a system of PDEs. Here, we consider only the model equations and are known of the physical background, e.g. transport parts of the model equations, etc.
2. Mathematical Decomposition, see [16]: The real problem is only given as a system of ODEs and we do not have the background, neither, e.g. semi-discretized PDEs, nor the physical background, e.g. equations based on reaction mechanics, etc.

#### Physical decomposition

In the physical decomposition, we decompose the multiscale problems based on the knowledge and information about the physical contributions (e.g. material laws, conservation laws, different physical behaviours of the equation operators), see [16].

Here, we can deal with two possibilities:

1. Direct Decoupling of the Multiscale equation:
   Here, we assume that we have a good overview and a good knowledge of all the parameters of the multiscale equation. That is, we assume directly the splitting into different operators, e.g. transport and reaction operators, strong anisotropy operator and isotropy operator, etc. Typical examples are of course the parabolic transport equations where we have the different operators of the transport part and the different operators of the reaction parts. We decompose with respect to these operators, see [33].
2. Indirect Decoupling of the Multiscale equation with respect to the underlying numerical methods:
   Here, the multiscale equations with their operators are not so obviously related to the physical behaviours, e.g. the coupling of each operator is strong and it is not obvious how to decompose them. In that case, we apply the discretization in time and space. Based on the underlying discretization methods, we are restricted to stability conditions, which are important to obtain stable solutions, see [34]. These additional conditions based on the schemes allow to couple the discretized operators to their physical parameters and we have a possible decomposition idea for the splitting schemes. One such important condition is the *Courant-Friedrichs-Lewy [CFL] condition*, see [35], which is important for all finite schemes to couple the time- and spatial-step with the underlying physical parameters. For explicit time-discretization schemes, we have an estimation of the next time step for stable numerical results. For implicit time-discretizations, we can limit ourselves to such a restriction to reduce numerical artefacts, which occur if we apply to large time-steps for implicit schemes, see [36].

*Example 1.2* For example, we deal with the transport-reaction equation:

$$\frac{\partial c}{\partial t} = v\frac{\partial c}{\partial x} + \lambda c, \text{ for } x \in \Omega,\ t \in [0, T], \tag{1.29}$$

$$c(x, 0) = c_0(x), \text{ for } x \in \Omega, \tag{1.30}$$

$$c(x, t) = g(x, t), \text{ for } x \in \partial\Omega,\ t \in [0, T], \tag{1.31}$$

where the velocity parameter is given as $v \in \mathbb{R}^+$, the reaction parameter is given as $\lambda \in \mathbb{R}^+$. After the space and time discretization with finite difference methods in time and space, we obtain the CFL conditions:

$$CFL = |\frac{v\ \tau_{flow}}{\Delta x}| \leq 1, CFL = |\lambda\ \tau_{react}| \leq 1, \tag{1.32}$$

where $\tau_{flow}$ is the time step for the flow and $\tau_{react}$ is the time step for the reaction. If we assume, for example,

$$\tau_{flow} \gg \tau_{react}, \tag{1.33}$$

we decompose the flow and the reaction part into different operators and can apply the splitting schemes, see [37].

**Mathematical decomposition**

In the mathematical decomposition method, we have only the information about the ODEs, i.e. we concentrate on the underlying operator.

Here, an indicator for a decomposition is the different eigenvalues of the operator.

Based on the different eigenvalues we can decompose the full operator in different operators with the same spectrum of the eigenvalues, see [33].

Here we can deal with different ideas:

1. Decomposition with respect to the maximal eigenvalues
   In this method, we assume an ordinary differential equation with different operators that are given by matrices.
   For each matrix $A_i,\ i = 1, \ldots, n$, we can estimate the spectral radius of a matrix $\rho(A_i)$.
   Based on the spectrum of the matrices, we derive the different operators for the splitting scheme.

*Example 1.3* For example, we assume to derive two operators where we define a $\rho_{stiff}$ and can separate the stiff and non-stiff parts of the operators to the new operators:

$$\tilde{A}_1 = \sum_{i \in I_1} A_i, \text{ where } i \in I_1 \text{ with } \rho(A_i) \leq \rho_{stiff}, \tag{1.34}$$

$$\tilde{A}_2 = \sum_{i \in I_2} A_i, \text{ where } i \in I_2 \text{ with } \rho(A_i) \geq \rho_{stiff}, \tag{1.35}$$

where $I_1 \cup I_2 = \{1, \ldots, n\}$.

*Remark 1.5* Often, it is important to decompose into stiff and non-stiff operators, while the stiff scales are finer than the non-stiff scales. The decomposition allows to have more appropriate time-steps for each operator and therefore saves computational time with respect to larger time steps.

2. Decomposition based on different norms of the operators
   Often, it makes sense to deal with different norms, e.g. maximum norm, $L_2$-norm of the operators and therefore give a classification of the underlying operators.
   Here, the idea is to select the operators with the same norm-behaviours and derive new operators for the splitting scheme, see [33].

*Remark 1.6* The mathematical decomposition can also be applied to one operator. Here, we split the full operator into sub-operators based on their different spectrums or norms, see [38].

## 1.4 Multiscale Analysis

In the following section, we deal with the multiscale analysis, which is in our case related to the ideas of separating the scales.

Therefore, we deal with multiscale methods to separate and decouple the scales, where we deal with the following ideas:

- Averaging (first-order perturbation theory, see [39]).
- Homogenization (second-order perturbation theory, see [39]).

Averaging and homogenization methods are often used with respect to oscillatory problems, while we could average or homogenize the high oscillations with respect to the slow scale.

*Example 1.4* We deal with the following modified example, see also the ideas in [20]:

$$\frac{dx}{dt} = \frac{1}{\varepsilon} f_1(y) + f_2(x, y), \tag{1.36}$$

$$\frac{dy}{dt} = g(x, y), \tag{1.37}$$

where $x$ is the fast and $y$ is the slow variable. Further, we assume $f_2$ and $g$ are periodic with respect to $y$, with period $[-\pi, \pi]$.

The approximated solutions are given as

$$y(t) \approx y_0(\tau) + \varepsilon y_1(\tau, t), \tag{1.38}$$

$$x(t) \approx x_0(t) + \varepsilon x_1(\tau, t), \tag{1.39}$$

where $\tau = \frac{t}{\varepsilon}$.

The leading terms are $x_0$ and $y_0$ and we have to assume that if $\tau \to \infty$, we have

$$\frac{y_1(\tau, t)}{\tau} \to 0, \tag{1.40}$$

$$\frac{x_1(\tau, t)}{\tau} \to 0. \tag{1.41}$$

We apply the approximated solutions to our differential equations and obtain the first leading-order equations $\mathcal{O}(\frac{1}{\varepsilon})$ and $\mathcal{O}(1)$:

$$\frac{dx_0}{d\tau}(\tau) - f_1(y_0(t)) = 0, \tag{1.42}$$

$$\frac{dy_0}{dt}(t) + \frac{\partial y_1}{\partial \tau}(\tau, t) - g(x_0(\tau), y_0(t)) = 0, \tag{1.43}$$

where we have for Eq. (1.45):

$$x_0(\tau) = \tau \; f_1(y_0) + \hat{x}_0, \tag{1.44}$$

where $\hat{x}_0$ is an initial condition. The second Eq. (1.43) is averaged over a large time interval $[0, \tau]$, with $\tau \to \infty$

$$\frac{dy_0}{dt}(t) = \lim_{\tau \to \infty} \int_0^{\tau} g(x_0(\tau'), y_0(t))\, d\tau' = \int_{-\pi}^{\pi} g(x_0(\tau'), y_0(t))\, d\tau', \tag{1.45}$$

and we have the averaged system of the slow variable $y_0$.

*Remark 1.7* The multiscale analysis in our consideration is based on averaging and homogenizing the disparate scales and upscale microscopic behaviours to the macroscopic model. Here are multiple timescale techniques, see [39, 40], important to recover a macroscopic model.

### *1.4.1 Analytical Methods*

One of the analytical methods is the method of multiple timescales, which is important to solve simultaneously different scales, see [23, 41].

The single time variable $t$ is replaced by a sequence of independent timescales $\varepsilon t, \varepsilon^2 t, \cdots$ and allows freedom degrees to solve multiscale problems.

We assume to have two different timescaled operators $A$ and $B$, while $||A|| \gg ||B|| \approx \varepsilon \to 0$.

$$\frac{\partial U(t)}{\partial t} = AU(t) \; + \; BU(t), \;\; \text{with} \;\; U(t^n) = U^n, \tag{1.46}$$

We derive a solution $U(t, \varepsilon)$ and apply:

$$U(t, \varepsilon) = U_0(t) + \varepsilon U_1(t) + \varepsilon^2 U_2(t) + \cdots + \varepsilon^J U_J(t), \tag{1.47}$$

with the initial conditions $U(0, \varepsilon) = U(0)$ and $J \in \mathbb{N}^+$ is a fixed iteration number.

Then the hierarchical equations are given as

$$\frac{\partial U_0(t)}{\partial t} = AU_0(t), \tag{1.48}$$

$$\frac{\partial U_1(t)}{\partial t} = AU_1(t) + \; BU_0(t), \tag{1.49}$$

$$\vdots$$

$$\frac{\partial U_J(t)}{\partial t} = AU_J(t) + BU_{J-1}, \tag{1.50}$$

and we have also to expand the initial conditions to $U_0(0) \; = \; U(0)$ and $U_j(0) \; = \; 0$, $\forall j = 1, \ldots, J$.

### 1.4.2 Multiscale Averaging

The multiscale averaging idea is based on the assumption that we can decouple the full model into two sub-models:

- Microscopic model (fast scales).
- Macroscopic model (slow scales).

The averaging method concentrates on two different scales:

- a set of fast variables (microscopic model),
- a set of slow variables (macroscopic model);

such separations allow to construct a more effective model for the slower scales by averaging the original or full model over the fast scales (or apply statistics for the fast variables).

*Remark 1.8* Here, we apply different techniques to average small time- or spatial scale and to embed into a model with larger time- or spatial scales, see [39].

### 1.4.3 Perturbation Methods

In the following, we discuss some ideas of perturbation problems that can also applied to multiscale problems.

- Homotopy perturbation method (HPM),
- Computational singular perturbation (CSP) method.

#### 1.4.3.1 Homotopy Perturbation Method

Perturbation methods are widely applied to solve nonlinear problems. The idea is to apply a homotopy technique, which means we could deform in a topology two continuous functions from one topological space to another. The deformation is called a homotopy between the two functions. Here, we apply a homotopy with an embedding of a small parameter $p \in [0, 1]$ and the method is called a homotopy perturbation method, see [42–44].

We deal with the following nonlinear differential, which can be decoupled into a linear and nonlinear part:

$$L(u) + N(u) - f(\mathbf{x}) = 0, \ \mathbf{x} \in \Omega, \tag{1.51}$$

$$G(u, \frac{\partial u}{\partial n}) = 0, \ \mathbf{x} \in \partial\Omega, \tag{1.52}$$

where $L$ is the linear operator (e.g. differential operator) and $N$ is the nonlinear operator, $f$ is a right-hand side, $G$ is a boundary operator.

In the following, we briefly introduce the homotopy technique proposed by Liao in [45], we can construct the following homotopy of Eq. (1.51) $v(\mathbf{x}, p) : \Omega \times [0, 1] \to \mathbb{R}$ which satisfies:

$$\mathscr{H}(v, p) = (1 - p)(L(v) - L(u_0)) + p(L(v) + N(v) - f(\mathbf{x})) = 0, \quad (\mathbf{x}, p) \in \Omega \times [0, 1], \tag{1.53}$$

or

$$\mathscr{H}(v, p) = L(v) - L(u_0) + pL(u_0) + p(N(v) - f(\mathbf{x})) = 0, \quad (\mathbf{x}, p) \in \Omega \times [0, 1], \tag{1.54}$$

where $p$ is the embedded parameter and $u_0$ is an initial approximation that satisfies the boundary conditions.

Further we have the results:

$$\mathscr{H}(v, 0) = L(v) - L(u_0), \tag{1.55}$$

$$\mathscr{H}(v, 1) = L(v) + N(v) - f(\mathbf{x}) = 0, \tag{1.56}$$

that result in a multiscale solution, see [46], if we apply $p = \varepsilon$, and we have

$$v = v_0 + pv_1 + p^2 v_2 + \cdots, \tag{1.57}$$

where the approximation of the solution of (1.51) is given as

$$u = \lim_{p \to 1} v = v_0 + v_1 + v_2 + \cdots. \tag{1.58}$$

We have the following first examples:

*Example 1.5* We deal with

$$y' + y^n = 0, \ x \geq 0, x \in \Omega \subset \mathbb{R}, \ y(0) = 1, \tag{1.59}$$

we apply the homotopy:

$$Y' - y_0' + py_0' + pY^n = 0, \ (x, p) \in \Omega \times [0, 1], \tag{1.60}$$

and we have the solution

$$Y = Y_0 + pY_1 + p^2 Y_2 + \cdots, \tag{1.61}$$

we derive the hierarchical equations as

$$p^0 : Y_0' = y_0', \tag{1.62}$$
$$p^1 : Y_1' + y_0' + Y_0^n = 0,\ Y_1(0) = 0, \tag{1.63}$$
$$p^2 : Y_2' + nY_0^{n-1}Y_1 = 0,\ Y_2(0) = 0, \tag{1.64}$$

and we have $Y_0 = y_0 = 1$. Then we obtain $Y_1 = -x$ and $Y_2 = \frac{n}{2}x^2$. The second-order approximation is given as

$$y = Y_0 + pY_1 + p^2Y_2 = 1 - px + p^2\frac{n}{2}x^2, \tag{1.65}$$

and for $p = 1$, we obtain

$$y = Y_0 + pY_1 + p^2Y_2 = 1 - x + \frac{n}{2}x^2. \tag{1.66}$$

The next example is related to a fast-slow dynamics.

*Example 1.6* We deal with

$$y' - y + y^2 = 0,\ x \geq 0, x \in \Omega \subset \mathbb{R},\ y(0) = 2, \tag{1.67}$$

we apply the homotopy:

$$Y' - Y - y_0' + y_0 + py_0' - py_0 + pY^2 = 0,\ (x, p) \in \Omega \times [0, 1], \tag{1.68}$$

and we have the solution

$$Y = Y_0 + pY_1 + p^2Y_2 + \cdots, \tag{1.69}$$

we derive the hierarchical equations as

$$p^0 : Y_0' - Y_0 - y_0' + y_0 = 0, \tag{1.70}$$
$$p^1 : Y_1' + y_0' - y_0 + Y_0^2 = 0,\ Y_1(0) = 0, \tag{1.71}$$
$$p^2 : Y_2' + 2Y_0Y_1 = 0,\ Y_2(0) = 0, \tag{1.72}$$

and for simplicity we have $Y_0 = y_0 = 2\exp(-x)$ (solution of $y_0' - y_0 = 0$). Then we obtain $Y_1 = 2\exp(-2x)$ and $Y_2 = \frac{8}{3}\exp(-3x)$. The second-order approximation is given as

$$y = Y_0 + pY_1 + p^2Y_2 = 2\exp(-x) + p^2\exp(-2x) + p^2\frac{8}{3}\exp(-3x), \tag{1.73}$$

and for $p = 1$, we obtain the approximation of the nonlinear differential equation (1.67).

*Remark 1.9* If we apply the perturbation of a fast-slow dynamics and we assume

$$y' - y + \varepsilon y^2 = 0,\ x \geq 0, x \in \Omega \subset \mathbb{R},\ y(0) = 2, \tag{1.74}$$

for $0 < \varepsilon \ll 1$, we obtain the solution based on the homotopy perturbation method (1.73) and we apply $p = \varepsilon$. Here the nonlinearity $y^2$ is the slow part of the equation.

### 1.4.4 Computational Singular Perturbation Method

The computational singular perturbation (CSP) method developed by Lam and Goussis [47] is an iterative method to reduce the dimensionality of differential equations with multiple timescales, see [48].

The idea is to decouple slow and fast scales and relax the fast scales to the slow scales, see [48].

We deal with a reaction system with $N$ species and $R$ reactions. The chemical kinetics equation is given as

$$\frac{d\boldsymbol{y}}{dt} = \boldsymbol{g}(\boldsymbol{y}), \tag{1.75}$$

where $\boldsymbol{g}$ is the global reaction rate, $\boldsymbol{y} = [y_1, \ldots, y_N]^T$ is the vector of the concentrations and the elementary reactions are given as

$$\boldsymbol{g}(\boldsymbol{y}) = \sum_{r=1}^{R} \boldsymbol{S}_r F^r, \tag{1.76}$$

where $\boldsymbol{S}_r$ is the stochiometric vector and $F^r$ is the reaction rate of the $r$th reactions.

The idea of the CSP method is to derive an ideal set of basis vectors for the derivation of the simplified models, e.g. decomposition into a slow and fast part of the reaction equations for example $\boldsymbol{g} = (\boldsymbol{g}_{slow}, \boldsymbol{g}_{fast})^T$.

One has to find a fixed basis $A \in \mathbb{R}^N$ with the relation:

$$\boldsymbol{g} = A\boldsymbol{f}, \tag{1.77}$$

and

$$\boldsymbol{f} = B\boldsymbol{g}, \tag{1.78}$$

with $BA = I \in \mathbb{R}^{N \times N}$.

Then we can focus on the dynamics of

$$\frac{d\boldsymbol{f}}{dt} = \lambda \boldsymbol{f}, \tag{1.79}$$

where $\Lambda$ is a linear operator which is given as $\Lambda = B(D\boldsymbol{g})A - B(DA)\boldsymbol{g} = B[A, \boldsymbol{g}]$ and $D\boldsymbol{g}$ is the Jacobian of $\boldsymbol{g}$.

Based on the decomposition into slow and fast parts, we have:

$$\Lambda = \begin{pmatrix} \Lambda^{1,1} & \Lambda^{1,2} \\ \Lambda^{2,1} & \Lambda^{2,2} \end{pmatrix} \tag{1.80}$$

$$= \begin{pmatrix} B^1[A_1, \boldsymbol{g}] & B^1[A_2, \boldsymbol{g}] \\ B^2[A_1, \boldsymbol{g}] & B^2[A_2, \boldsymbol{g}] \end{pmatrix}, \tag{1.81}$$

and due to CSP algorithm, we compute a block-orthogonalization and we obtain the decomposed matrix:

$$\Lambda = \begin{pmatrix} B^{s,\perp}[A_f, \boldsymbol{g}] & B^{s,\perp}[A_s, \boldsymbol{g}] \\ B^{f,\perp}[A_f, \boldsymbol{g}] & B^{f,\perp}[A_s, \boldsymbol{g}] \end{pmatrix} \tag{1.82}$$

$$= \begin{pmatrix} B^s[A_f, \boldsymbol{g}] & 0 \\ 0 & B^f[A_s, \boldsymbol{g}] \end{pmatrix}. \tag{1.83}$$

Now we have a decomposition into a slow and fast regime:

$$\frac{d\boldsymbol{f}_{slow}}{dt} = \Lambda^{1,1} \boldsymbol{f}_{slow}, \tag{1.84}$$

$$\frac{d\boldsymbol{f}_{fast}}{dt} = \Lambda^{1,1} \boldsymbol{f}_{fast}, \tag{1.85}$$

the detailed ideas of the algorithm are given in [47].

*Remark 1.10* We can compare the perturbation method with the averaging and homogenization method. While the averaging method is a first-order perturbation and the homogenization method is a second-order perturbation, see [39], the singular perturbation method is constructed to take into account the different oscillatory scales and separate them into different operators (matrices). Such different matrices can be applied or skipped in the computations, see [47].

*Remark 1.11* In general, the CSP method is only one method for automatic model reduction (slow-fast decomposition) of dynamical systems. We have also other alternative model reduction methods of multiscale systems. In the following, we present some of the recent publications of such methods.

### *1.4.5 Alternative Modern Systematic Model Reduction Methods of Multiscale Systems*

In the past years, many modern systematic model reduction methods of multiscale systems were developed.

In the following, we give a small overview of such alternative methods with respect to the discussed CSP method:

1. The Intrinsic Low-Dimensional Manifold (ILDM) (see the seminal paper [49]); here the idea is based on simplifying chemical kinetics based on the dynamical systems approach. The variables to the procedure are the detailed kinetics mechanism and the number of degrees of freedom required in the simplified scheme. The dynamical system approach is used to develop a scheme that reduces the state space of a reaction system globally in such a way that it can be tabulated for subsequent use in turbulent combustion calculations.
2. The Reaction–Diffusion Manifolds (REDIMs) approach, see the ideas Bykov and Maas [50] and more details are discussed in the papers [51–54]. The idea of the method is based on the decomposition of timescales. They assume an existence of invariant slow manifolds in the thermo-chemical composition space (state space) of a reacting flow, such that they can predict a detailed dynamical system. A manifold of the reduced model can be approximated by applying an invariance condition together with repeated integrations of the reduced model in an iterative way. At the end, they can derive a full stationary system dynamics governed by detailed chemical kinetics and the molecular transport in the case of a one-dimensional reduced model, which is also the limiting case, see [50].
3. The relaxation redistribution method (RRM), which is discussed in the papers and books of Chiavazzo et al., see [55–59], is based on the construction of accurate discrete approximations of slow invariant manifolds.
   The ideas are based on two steps:

   - Method of Invariant Manifold (MIM), see [60], where we have given a autonomous system:

$$\frac{d\phi}{dt} = f(\phi), \tag{1.86}$$

     where $\phi \in U$ is the state, $U$ is the phase space and $f$ is the reaction mechanism. The MIM is based on the idea to reconstruct a slow invariant manifold $\Omega$, which is embedded in $U$ and we have a function $F(\xi)$, which maps the macroscopic state into the microscopic state. The SIM (slow invariant manifold) is seen as a stable fixpoint of the film equation:

$$\frac{dF(\xi)}{dt} = f(F(\xi)) - P(f(F(\xi))), \tag{1.87}$$

     where $P$ is the projection onto the tangent space of the manifold $\Omega$.
   - Relaxation method based on the RRM (Relaxation Redistribution Method): The reconstruction of a macro-state $\xi$ into a micro-state $F(\xi)$, i.e. the practical application of Eq. (1.87), is based on a grid-refinement of two steps:
     a. Relaxation: The grid nodes are relaxed to the slow invariant manifold.
     b. Redistribution: The relaxed states are redistributed on a grid related to the parameter $\xi$.

     The idea of the algorithm is presented in Fig. 1.3.

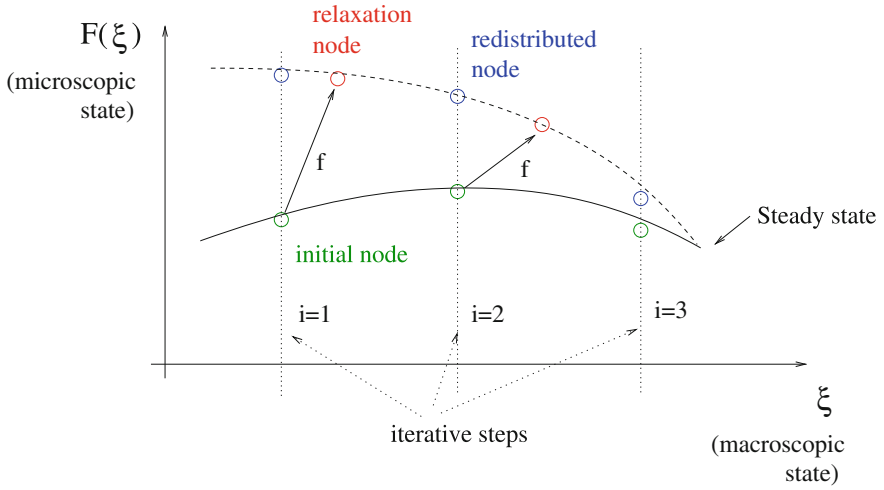



**Fig. 1.3** Grid approximation of the relaxation redistribution method

4. The G-scheme, which is discussed in [61], the main idea is based on resolving only a range of active time scales and neglect very-slow and very-fast timescales. Therefore, a general idea of the numerical solution is obtained using practical error tolerances, by which a numerical solution is approximating an exact solution. The algorithm is based on applying an adaptive coordinate transformation, such that it is possible to consider the modes (eigenvalues) of the solutions which, within a threshold criterion, can be frozen or are given in an equilibrium. The G-scheme applies such a transformation and deals only with the extracted reduced active system.
5. The method of Invariant Grid (MIG), which is discussed in [62, 63], is based on the model reduction concept of slow manifolds (SIM). The MIG algorithm is based on the idea to approximate the SIM by a set of grid nodes in the invariant grid (concentration space). The MIG can be used as a computational realization of the method of Invariant Manifolds (MIM), given in Eq. (1.87) for the SIM.
6. The Invariant Constrained-equilibrium Edge Pre-Image Curve (ICE-PIC) approach is discussed by S. Pope et al., see [64]. It is a method to simplify chemical kinetics based on dimension-reduction ideas. The dimension-reduction method is based on the idea to construct a low-dimensional manifold (explicitly or implicitly) in the full composition space. Here, the ICE-PIC method employed the low-dimensional manifold an invariant, trajectory-generated manifold. In addition, the ICE-PIC method applies the species reconstruction locally on the low-dimensional invariant manifold. Such a technique allows a more detailed reconstruction and the invariant manifold exists and is continuous.
7. The various variational approaches by D. Lebiedz, see [65], are also automatic procedures to replace higher dimensional dynamics by lower dimensional approximation with given error estimates to the original solution. The variational approach applies the idea to minimalize the entropy production. That is, one can

concentrate on a remaining system dynamics, while the other species concentrations of the system are close to the attractor and are maximally relaxed. This relaxation can be given as a minimal entropy production for the single reaction steps along their phase space trajectories.

### 1.4.6 Multiscale Expansion (Embedding of the Fast Scales)

Here the motivation arose to modify and reduce a multiscale equation with respect to its fast scale to an averaged or homogenized equation, see [39]. These techniques allow to embed the fast scale, while for long times such fast scales are averaged or homogenized and can be embedded to the slow scales. That is, we can reduce the delicate multiscale equations to simpler scale equations, taking into account averaging and centering ideas.

In the following, we start with a simple example to show the ideas of such schemes and at the end we present some recipes to apply such schemes to parabolic PDEs.

*Example 1.7* We deal with a advection-diffusion equation, given as

$$\frac{\partial u}{\partial t} = a\nabla u + D\Delta u, \text{ for } (x,t) \in \mathbb{R}^d \times \mathbb{R}^+, \tag{1.88}$$

$$u(x,0) = f(\varepsilon x), \tag{1.89}$$

with $0 < \varepsilon \leq 1$, where $a(x)$ is assumed to be smooth and periodic in space with period 1. We assume to see only a slow behaviour in the solution related to the convection or diffusion term and we therefore embed the fast solutions to a simplified equation.

- We deal first with diffusion-dominant scaling (diffusion scaling) and assume to skip the advection term by a centering condition.
  We apply the rescaling: $x \to \varepsilon^{-1}x$ and $t \to \varepsilon^{-2}t$.
  Hence, we have the multiscale expansion given as $u_\varepsilon(x) = u_0 + \varepsilon u_1(x, \frac{x}{\varepsilon}) + \mathcal{O}(\varepsilon^2)$, and we obtain

$$\frac{\partial u_\varepsilon}{\partial t} = \frac{1}{\varepsilon} a_\varepsilon \nabla u_\varepsilon + D\Delta u_\varepsilon, \text{ for } (x,t) \in \mathbb{R}^d \times \mathbb{R}^+, \tag{1.90}$$

$$u_\varepsilon(x,0) = f(\varepsilon x). \tag{1.91}$$

  Further we assume that the fast scale $a_\varepsilon = a(\frac{x}{\varepsilon})$.
  We define the operator as

$$L_0 = a(y) \cdot \nabla_y + D\Delta_y, \tag{1.92}$$

  with the periodic boundary conditions $[0,1]^d$ and can refer to the characteristics of the equation where the operator $L_0$ is the generator of the Markov process $y(t)$ and we have

$$\frac{dy}{dt} = a(y) + \sqrt{2D}\frac{dW}{dt}, \tag{1.93}$$

with periodic boundary conditions and $W(t)$ is a standard Brownian motion and solved on the unit torus $\mathbb{T}^d$.
Here, we can define the invariant distribution $\xi(y)$, see [39], as a stationary solution to the adjoint equation:

$$L_0^*\xi = 0. \tag{1.94}$$

We assume that the vector field $a(y)$ satisfies the centering condition to the invariant distribution:

$$\int_{\mathbb{T}^d} a(y)\xi(y)\, dy = 0, \tag{1.95}$$

such that it is averaged out. At least, we only see the diffusive behaviour.

The simplified equation in a first order is given as

$$\frac{\partial u}{\partial t} = D\nabla_x\nabla_x u, \text{ for } (x,t) \in \mathbb{R}^d \times \mathbb{R}^+, \tag{1.96}$$

$$u(x,0) = f. \tag{1.97}$$

Here, we have assumed that we can average out the advection term.

- Next, we deal with the advection-dominant scaling (diffusion scaling) and assume to deal with a pure advection equation, while we can neglect the influence of the diffusion term, e.g. for the divergence-free flows, see [39].
We apply the rescaling: $x \to \varepsilon^{-1}x$ and $t \to \varepsilon^{-1}t$.
Hence, we have the multiscale expansion given as $u_\varepsilon(x) = u_0 + \varepsilon u_1(x, \frac{x}{\varepsilon}) + \mathcal{O}(\varepsilon^2)$, and we obtain

$$\frac{\partial u_\varepsilon}{\partial t} = a_\varepsilon \nabla u_\varepsilon + \varepsilon D\Delta u_\varepsilon, \text{ for } (x,t) \in \mathbb{R}^d \times \mathbb{R}^+, \tag{1.98}$$

$$u_\varepsilon(x,0) = f(\varepsilon x). \tag{1.99}$$

Further we assume that the fast scale $a_\varepsilon = a(\frac{x}{\varepsilon})$.
We define the operator as for the previous diffusion dominant scaling

$$L_0 = a(y)\cdot\nabla_y + D\Delta_y, \tag{1.100}$$

with the periodic boundary conditions.
Further, we can define the invariant distribution $\xi(y)$ to the adjoint equation as

$$L_0^*\xi = 0, \tag{1.101}$$

and we assume that the vector field $a(y)$ is not averaged out and we have

$$\tilde{a} = \int_{\mathbb{T}^d} a(y)\xi(y)\, dy. \tag{1.102}$$

The simplified equation in a first order is given as

$$\frac{\partial u}{\partial t} = \tilde{a} \cdot \nabla_x u, \tag{1.103}$$

$$u(x, 0) = f. \tag{1.104}$$

Here, we deal with a dominant advection part and we have a linear transport equation.

We apply a next problem given in the impact oscillator problem, see [66].

*Example 1.8* The Fokker-Planck equations are given as

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} - E(x)\frac{\partial f}{\partial v} = \frac{\partial}{\partial v}\left(-\gamma v f + \beta^{-1}\gamma\frac{\partial f}{\partial v}\right), \tag{1.105}$$

where we could decouple such an FP equation into the PIC (particle in cell) part and the SDE part.

- PIC part

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} - E(x)\frac{\partial f}{\partial v} = 0, \tag{1.106}$$

- SDE part

$$\frac{\partial f}{\partial t} = \frac{\partial}{\partial v}\left(-\gamma v f + \beta^{-1}\gamma\frac{\partial f}{\partial v}\right), \tag{1.107}$$

where we solve the characteristics:

- PIC part

$$\frac{dx}{dt} = v, \tag{1.108}$$

$$\frac{dv}{dt} = -E(x) = \frac{\partial U}{\partial x}, \tag{1.109}$$

where $U$ is the potential.

- SDE part

$$\frac{dx}{dt} = 0, \tag{1.110}$$

$$dv = -\gamma v dt + \sqrt{2\beta^{-1}\gamma}dW, \tag{1.111}$$

where we assume $\gamma = \frac{1}{\varepsilon}$ and $\beta^{-1}\gamma \approx const.$

When we apply the rescaling: $v \to v/\varepsilon$ and $t \to t/\varepsilon^2$, then the SDE part is

$$dv = -\frac{1}{\varepsilon} v dt + \sqrt{2D} dW, \tag{1.112}$$

and our Eq. (1.111) has at least a diffusive behaviour:

$$dv = \sqrt{2D} dW. \tag{1.113}$$

*Remark 1.12* The benefit of multiscale expansion is to decompose the full equations into simpler equations, while for the rescaling, we can neglect some parts of the equations. Such a rescaling or multiscale expansion allows to derive simpler equations, which fulfil for the assumption (e.g. splitting, averaging or homogenization) the dominant behaviour of the full equations.

In Figs. 1.4 and 1.5, we present some methods to apply such multi-expansion methods.

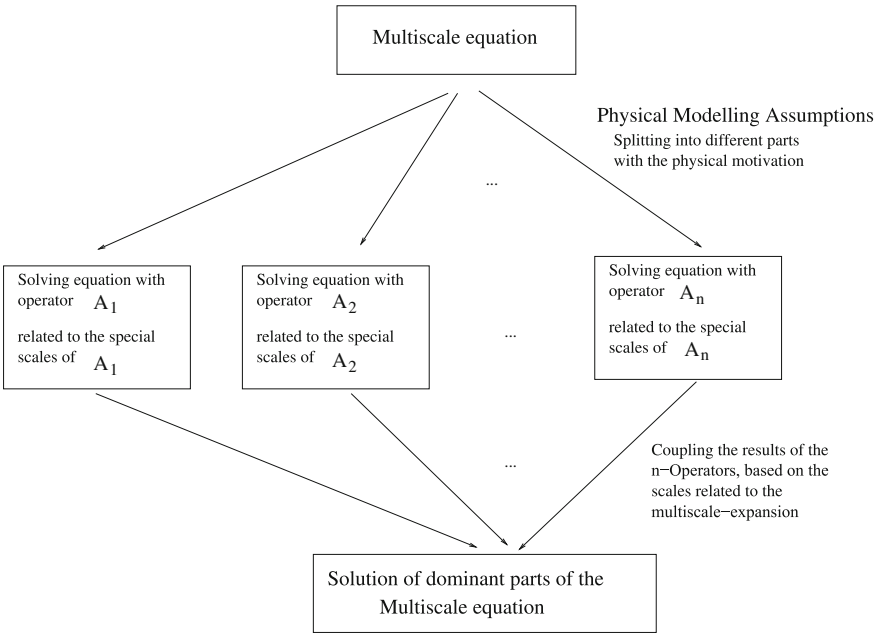



**Fig. 1.4** Application of splitting methods

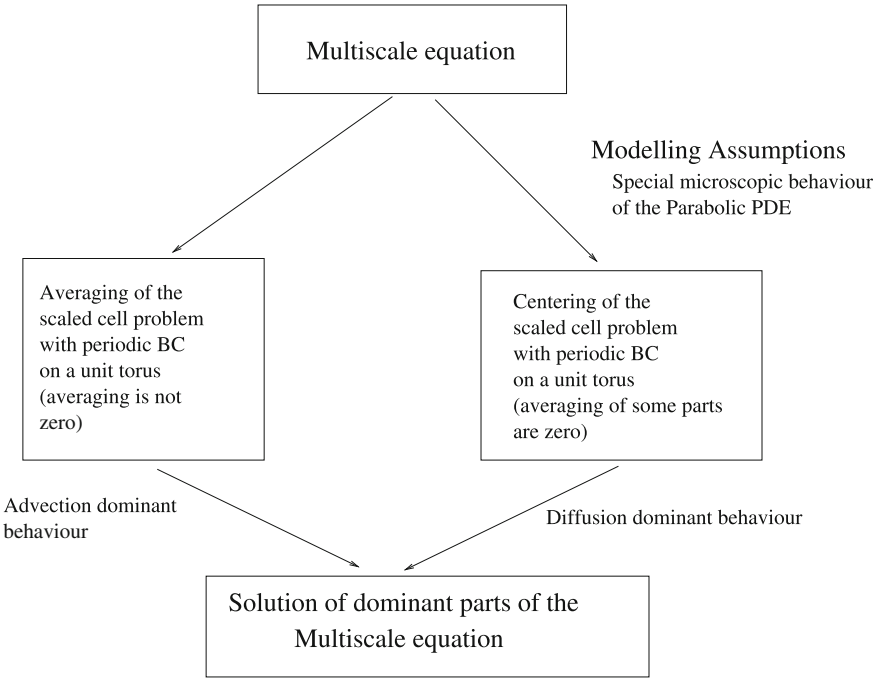



**Fig. 1.5** Application of averaging and homogenization methods

*Remark 1.13* For all methods it is important to extract some dominances of the multiscale equations, while the multiscale expansion can give some hints to these parts. While in splitting methods often the physical background motivates to decompose the parts of the multiscale equations, the averaging and homogenization methods are motivated to deal with the underlying microscopic problem and embed that part into the upscaled equations. Both methods have at least some neglections and we derive a reduced multiscale equation which extracts the dominant solutions of the slower scales.

## References

1. A. Ern, V. Giovangigli, *Multicomponent Transport Algorithms*. Lecture Notes in Physics Monographs, vol. M24 (1994)
2. C. Crowe, *Multiphase Flow Handbook* (CRC Press, Boca Raton, 2005)
3. V. Giovangigli, *Multicomponent Flow Modeling* (Birkhäuser, Boston, 1999)
4. V.M. Zhdanov, *Transport Processes in Multicomponent Plasma* (CRC Press, Boca Raton, 2002)
5. V. Giovangigli, Multicomponent flow. Scholarpedia **9**(4), 11930 (2014). http://www.scholarpedia.org/article/Multicomponent_Flow
6. K.R. Westerterp, W.P.M. Van Swaaij, A.A.C.M. Beenackers, *Chemical Reactor Design and Operation* (Wiley, New York, 1988)

7. E.S. Oran, J.P. Boris, *Numerical Simulation of Reactive Flow* (Cambridge University Press, Cambridge, 2000)
8. T. Poinsot, D. Veynante, *Theoretical and Numerical Combustion*, 2nd edn. (R.T. Edwards Inc., Philadelphia, 2005)
9. J. Geiser, *Models and Simulation of Deposition Processes with CVD Apparatus*. Monograph, Series: Groundwater Modelling, Management and Contamination (Nova Science Publishers, New York, 2009)
10. J. Geiser, M. Arab, *Simulation of Deposition Processes with PECVD Apparatus* (Nova Science Publishers, Inc., Huntington, 2012)
11. Z. Zlatev, *Computer Treatment of Large Air Pollution Models* (Kluwer Academic Publishers, Dordrecht, 1995)
12. C. Brennen, *Fundamentals of Multiphase Flow* (Cambridge University Press, Cambridge, 2005)
13. Wikipedia Reference: Multiphase Flow. http://en.wikipedia.org/wiki/Multiphase_flow. Wikipedia, June 2014
14. V.B. Zalesny, G.I. Marchuk, V.I. Agoshkov, A.V. Bagno, A.V. Gusev, N.A. Diansky, S.N. Moshonkin, R. Tamsalu, E.M. Volodin, Numerical simulation of large-scale ocean circulation based on the multicomponent splitting method. Russ. J. Numer. Anal. Math. Model. **25**(6), 581–609 (2010)
15. J. Geiser, Multiscale splitting method for the Boltzmann-Poisson equation: application to the dynamics of electrons. Int. J. Differ. Equ. **2014**, Article ID 178625, 8 pp. (2014)
16. J. Geiser, in *Decomposition Methods for Partial Differential Equations: Theory and Applications in Multiphysics Problems*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, F. Lai (CRC Press, Chapman & Hall/CRC, Boca Raton, 2009)
17. R.I. McLachlan, G.R.W. Quispel, Splitting methods. Acta Numer. **11**, 341–434 (2002)
18. J. Geiser, Numerical methods of the Maxwell-Stefan diffusion equations and applications in plasma and particle transport. Preprint, arxiv:1501.05792, January 2015
19. N. Antonic, C.J. van Duijn, W. Jäger, A. Mikelic, Multiscale problems in science and technology: challenges to mathematical analysis and perspectives in *Proceedings of the Conference on Multiscale Problems in Science and Technology*, Dubrovnik, Croatia, 3–9 September 2000, ed. by N. Antonic, C.J. van Duijn, W. Jäger, A. Mikelic (Springer, Berlin, 2002)
20. W.E. *Principle of Multiscale Modelling* (Cambridge University Press, Cambridge, 2010)
21. J. Geiser, Iterative splitting methods for multiscale problems, in *Distributed Computing and Applications to Business, Engineering Science (DCABES)* 2–4 September 2013, London, UK (2013), pp. 3–6
22. J. Geiser, in *Coupled Systems: Theory, Models and Applications in Engineering*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, F. Lai (CRC Press, Chapman & Hall/CRC, Boca Raton, 2014)
23. R.S. Johnson, *Singular Perturbation Theory* (Springer, New York, 2005)
24. W.E.B. Engquist, X. Li, W. Ren, E. Vanden-Eijnden, Heterogeneous multiscale methods: a review. Commun. Comput. Phys. **2**(3), 367–450 (2007)
25. I.G. Kevrekidis, G. Samaey, Equation-free multiscale computation: algorithms and applications. Annu. Rev. Phys. Chem. **60**, 321–344 (2009)
26. J. Geiser, Recent advances in splitting methods for multiphysics and multiscale: theory and applications. J. Algorithms Comput. Technol., Multi-Sci., Brentwood, Essex, UK, accepted August 2014 (to be published second issue 2015)
27. L. Rosso, A.F de Baas, Review of materials modelling: what makes a material function? Let me compute the ways ... European Commision, General for Research and Innovation Directorate, Industrial Technologies, Unit G3 Materials (2014) http://ec.europa.eu/research/industrial_technologies/modelling-materials_en.html
28. C.W. Gear, J.M. Hyman, P.G. Kevrekidid, I.G. Kevrekidis, O. Runborg, C. Theodoropoulos, Equation-free, coarse-grained multiscale computation: enabling microscopic simulators to perform system-level analysis. Commun. Math. Sci. **1**(4), 715–762 (2003)

29. H. Kim, *Multiscale and Multiphysics Computational Frameworks for Nano- and Bio-Systems*. Springer Theses (Springer, Heidelberg, 2011)
30. J. Geiser, in *Iterative Splitting Methods for Differential Equations*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, F. Lai (Chapman & Hall/CRC, Boca Raton, 2011)
31. O. Nevanlinna, Remarks on Picard-Lindelöf iteration, part I. BIT **29**, 328–346 (1989)
32. C.T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*. SIAM Frontiers in Applied Mathematics, vol. 16. SIAM, Philadelphia (1995)
33. J. Geiser, Discretization methods with analytical characteristic methods and applications. M2AN EDP Sci. Fr. **43**(6), 1157–1183 (2009)
34. A. Quarteroni, A. Valli, *Numerical Approximation of Partial Differential Equations*. Springer Series in Computational Mathematics (Springer, Berlin, 1997)
35. R. Courant, K.O. Friedrichs, H. Lewy Collatz, Über die partiellen Differenzengleichungen der mathematischen Physik. Math. Ann. **100**, 32–74 (1928)
36. W. Hundsdorfer, J.G. Verwer, *Numerical Solution of Time-dependent Advection-diffusion-Reaction Equations* (Springer, Berlin, 2003)
37. A. Quarteroni, A. Valli, *Domain Decomposition Methods for Partial Differential Equations*. Series: Numerical Mathematics and Scientific Computation (Clarendon Press, Oxford, 1999)
38. L.N. Trefethen, M. Embree, *Spectra and Pseudospectra: The Behaviour of Nonnormal Matrices and Operators* (Princeton University Press, Princeton, 2005)
39. G.A. Pavliotis, A.M. Stuart, *Multiscale Methods: Averaging and Homogenization* (Springer, Heidelberg, 2008)
40. J. Cronin, R.E. O'Malley, Analyzing Multiscale Phenomena Using Singular Perturbation Methods. American Mathematical Society Short Course, 5–6 January, 1998, Baltimore, Maryland, ed. by J. Cronin, R.E. O'Malley (1999)
41. P. Jakobsen, Introduction to the method of multiple scales (2014). arXiv:1312.3651
42. S.J. Liao, An optimal homotopy-analysis approach for strongly nonlinear differential equations. Commun. Nonlinear Sci. Numer. Simul. **15**, 2003–2016 (2010)
43. S.J. Liao, On the homotopy analysis method for nonlinear problems. Appl. Math. Comput. **147**, 499–513 (2004)
44. J.-H. He, Homotopy perturbation method: a new nonlinear analytical technique. J. Appl. Math. Comput. **135**(1), 73–79 (2003)
45. S.J. Liao, An approximate solution technique which does not depend upon small parameters: a special example. Int. J. Non-Linear Mech. **30**, 371–380 (1995)
46. J. Kevorkian, J.D. Cole, *Multiple Scale and Singular Perturbation Methods* (Springer, New York, 1996)
47. S.H. Lam, D.A. Goussis, The CSP method for simplifying kinetics. Int. J. Chem. Kinet. **26**, 461–486 (1994)
48. A. Zagaris, H.G. Kaper, T.J. Kaper, Fast and slow dynamics for the computational singular perturbation method. Multiscale Model. Simul. **2**(4), 613–638 (2004)
49. U. Maas, S.B. Pope, Simplifying chemical kinetics: intrinsic low-dimensional manifolds in composition space. Combust. Flame **88**, 239–264 (1992)
50. V. Bykov, U. Maas, Problem adapted reduced models based on Reaction Diffusion Manifolds (REDIMs). Institut für Technische Thermodynamik, Karlsruhe University, Kaiserstra"se 12, D-76128 Karlsruhe, Germany, Proceedings of the Combustion Institute 01/2009 (2009)
51. U. Maas, V. Bykov, The extension of the reaction/diffusion manifold concept to systems with detailed transport models, in *Proceedings of The Combustion Institute—PROC COMBUST INST 01/2011* (2011)
52. V. Bykov, A. Neagos, U. Maas, On transient behavior of non-premixed counter-flow diffusion flames within the REDIM based model reduction concept. Proc. Combust. Inst. **34**(1), 197–203 (2013)
53. V. Bykov, U. Maas, The extension of the ILDM concept to reaction diffusion manifolds. Combust. Theory Model. **11**(6), 839–862 (2007)

54. K. König, V. Bykov, U. Maas, Investigation of the dynamical response of methane-air counterflow flames to inflow mixture composition and flow field perturbations. Flow Turbul. Combust. **83**(1), 105–129 (2009)
55. E. Chiavazzo, I. Karlin, Adaptive simplification of complex multiscale systems. Phys. Rev. E **83**, 036706 (2011)
56. E. Chiavazzo, Approximation of slow and fast dynamics in multiscale dynamical systems by the linearized Relaxation Redistribution Method. J. Comput. Phys. **231**(4), 1751–1765 (2012)
57. E. Chiavazzo, P. Asinari, F. Visconti, Fast computation of multi-scale combustion systems. Philos. Trans. R. Soc. A **369**, 2396–2404 (2011)
58. E. Chiavazzo, I.V. Karlin, Adaptive simplification of complex systems: a review of the relaxation redistribution approach, in *Coping with Complexity: Model Reduction and Data Analysis*, ed. by A. Gorban, D. Roose (Springer, Berlin, 2011), pp. 231–240
59. M. Kooshkbaghi, C.E. Frouzakis, E. Chiavazzo, K. Boulouchos, I.V. Karlin, The global relaxation redistribution method for reduction of combustion kinetics. J. Chem. Phys. **141**, 044102 (2014)
60. A.N. Gorban, I.V. Karlin, *Invariant Manifolds for Physical and Chemical Kinetics*. Lecturer Notes in Physics, vol. 660, (Springer, Berlin, 2005)
61. M. Valorani, S. Paolucci, The G-scheme: a framework for multi-scale adaptive model reduction. J. Comput. Phys. **228**(13), 4665–4701 (2009)
62. E. Chiavazzo, I.V. Karlin, K. Boulouchos, Method of invariant grid for model reduction of hydrogen combustion. Proc. Combust. Inst. **32**(1), 519–526 (2009)
63. E. Chiavazzo, I.V. Karlin, A.N. Gorban, K. Boulouchos, Coupling of the model reduction technique with the lattice Boltzmann method for combustion simulations. Combust. Flame **157**(10), 1833–1849 (2010)
64. Z. Ren, S.B. Pope, A. Vladimirsky, J.M. Guckenheimer, Application of the ICE-PIC method for the dimension reduction of chemical kinetics coupled with transport. Proc. Combust. Inst. **31**, 473–481 (2007)
65. D. Lebiedz, Computing minimal entropy production trajectories—an approach to model reduction in chemical kinetics. J. Chem. Phys. **120**, 6890 (2004)
66. N. Bou-Rabee, E. Vanden-Eijnden, A patch that imparts unconditional stability to explicit integrators for Langevin-like equations. J. Comput. Phys. **231**, 2565–2580 (2012)

# Chapter 2
# Theoretical Part: Functional Splitting

**Abstract** We describe a general method, which is based on a splitting approach and the knowledge of the exact solutions of some sub-problems. Such additional information is taken into account and has an important role in accelerating the computations. We apply a functional splitting idea to decompose the initial problem into several sub-problems where some of them are known with the analytical solutions. The sub-problems with unknown solutions are solved numerically by standard numerical methods, e.g. finite volume methods. This paper can be divided into four parts. In the first part, we introduce the model and its application. In the second part, we discuss the analytical solutions of coupled systems of convection-reaction equations. Functional splitting methods are developed in the third part.

## 2.1 Ideas of the Functional Splitting

The ideas of functional splitting are applied in different areas of decomposing multicomponent flow problems, see [1, 2].

The motivation is to reduce the problems of solving reacting flows whose complexity comes from the fact of a wide range of timescales.

Such complexity leads to numerical difficulties related, e.g. to stiffness of the reaction terms.

Here, the idea is to split the model equations additively into flow terms (e.g. advective transport, diffusive transport) and reaction terms (e.g. chemical transformations).

In the following, we discuss the different splitting techniques, that are applied in multi-component flow problem, see [3].

### *2.1.1 Flow Equations*

We deal with a system of flow equations, which are coupled by the different flow operators, e.g. advection, diffusion, dispersion, etc. Here the main ideas are to decompose such delicate multi-operator equation into simpler one-operator equations.



J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5_2

Therefore, we can treat each simpler one-operator equation with more adequate solver and discretization schemes and optimize their computational time. Splitting techniques allow to decompose the operators and couple the results of each simpler operator equation together to the full result, e.g. with overlaps in the initialization of each simpler operator equation (initial condition coupling).

#### 2.1.1.1 Splitting of Physical Processes

So one splitting technique is based on the idea to decompose the discretized operator

$$\frac{\partial u}{\partial t} + Au = f,\ t \in [0, T], \tag{2.1}$$

where $A = \sum_{j=1}^{I} A_j$, $A_j \geq 0$ ($A_j$ is positive definite) and $f = \sum_{j=1}^{I} f_i$ and $i = 1, 2, \ldots, I$.

The solution of the simpler equations are given as:

$$\frac{u^{j+1/I} - u^j}{\Delta t} + A_1(\alpha u^{j+1/I} + (1 - \alpha u^j)) = f_1,\ t \in [0, T], \tag{2.2}$$

$$\frac{u^{j+2/I} - u^{j+1/I}}{\Delta t} + A_1(\alpha u^{j+2/I} + (1 - \alpha u^{j+1/I})) = f_2,\ t \in [0, T], \tag{2.3}$$

$$\vdots \tag{2.4}$$

$$\frac{u^{j+1} - u^{j+(I-1)/I}}{\Delta t} + A_1(\alpha u^{j+1} + (1 - \alpha u^{j+(I-1)/I})) = f_I,\ t \in [0, T], \tag{2.5}$$

where for $\alpha = 1$ is an implicit scheme of first order, $\alpha = 0$ is an explicit scheme of first order and for $\alpha = 1/2$ we have a Crank–Nicolson scheme of second order, see [4].

#### 2.1.1.2 Splitting of Physical Processes and Solution Components

Another splitting idea is based on splitting the components of the solutions with respect to their different scales, e.g. vertical and horizontal velocity in ocean circulation or decompose the velocity field into a time average motion and a turbulent fluctuation (Reynolds-averaging idea, see [5]).

The idea is based on two different velocity scales, i.e. a fast scale (turbulent fluctuation) and a slow scale (averaged motion), see Fig. 2.1.

We decompose the velocity into:

$$u = \overline{u} - u', \tag{2.6}$$

where $\overline{u} = \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} u(s)\, ds$ and $\Delta t = t^{n+1} - t^n$.

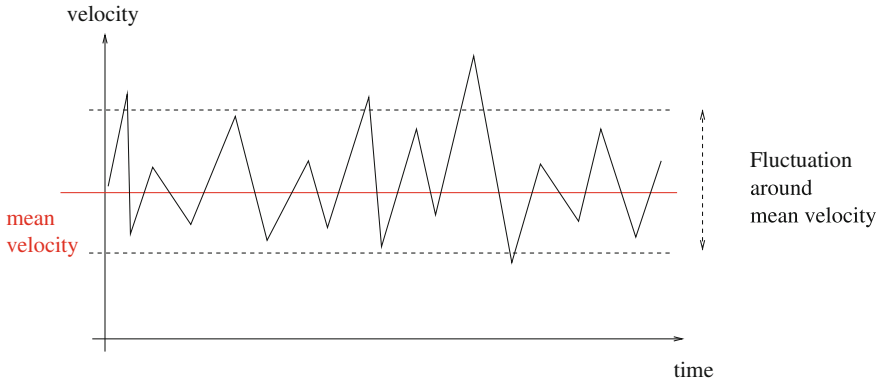



**Fig. 2.1** Splitting approach to convection-diffusion-reaction equations

*Example 2.1* Decomposition of a turbulent flow into an averaged flow and fluctuation flow. Such an application is known in the Navier–Stokes simulations, see [6].

We apply a flow-equation given with two flow variables $u$, $v$ and have:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}(uv) = Q_u, \tag{2.7}$$

where $Q_u$ is a source of $u$ and we have the following decomposition:

$$u = \overline{u} + u', \tag{2.8}$$
$$v = \overline{v} + v', \tag{2.9}$$
$$S_u = \overline{S_u} + S_u', \tag{2.10}$$

and we decompose into

$$\frac{\partial(\overline{u} + u')}{\partial t} + \frac{\partial}{\partial x}((\overline{u} + u')(\overline{v} + v')) = \overline{Q_u} + Q_u', \tag{2.11}$$

and we have:

$$\frac{\partial(\overline{u} + u')}{\partial t} + \frac{\partial}{\partial x}((\overline{uv} + \overline{u}v' + u'\overline{v} + u'v')) = \overline{Q_u} + Q_u', \tag{2.12}$$

we apply the averaging operator and have:

$$\frac{\partial\overline{(\overline{u} + u')}}{\partial t} + \frac{\partial}{\partial x}(\overline{(\overline{uv} + \overline{u}v' + \overline{u'\overline{v}} + \overline{u'v'})}) = \overline{\overline{Q_u} + Q_u'}, \tag{2.13}$$

then, we skip the fast perturbations means $\overline{u'} = 0$, $\overline{S'_u}$ and obtain:

$$\frac{\partial \overline{u}}{\partial t} + \frac{\partial}{\partial x}(\overline{\overline{u}\overline{v}} + \overline{\overline{u}v'} + \overline{u'\overline{v}} + \overline{u'v'})) = \overline{Q_u}, \tag{2.14}$$

then based on the continuity equation we have $\frac{\partial \overline{u}}{\partial x} = 0$ and $\frac{\partial \overline{v}}{\partial x} = 0$ such that we can skip the mixed terms and we obtain:

$$\frac{\partial \overline{u}}{\partial t} + \frac{\partial}{\partial x}(\overline{u}\overline{v} + \overline{u'v'}) = \overline{Q_u}, \tag{2.15}$$

and by applying the operator parts of the equations, which splits the flow-field and the source-term (reaction part), we have:

$$\frac{\partial \overline{u}}{\partial t} = \overline{Q_u}, \tag{2.16}$$

and

$$\frac{\partial \overline{u}}{\partial t} + \frac{\partial}{\partial x}(\overline{u}\overline{v} + \overline{u'v'}) = 0, \tag{2.17}$$

*Example 2.2* A next example in ocean modelling, here we have also different scales (horizontal and vertical velocities).

We assume the following linearized model, see [7], while we choose the adjustment equation given in a linearized form:

$$\frac{\partial u}{\partial t} - fv = -P_x, \frac{\partial v}{\partial t} + fu = -P_y, \tag{2.18}$$

where $f$ is a function depending on time and space, $p = (P_x, P_y)^t$ is the pressure vector. We first decompose into the different physical processes (reaction and pressure part) and we have:

$$\frac{\partial u}{\partial t} = -P_x, \frac{\partial v}{\partial t} = -P_y, \tag{2.19}$$

and the second part:

$$\frac{\partial u}{\partial t} - fv = 0, \frac{\partial v}{\partial t} + fu = 0, \tag{2.20}$$

is further decomposed into:

$$u = \overline{u} + u', \tag{2.21}$$

$$v = \overline{v} + v', \tag{2.22}$$

and we get

$$\frac{\partial(\overline{u}+u')}{\partial t} - f(\overline{v}+v') = 0, \tag{2.23}$$

$$\frac{\partial(\overline{v}+v')}{\partial t} + f(\overline{u}+u') = 0, \tag{2.24}$$

and we apply the averaging and obtain:

$$\frac{\partial \overline{u}}{\partial t} - f\overline{v} = 0, \tag{2.25}$$

$$\frac{\partial \overline{v}}{\partial t} + f\overline{u} = 0. \tag{2.26}$$

Further, we can also solve the fluctuations or so-called inertia adjustments:

$$\frac{\partial u'}{\partial t} - fv' = 0, \tag{2.27}$$

$$\frac{\partial v'}{\partial t} + fu' = 0. \tag{2.28}$$

Here, we have decoupled the fast and slow velocities and also taken into account the different physical behaviours of the equation parts.

### *2.1.2 Decomposition of Convection-Diffusion-Reaction Problems*

The motivation of decomposing convection-diffusion-reaction (CDR) problems are important, while time-consuming standard numerical approaches, e.g. Runge–Kutta methods for the the whole equation parts, have their drawbacks in resolving the finest scales. More and more complexities of coupling all the equations parts need to apply novel methods, that can overcome the restriction to time- and spatial steps, see [8]. Nowadays CDR problems are used to simulate delicate transport and reaction processes in engineering applications, e.g. chemical reactors [9], combustion flames [10], and bioremediation [11, 12].

Because of the drawback of losing accuracy or dealing with numerical artefacts with large time-steps to classical discretization and splitting schemes, we propose the following splitting strategies for global multiphase convection-diffusion-reaction equation, see [13].

- Time Splitting: Decoupling of convection-reaction and diffusion equation to solve them separately

- Dimensional Splitting: Exact solving of the 1D time-dependent systems of the convection-reaction equations
- Functional Splitting: Laplace transformation of the 1D time-dependent systems of convection-reaction equations and solving analytically the resulting systems of ordinary differential equations
- Iterative Splitting: Fix-point schemes, which couple the sub-problems of the global problem, which are then solved in advance independently using an analytical approach.

The technique called functional splitting has been tried as a means of solving decomposable problems, see [2]. Functional splitting is implemented in a splitting approach, where the knowledge of the exact solutions of some sub-problems has an important role in obtaining a-priori test-functions for solving the systems of differential equations. The solutions can be used as test-functions to improve the discretization schemes, e.g. finite volume schemes, or to solve analytically sub-problems which are coupled in the splitting approach, see [3].

Here are the Assumptions 2.1 of Functional Splitting approaches.

**Assumption 2.1** In the following, we assume that our underlying problem has the following characteristics:

- Each sub-problem can be solved analytical or semi-analytical.
- The sub-problems can be coupled via splitting approaches, e.g. additive, multiplicative or iterative splitting methods.
- The underlying spatial discretization scheme, e.g. finite difference or finite volume method, can embed the one-dimensional analytical or semi-analytical solutions with a small splitting error, see Godunov's method [14, 15].
- Multiscale methods, e.g. multiscale expansion methods, can be applied and decompose to fine and coarse parts of the full model and apply multiscale splitting approaches, see [16].

In the following Fig. 2.2, we present the ideas of the this functional splitting approach to a coupled multiphase convection-diffusion-reaction (MCDR) equation. We start from the MCDR equation, while each parts, means the convection-, reaction-, diffusion- and multiphase- part (mobile and immobile parts) have their different spatial and time scales. In the step of the decomposition, we collect the different scales of equal or nearly equal part, so here in the Fig. 2.2, we can combine the convection and reaction part, immobile part. Now, we can concentrate on the four different model problems, e.g. convection-reaction equation, diffusion equation and mobile–immobile equations. In the next step, we apply the so-called Functional Splitting approach, see the Assumptions 2.1. Means, we can reconstruct one-dimensional solutions of each sub-problem, that has a highly accuracy, e.g., analytical or semi-analytical solution, and that the underlying spatial discretization scheme can embed such dimensional-splitted solutions. Further, we can concentrate on each simpler equation and apply multiscale approaches. In the final step, we couple the results of each sub-problem and apply the coupling approaches of the different splitting methods, see [17]. The errors of the applied methods, e.g. dimensional splitting error, time

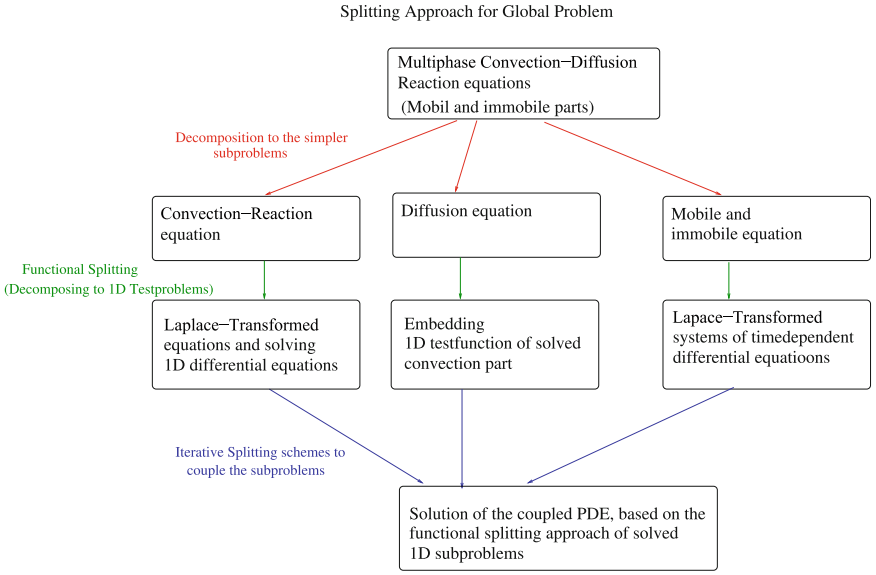



**Fig. 2.2** Splitting approach to convection-diffusion-reaction equations

splitting error, can be reduced by applying higher order schemes of each underlying method.

Such splitting approaches allow of accelerating the solver process, so one can employ larger time-steps. Taking into account the different scales of these multiscale problems, one solves each singlescale problem with its optimal accuracy, see [8].

Our contribution is to derive the framework of a splitting approach to solve time-dependent coupled transport and reaction equations with different splitting schemes producing analytically solvable one-dimensional equations, whose solutions are then used as test-function. This framework is more economical since it uses only standard approaches such as finite volume schemes.

*Remark 2.1* Furthermore, one could, hence, use more delicate chemical reaction terms and embed the semi-analytical solutions of their coupled convection-reaction systems into the schemes, or use iterative approaches to couple mixed mobile and immobile sub-models, which are delicate, to say the least, to solve only semi-analytical, see [17].

### *2.1.3 Functional Splitting with Respect to the Multiscale Approach*

Often it is necessary to deal with a multiscale model with different underlying models, e.g., microscopic and macroscopic model.

Numerically, we deal with a multiscale method, that solves each individual model and couple the datatransfer between the different models, see [18].

Then, we deal with a hierarchical Decomposition of the underlying different models, means in each hierarchy, e.g. microscopic part or macroscopic part, we deal with different decomposition methods.

In the following Fig. 2.3, we present some recipes to apply a hierarchical splitting approach. Here, we apply in the different model hierarchies the optimal splitting approaches. Such that we can minimize the underlying splitting error and reduce optimal the computational time.

In the following example, we deal with a multi-flow problem based on a macroscopic and microscopic convection-diffusion-reaction equation, see Example 2.3.

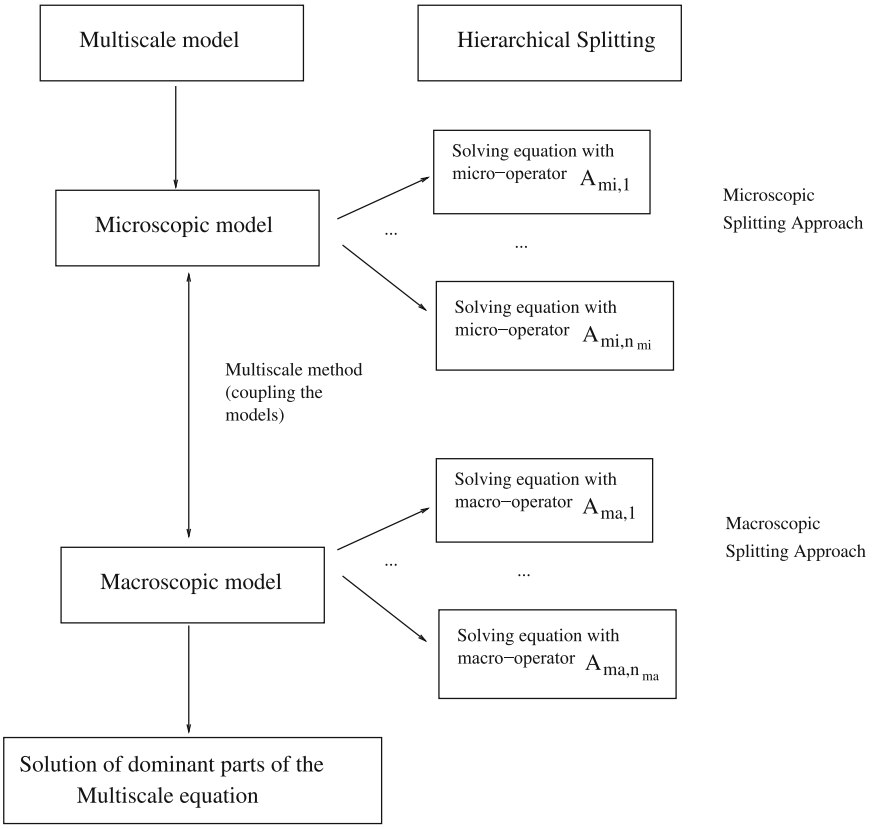



**Fig. 2.3** Hierarchical splitting approach (Coupling of micro–macro and macro–micro)

*Example 2.3* We have the following multi-flow problem, which is a coupled problem of fine- and coarse-scale CDR equations.

1. Macroscopic Equation:

$$\frac{du_{macro}}{dt} = F_1(u_{macro}, u_{micro}) + F_2(u_{macro}, u_{micro}). \tag{2.29}$$

   where $F_1$ is the convection-reaction operator and $F_2$ is diffusion operator.
2. Microscopic Equation:

$$\frac{du_{micro}}{dt} = -\frac{1}{\varepsilon}(\tilde{F}_1(u_{micro}) + \tilde{F}_2(u_{micro}) - \phi(u_{macro})). \tag{2.30}$$

   where $\tilde{F}_1$ is the convection-reaction operator and $\tilde{F}_2$ is diffusion operator. Further $u_{macro}$ is the slow time-dependent and $u_{micro}$ is the fast time-dependent variable.

In the following, we apply the HMM and the splitting of the different scale-dependent-equations in Algorithm 2.2.

**Algorithm 2.2** We first apply the HMM algorithm.

- We solve the microscopic equation:

$$u^{n,m+1}_{micro} = u^{n,m}_{micro} - \frac{\delta t}{\varepsilon}((\tilde{F}_1 u_{micro} + \tilde{F}_2 u^{n,m}_{micro}) - \phi(u^n_{macro})), \tag{2.31}$$

  where $m = 0, 1, \ldots, M-1$, z.B. $\delta t \leq \Delta t/M$ is applied as microscopic time-step.
- We apply the operator splitting method with respect to the microscopic equation:

$$u^{n,m+1}_{micro,1} = u^{n,m}_{micro,1} - \frac{\delta t}{\varepsilon}(\tilde{F}_1 u_{micro} - 0.5\phi(u^n_{macro})), \text{ with } u^{n,m}_{micro,1} = u^{n,m}_{micro}, \tag{2.32}$$

$$u^{n,m+1}_{micro,2} = u^{n,m+1}_{micro,1} - \frac{\delta t}{\varepsilon}(\tilde{F}_2 u^{n,m}_{micro} - 0.5\phi(u^n_{macro})), \text{ with } u^{n,m}_{micro,2} = u^{n,m+1}_{micro,1}, \tag{2.33}$$

  where $m = 0, 1, \ldots, M-1$, z.B. $\delta t \leq \Delta t/M$ is applied as microscopic time-step and next intermediate solution is given as $u^{n,m+1}_{micro} = u^{n,m+1}_{micro,2}$.
- Equilibration of the Microscopic operators (reconstruction):

$$\tilde{F}^n = \frac{1}{M}\sum_{m=1}^{M}(F_1(u^n_{macro}, u^{n,m}_{micro}) + F_2(u^n_{macro}, u^{n,m}_{micro})). \tag{2.34}$$

- Solving of the Macroscopic Equation:

$$u_{macro}^{n+1} = u_{macro}^{n} - \Delta t(\tilde{F}_1^n + \tilde{F}_2^n). \tag{2.35}$$

  with $\Delta t$ as macroscopic time-step.
- We apply the operator splitting method with respect to the macroscopic equation:

$$u_{macro,1}^{n+1} = u_{macro}^{n} - \Delta t \tilde{F}_1^n, \text{ with } u_{macro,1}^{n} = u_{macro}^{n}, \tag{2.36}$$

$$u_{macro,2}^{n+1} = u_{macro,1}^{n+1} - \Delta t \tilde{F}_2^n, \text{ with } u_{micro,2}^{n} = u_{macro,1}^{n+1}, \tag{2.37}$$

  where the next intermediate solution is given as $u_{macro}^{n+1} = u_{macro,2}^{n+1}$.
- We apply the next microscopic step, till we have resolved the full time interval.

*Remark 2.2* Here, we can apply the discrete macroscopic time-steps with respect to a fast splitting approach. Further also with the microscopic equation. The benefit is also to resolve only parts of the microscopic time interval such that we can also accelerate the multiscale computation.

## References


1. A. Araujo, J.A. Ferreira, P. de Oliveira, F. Patricio, P. Rosa, The use of splitting methods in the numerical simulation of reacting flows. Comput. Vis. Sci. **6**(2–3), 59–66 (2004)
2. L. Reifschneider, Solving inflow–outflow problems with functional splitting, in *Proceedings of the American Institute of Aeronautics and Astronautics, 22nd Meeting of the Aerospace Sciences*, Reno, NV, USA, 9–12, 1984
3. J. Geiser, in *Decomposition Methods for Partial Differential Equations: Theory and Applications in Multiphysics Problems*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules C.-H. Lai (CRC Press, Chapman & Hall/CRC, Boca Raton, 2009)
4. E.J. Davison, Ahigh order Crank-Nicholson technique for solving differential equations. Comput. J. **10**(2), 195–197 (1967)
5. P. Müller, *The Equations of Oceanic Motions* (Cambridge University Press, Cambridge, 2006)
6. S.B. Pope, *Turbulent Flows* (Cambridge University Press, Cambridge, 2000)
7. V.B. Zalesny, G.I. Marchuk, V.I. Agoshkov, A.V. Bagno, A.V. Gusev, N.A. Diansky, S.N. Moshonkin, R. Tamsalu, E.M. Volodin, Numerical simulation of large-scale ocean circulation based on the multicomponent splitting method. Russ. J. Numer. Anal. Math. Model. **25**(6), 581–609 (2010)
8. W. Hundsdorfer, J.G. Verwer, *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations* (Springer, Berlin, 2003)
9. H.A. Jakobsen, *Chemical Reactor Modeling: Multiphase Reactive Flows* (Springer, Berlin, 2008)
10. R.P. Fox, *Computational Models for Turbulent Reacting Flows* (Cambridge University Press, Cambridge, 2003)
11. J. Bear, Y. Bachmat, *Introduction to Modeling of Transport Phenomena in Porous Media* (Kluwer Academic Publishers, Dordrecht, 1991)
12. R.E. Ewing, Up-scaling of biological processes and multiphase flow in porous media.*IIMA Volumes in Mathematics and its Applications* (Springer, Berlin, 2002), pp. 195–215
13. J. Geiser, Discretization methods with analytical solutions for convection-diffusion-dispersion-reaction-equations and applications. J. Eng. Math. **57**(1), 79–98 (2007)

14. S. Godunov, Difference methods for the numerical calculations of discontinuous solutions of the equations of fluid dynamics. Mat. Sb. **47**, 271–306 (1959)
15. E.F. Toro, *Riemann Solvers and Numerical Methods for Fluid Dynamics* (Springer, Berlin, 1999)
16. J. Geiser, Recent advances in splitting methods for multiphysics and multiscale: theory and applications. J. Algorithms Comput. Technol., Multi-Sci., Brentwood, Essex, UK, acceptedAugust 2014 (to be published second issue 2015)
17. J. Geiser, in *Iterative Splitting Methods for Differential Equations*, Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.-H. Lai. (Chapman & Hall/CRC, Boca Raton 2011)
18. W.E.B. Engquist, X. Li, W. Ren, E. Vanden-Eijnden, Heterogeneous multiscale methods: a review. Commun. Comput. Phys. **2**(3), 367–450 (2007)

# Chapter 3
# Algorithmic Part

**Abstract** In this chapter, we discuss the algorithmic parts with respect to the different methods we applied in the application part.

## 3.1 Introduction

In the following, we discuss different methods based on iterative and additive ideas to decompose scale-dependent equations.

Based on the different scale-dependent operators of the equations, we deal with the ideas of decomposing into simpler and faster computable equations.

Basic idea is that to decompose the operator with respect to their spatial and time scales into different scale-dependent operators, e.g. we decompose the operator

$$A = A_{macro} + A_{micro}, \tag{3.1}$$

where the operators are given as

- $A_{macro}$ (macroscopic operator) has larger in order entries, and then
- $A_{micro}$ (microscopic operator) has smaller in order entries,

while $|A_{micro,ij}| \leq \varepsilon \ |A_{macro,ij}|, \ \forall i, j \in I, 0 < \varepsilon \ll 1$, i.e. we decompose the different scales of two operators.

To solve the evolution equation,

$$\frac{\partial c}{\partial t} = A_{macro}(c)c + A_{micro}(c)c, \tag{3.2}$$

where $c(0) = c_0$ is the initial condition and we assume that the semi-discretized operator $A$ has included the boundary conditions.

Two different solver ideas are discussed:

- Iterative Scheme: Based on iterative cycles, we solve the underlying decomposed equations based on the successive approximation or fixpoint scheme.
- Additive Scheme: Based on decomposing into tridiagonal matrices, we solve sequentially simpler equations.



J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5_3

## 3.2 Iterative Methods

This model is reformulated by the semi-discretization of the spatial operators to the following Cauchy problem, while $c$ is now in the following vectorial function:

$$\frac{\partial c(t)}{\partial t} = Ac(t) + f(t) \tag{3.3}$$

$$= A_1 c(t) + A_2 c(t) + f(t), \text{ with } t \in [0, T],\ c(0) = c_0, \tag{3.4}$$

where the initial function $c_0$ is given. $A_1$ and $A_2$ are assumed to be bounded, constant, linear operators in an appropriate Banach space $\mathbf{X}$ with $A_1, A_2 : \mathbf{X} \to \mathbf{X}$ with an appropriate vector and matrix norm $||\cdot||$.

In the following, we deal with the following definition of the stiff operators, see also [1, 2].

**Definition 3.1** We consider the stiffness in the following sense: $A_1$ is supposed to be stiff and $A_2$ non-stiff. Stiffness means that $\tau A_1$ is huge in norm for the range of step size $\tau$, see [3]. Here, the step size represents a splitting step size. So we assume

$$||\tau A_1|| \gg 1,\ \ ||\tau A_2|| = O(\tau). \tag{3.5}$$

For the notation of the eigenvalues, we have

$$Re(\tau\lambda) \ll -1,\ \ |\tau\mu| = O(\tau), \tag{3.6}$$

where $\lambda$ is a stiff eigenvalue of $A_1$ and $\mu$ a non-stiff eigenvalue of $A_2$.

In the next subsection, we present the iterative schemes.

### *3.2.1 Iterative Schemes*

We then consider the following forms of the iterative splitting schemes to solve the linear model equation:
1. Iterative splitting with respect to a diagonal matrix part (Jacobi Scheme):

$$\frac{\partial c_i(t)}{\partial t} = A_1 c_i(t) + A_2 c_{i-1}(t) + f(t), \text{ with } c_i(t^n) = c^n \tag{3.7}$$

$$\frac{\partial c_{i+1}(t)}{\partial t} = A_1 c_{i-2}(t) + A_2 c_{i+1}(t) + f(t), \text{ with } c_{i+1}(t^n) = c^n, \tag{3.8}$$

$$i = 1, 3, \ldots, 2m + 1,$$

$c_0(t) = 0$ and $c_{-1}(t) = 0$, after each iterative step we update $i = i + 1$.

2. Iterative splitting with respect to a full matrix part (Gauss–Seidel Scheme):

$$\frac{\partial c_i(t)}{\partial t} = A_1 c_i(t) + A_2 c_{i-1}(t) + f(t), \text{ with } c_i(t^n) = c^n \tag{3.9}$$

$$\frac{\partial c_{i+1}(t)}{\partial t} = A_1 c_i(t) + A_2 c_{i+1}(t) + f(t), \text{ with } c_{i+1}(t^n) = c^n, \tag{3.10}$$

$$i = 1, 3, \ldots, 2m+1,$$

3. Unsymmetrical weighted iterative splitting (JOR, Jacobian Overrelaxation Scheme):

$$\frac{\partial c_i(t)}{\partial t} = \frac{1}{\omega} A_1 c_i(t) + A_2 c_{i-1} + \left(1 - \frac{1}{\omega}\right) A_1 c_{i-2}(t) + f(t), \tag{3.11}$$

$$\text{with } c_i(t^n) = c^n$$

$$\frac{\partial c_{i+1}(t)}{\partial t} = A_1 c_{i-2}(t) + \frac{1}{\omega} A_2 c_{i+1}(t) + \left(1 - \frac{1}{\omega}\right) A_2 c_{i-1}(t) + f(t), \tag{3.12}$$

$$\text{with } c_{i+1}(t^n) = c^n,$$

$$i = 1, 3, \ldots, 2m+1,$$

where $\omega \in (0, 1]$.

4. Symmetrical weighted iterative splitting (SOR: Successive Overrelaxation Scheme):

$$\frac{\partial c_i(t)}{\partial t} = \frac{1}{\omega} A_1 c_i(t) + A_2 c_{i-1} + \left(1 - \frac{1}{\omega}\right) A_1 c_{i-2}(t) + f(t), \text{ with } c_i(t^n) = c^n \tag{3.13}$$

$$\frac{\partial c_{i+1}(t)}{\partial t} = A_1 c_i(t) + \frac{1}{\omega} A_2 c_{i+1}(t) + \left(1 - \frac{1}{\omega}\right) A_2 c_{i-1}(t) + f(t), \text{ with } c_{i+1}(t^n) = c^n,$$

$$i = 1, 3, \ldots, 2m+1, \tag{3.14}$$

where $\omega \in (0, 1]$.

*Remark 3.1* For all schemes, we assume that the operator $A_1$ has a large time scale and $A_2$ has a small time scale. In addition, the initialization is given as $c_0(t) = 0$, $c_{-1}(t) = 0$, while $c^n$ is the known split approximation at the time level $t = t^n$. The split approximation at the time level $t = t^{n+1}$ is defined as $c^{n+1} = c_{2m+1}(t^{n+1})$, with $n = 1, \ldots, N-1$ and the final time is given as $T = N\,\tau$.

### 3.2.2 Reformulation to Waveform Relaxation Scheme

In the following, we reformulate in the notation of the waveform relaxation scheme. We obtain the following schemes, see also [4, 5]:

$$\frac{dU_{\tilde{i}}}{dt} = \mathscr{P}U_{\tilde{i}} + \mathscr{Q}U_{\tilde{i}-1} + F, \tag{3.15}$$

$$U_{\tilde{i}}(t^n) = U(t^n), \tag{3.16}$$

$$\tilde{i} = 1, 2, \ldots, m, \tag{3.17}$$

where $U_{\tilde{i}-1} = (c_{i-2}, c_{i-1})^t$, $U_{\tilde{i}} = (c_i, c_{i+1})^t$ and the initialization $U_0(t) = (0, 0)^t$ is given with the zero vectors. Furthermore, we define that $\mathscr{P}$ and $\mathscr{Q}$ are the diagonal and outerdiagonal matrices of the underlying splitting methods given in Sect. 3.2.1 and $\mathscr{A} = \mathscr{P} + \mathscr{Q}$ is the full matrix.

We embed the iterative splitting schemes in the following waveform relaxation schemes:

(1) Jacobian:

$$\mathscr{P} = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix}, \mathscr{Q} = \begin{pmatrix} 0 & A_2 \\ A_1 & 0 \end{pmatrix}, F = \begin{pmatrix} f \\ f \end{pmatrix}. \tag{3.18}$$

(2) Gauss–Seidel:

$$\mathscr{P} = \begin{pmatrix} A_1 & 0 \\ A_1 & A_2 \end{pmatrix}, \mathscr{Q} = \begin{pmatrix} 0 & A_2 \\ 0 & 0 \end{pmatrix}, F = \begin{pmatrix} f \\ f \end{pmatrix}. \tag{3.19}$$

(3) JOR:

$$\mathscr{P} = \frac{1}{\omega}\begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix}, \mathscr{Q} = \left(1 - \frac{1}{\omega}\right)\begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} + \begin{pmatrix} 0 & A_2 \\ A_1 & 0 \end{pmatrix}, F = \begin{pmatrix} f \\ f \end{pmatrix}. \tag{3.20}$$

(4) SOR:

$$\begin{aligned} \mathscr{P} &= \frac{1}{\omega}\begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ A_1 & 0 \end{pmatrix}, \\ \mathscr{Q} &= \left(1 - \frac{1}{\omega}\right)\begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix} + \begin{pmatrix} 0 & A_2 \\ 0 & 0 \end{pmatrix}, F = \begin{pmatrix} f \\ f \end{pmatrix}. \end{aligned} \tag{3.21}$$

*Remark 3.2* We have also extended the application to non-autonomous differential equations, by adding the right-hand side term.

## 3.3 Additive Methods

The idea of the additive methods is to decompose in an additive manner the different operators of the differential equation. We concentrate on solving such semi-discretized linear evolution equations and the notation for such a differential equation is

$$B\partial_t\, u = Au,\ u(0) = u_0, \tag{3.22}$$

where $A$ and $B$ can be unbounded operators. We obtain large-scale differential equation, which are delicate to solve with standard solvers.

The evolution equation (3.22) is solved with the following underlying splitting schemes:

- Additive Splitting schemes and
- Iterative Splitting schemes.

### *3.3.1 Additive Splitting Schemes*

We deal with the following equation:

$$\sum_{\beta=1}^{p} B_{\alpha\beta}\partial_t\, u_\beta = \sum_{\beta=1}^{p} A_{\alpha\beta}u_\beta + f_\alpha,\ \alpha = 1, 2, \ldots, p, \tag{3.23}$$

$$u_\alpha(0) = u_{\alpha,0},\ \alpha = 1, 2, \ldots, p. \tag{3.24}$$

Furthermore, we assume that $A$ and $B$ are self-adjoint.

We apply the discretization with the schemes of weights and obtain

$$B\frac{u^{n+1} - u^n}{\tau} - A\left(\sigma u^{n+1} + (1-\sigma)u^n\right) = f\left(\sigma t^{n+1} + (1-\sigma)t^n\right), \tag{3.25}$$

By the transition to a new time level, we require

$$(B - A\sigma\tau)u^{n+1} = \phi^n, \tag{3.26}$$

while $\phi^n = (1-\sigma)\tau Au^n + Bu^n + f(\sigma t^{n+1} + (1-\sigma)t^n)$.

With the idea of splitting this into two problems, the original problem can be transformed to

$$\sum_{\beta=1}^{p}(B_{\alpha\beta} - A_{\alpha\beta}\sigma\tau)u_\beta^{n+1} = \phi_\alpha^n, \alpha = 1, 2, \ldots, p, \tag{3.27}$$

where $(\tilde{B} - A\sigma\tau) = (B - A_1\sigma\tau)B^{-1}(B - A_2\sigma\tau)$ and $\sigma \in (0, 1)$.

We have to solve the following pair of linear equations:

$$(B - A_1\sigma\tau)\psi^n = \phi^n, \tag{3.28}$$

$$(B - A_2\sigma\tau)u^{n+1} = \psi^n. \tag{3.29}$$

By a change to a sequence of simpler problems, we have

$$\left(B_{\alpha\alpha} - \frac{1}{2}A_{\alpha\alpha}\sigma\tau\right)u_\beta^{n+1/2} = \tilde{\psi}_\alpha^n, \alpha = 1, 2, \ldots, p, \tag{3.30}$$

$$\left(B_{\alpha\alpha} - \frac{1}{2}A_{\alpha\alpha}\sigma\tau\right)u_\beta^{n+1} = \hat{\psi}_\alpha^n, \alpha = 1, 2, \ldots, p. \tag{3.31}$$

Here, we have the benefit of needing to invert only the diagonal parts of the matrices and use the idea of solving the triangular splitting of the operator $A = A_1 + A_2$.

The second-order algorithm is given as a two-step method, see Algorithm 3.1.

**Algorithm 3.1** (1) Compute
$\tilde{\psi}^{n+1} = (\tilde{\psi}_1^{n+1}, \ldots, \tilde{\psi}_p^{n+1})^T$ with $\phi^n = (\phi_1^{n+1}, \ldots, \phi_p^{n+1})^T$

$$\tilde{\psi}_1^{n+1} = \left(I - \frac{1}{2}A_{11}B_{11}^{-1}\sigma\tau\right)^{-1}\phi_1^n \tag{3.32}$$

$$\tilde{\psi}_2^{n+1} = \left(I - \frac{1}{2}A_{11}B_{11}^{-1}\sigma\tau\right)^{-1}\left(\phi_2^n + A_{21}\sigma\tau B_{11}^{-1}\tilde{\psi}_1^{n+1}\right) \tag{3.33}$$

$$\ldots \tag{3.34}$$

$$\tilde{\psi}_p^{n+1} = \left(I - \frac{1}{2}A_{pp}B_{pp}^{-1}\sigma\tau\right)^{-1}\left(\phi_p^n + \sum_{i=1}^{p-1}A_{pi}\sigma\tau B_{ii}^{-1}\tilde{\psi}_i^{n+1}\right), \tag{3.35}$$

while $\phi^n = (1 - \sigma)\tau Au^n + Bu^n + f(\sigma t^{n+1} + (1 - \sigma)t^n)$.

(2) Compute $u^{n+1} = (u_1^{n+1}, \ldots, u_p^{n+1})^T$ with $\tilde{\psi}^{n+1} = (\tilde{\psi}_1^{n+1}, \ldots, \tilde{\psi}_p^{n+1})^T$

$$u_p^{n+1} = \left(B_{pp} - \frac{1}{2}A_{pp}\sigma\tau\right)^{-1}\tilde{\psi}_p^n \tag{3.36}$$

$$u_{p-1}^{n+1} = \left(B_{p-1p-1} - \frac{1}{2}A_{p-1p-1}\sigma\tau\right)^{-1}\left(\tilde{\psi}_{p-1}^n + A_{p-1p}\sigma\tau u_p^{n+1}\right) \tag{3.37}$$

$$\ldots \tag{3.38}$$

$$u_1^{n+1} = \left(B_{11} - \frac{1}{2}A_{11}\sigma\tau\right)^{-1}\left(\tilde{\psi}_1^n + \sum_{i=2}^{p}A_{1i}\sigma\tau u_i^{n+1}\right). \tag{3.39}$$

**Theorem 3.2** *If we choose $\sigma \geq \frac{1}{2}$, then the splitting scheme (3.30) and (3.31) is absolutely stable in an appropriate Hilbert space.*

*Proof* The outline of the proof is given in [6].

*Example 3.1* We have $2n \times 2n$ matrices.

The algorithm is as follows:

(1) Compute $\tilde{\psi}^{n+1} = (\tilde{\psi}_1^{n+1}, \tilde{\psi}_2^{n+1})^T$ with $\phi^n = (\phi_1^n, \phi_2^n)^T$

$$\tilde{\psi}_1^{n+1} = \left(I - \frac{1}{2}A_{11}B_{11}^{-1}\sigma\tau\right)^{-1}\phi_1^n \tag{3.40}$$

$$\tilde{\psi}_2^{n+1} = \left(I - \frac{1}{2}A_{22}B_{22}^{-1}\sigma\tau\right)^{-1}\left(\phi_2^n + A_{21}\sigma\tau B_{11}^{-1}\tilde{\psi}_1^{n+1}\right), \tag{3.41}$$

while $\phi^n = (1-\sigma)\tau Au^n + Bu^n + f(\sigma t^{n+1} + (1-\sigma)t^n)$.

(2) Compute $u^{n+1} = (u_1^{n+1}, u_2^{n+1})^T$ with $\tilde{\psi}^{n+1} = (\tilde{\psi}_1^{n+1}, \tilde{\psi}_2^{n+1})^T$

$$u_2^{n+1} = \left(B_{22} - \frac{1}{2}A_{22}\sigma\tau\right)^{-1}\tilde{\psi}_2^n \tag{3.42}$$

$$u_1^{n+1} = \left(B_{11} - \frac{1}{2}A_{11}\sigma\tau\right)^{-1}\left(\tilde{\psi}_1^n + A_{12}\sigma\tau u_2^{n+1}\right), \tag{3.43}$$

### *3.3.2 Higher Order Additive Splitting Method*

The drawback of the standard additive splitting method is the restriction to a second-order scheme.

To overcome this limitation, an extension can be made in the direction of the higher order Crank–Nicolson scheme, see [7].

The higher order Crank–Nicolson method can be derived as follows (see also [7]):

$$\begin{aligned} u(t^{n+1}) &= u(t^n) + h\frac{du}{dt}(t^n) \\ &\quad + \frac{h^2}{2!}\frac{d^2u}{dt^2}(t^n) + \frac{h^3}{3!}\frac{d^3u}{dt^3}(t^n) + \frac{h^4}{4!}\frac{d^4u}{dt^4}(t^n)\ldots, \end{aligned} \tag{3.44}$$

$$\begin{aligned} u(t^n) &= u(t^{n+1}) - h\frac{du}{dt}(t^{n+1}) \\ &\quad + \frac{h^2}{2!}\frac{d^2u}{dt^2}(t^{n+1}) - \frac{h^3}{3!}\frac{d^3u}{dt^3}(t^{n+1}) + \frac{h^4}{4!}\frac{d^4u}{dt^4}(t^{n+1})\ldots, \end{aligned} \tag{3.45}$$

subtracting the two equations and applying it to Eq. (5.534) in the form $\partial_t u = B^{-1}Au = \tilde{A}u$,

$$\begin{aligned} u(t^{n+1}) - u(t^n) &= \frac{h}{2}\left(\tilde{A}u(t^{n+1}) + \tilde{A}u(t^n)\right) \\ &+ \frac{h^2}{2\,2!}\left(-\tilde{A}^2u(t^{n+1}) + \tilde{A}^2u(t^n)\right) \\ &+ \frac{h^3}{2\,3!}\left(\tilde{A}^3u(t^{n+1}) + \tilde{A}^3u(t^n)\right)\ldots, \end{aligned} \tag{3.46}$$

we obtain

$$\left(I - \frac{h}{2}\tilde{A} + \frac{h^2}{2\,2!}\tilde{A}^2\right)u(t^{n+1}) = \left(I + \frac{h}{2}\tilde{A} + \frac{h^2}{2\,2!}\tilde{A}^2\right)u(t^n), \tag{3.47}$$

which is a third-order scheme.

The same can be obtained by the fractional step scheme

$$\begin{aligned} &\left(I - \sigma h\tilde{A} + \frac{\sigma\, h^2}{2!}\tilde{A}^2\right)u(t^{n+1}) \\ &= \left(I + (1-\sigma)\, h\tilde{A} + \frac{(1-\sigma)\, h^2}{2!}\tilde{A}^2\right)u(t^n), \end{aligned} \tag{3.48}$$

which is a third-order scheme for $\sigma = \frac{1}{2}$.

There is a decomposition idea based on a splitting into tridiagonal matrices.

The higher order additive splitting algorithm is given in the following scheme:

$$\begin{aligned} &\left(I - \sigma h\tilde{A} + \sigma\frac{h^2}{2!}\tilde{A}^2\right) \\ &= \left(I - \sigma h\tilde{A}_1 + \sigma\frac{h^2}{2!}\tilde{A}_1^2\right) \\ &\cdot\left(I - \sigma h\tilde{A}_2 + \sigma\frac{h^2}{2!}\tilde{A}_2^2\right) + \sigma\frac{h^2}{2!}[\tilde{A}_2, \tilde{A}_1] + O(h^3), \end{aligned} \tag{3.49}$$

where $\tilde{A} = B^{-1}A$ and $A = A_1 + A_2$ where $A_1 = A_2^t$.
The commutator is $[A_2, A_1] = A_2A_1 - A_1A_2$.

By the transition to a new time level, we require

$$\left(I - \sigma h\tilde{A}_1 + \sigma\frac{h^2}{2!}\tilde{A}_1^2\right)\left(I - \sigma h\tilde{A}_2 + \sigma\frac{h^2}{2!}\tilde{A}_2^2\right)u(t^{n+1}) = \phi^n, \tag{3.50}$$

where $\phi^n = \left(I + (1-\sigma)\, h\tilde{A} + (1-\sigma)\frac{h^2}{2!}\tilde{A}^2 - \sigma\frac{h^2}{2!}[\tilde{A}_2, \tilde{A}_1]\right)u(t^n)$.

We have to solve the following pair of linear equations:

$$\left(I - \sigma h \tilde{A}_1 + \sigma \frac{h^2}{2!} \tilde{A}_1^2\right) \psi^{n+1} = \phi^n, \tag{3.51}$$

$$\left(I - \sigma h \tilde{A}_2 + \sigma \frac{h^2}{2!} \tilde{A}_2^2\right) u^{n+1} = \psi^{n+1}, \tag{3.52}$$

where the $\psi^{n+1}$ are the intermediate solutions of the scheme.

The third-order algorithm is given as a three-step method, presented in the following Algorithm 3.3.

**Algorithm 3.3** (1) Compute $\psi^{n+1} = (\psi_1^{n+1}, \ldots, \psi_p^{n+1})^T$ with $\phi^n = (\phi_1^{n+1}, \ldots, \phi_p^{n+1})^T$

$$\psi_1^{n+1} = \left(I - \frac{1}{2} A_{11} \sigma h + \left(\frac{1}{2} A_{11}\right)^2 \sigma \frac{h^2}{2!}\right)^{-1} \phi_1^n \tag{3.53}$$

$$\psi_2^{n+1} = \left(I - \frac{1}{2} A_{22} \sigma h + \left(\frac{1}{2} A_{22}\right)^2 \sigma \frac{h^2}{2!}\right)^{-1} \left(\phi_2^n + \left(A_{21} \sigma h - \{A_1 A_1\}_{21}\, \sigma \frac{h^2}{2!}\right) \psi_1^{n+1}\right) \tag{3.54}$$

$$\ldots \tag{3.55}$$

$$\psi_p^{n+1} = \left(I - \frac{1}{2} A_{pp} \sigma h + \left(\frac{1}{2} A_{pp}\right)^2 \sigma \frac{h^2}{2!}\right)^{-1} \left(\phi_p^n + \sum_{i=1}^{p-1} \left(A_{pi} \sigma h - \{A_1 A_1\}_{pi}\, \sigma \frac{h^2}{2!}\right) \psi_i^{n+1}\right), \tag{3.56}$$

while $\phi^n = \left(I + (1-\sigma)\, hA + (1-\sigma) \frac{h^2}{2!} A^2 - \sigma \frac{h^2}{2!} [A_2, A_1]\right) u(t^n)$ and the matrix multiplication $\{A_1 A_1\}_{ij} = \sum_{k=1}^{p} A_{1,ik} A_{1,kj}$, where $p$ is the rank of the matrix $A_1$ and $A_{1,ij}$ is the $i$, $j$th element of the matrix $A_1$.

(2) Compute $u^{n+1} = (u_1^{n+1}, \ldots, u_p^{n+1})^T$ with $\psi^{n+1} = (\psi_1^{n+1}, \ldots, \psi_p^{n+1})^T$

$$u_p^{n+1} = \left(I - \frac{1}{2} A_{pp} \sigma h + \left(\frac{1}{2} A_{pp}\right)^2 \sigma \frac{h^2}{2!}\right)^{-1} \psi_p^n \tag{3.57}$$

$$u_{p-1}^{n+1} = \left(I - \frac{1}{2} A_{p-1p-1} \sigma h + \left(\frac{1}{2} A_{p-1p-1}\right)^2 \sigma \frac{h^2}{2!}\right)^{-1} \cdot \left(\psi_{p-1}^n + \left(A_{p-1p} \sigma h - \{A_2 A_2\}_{p-1p}\, \sigma \frac{h^2}{2!}\right) u_p^{n+1}\right) \tag{3.58}$$

$$\ldots \tag{3.59}$$

$$u_1^{n+1} = \left(I - \frac{1}{2} A_{11} \sigma \frac{h}{2} + \left(\frac{1}{2} A_{11}\right)^2 \sigma \frac{h^2}{2!}\right)^{-1} \cdot \left(\psi_1^n + \sum_{i=2}^{p} \left(A_{1i} \sigma h - \{A_2 A_2\}_{1i}\, \sigma \frac{h^2}{2!}\right) u_i^{n+1}\right), \tag{3.60}$$

and the matrix multiplication $\{A_2 A_2\}_{ij} = \sum_{k=1}^{p} A_{2,ik} A_{2,kj}$, where $p$ is the rank of the matrix $A_2$ and $A_{2,ij}$ is the $i, j$th element of the matrix $A_2$.

**Theorem 3.4** *If we choose $\sigma \geq \frac{1}{2}$, then the splitting scheme (3.51) and (3.52) is absolutely stable in an appropriate Hilbert space.*

*Proof* The outline of the proof is given in [6].

### *3.3.3 Iterative Splitting Method*

The following algorithm is based on an iteration with a fixed splitting discretization step size $\tau$, namely, on the time interval $[t^n, t^{n+1}]$, we solve the following subproblems consecutively for $i = 0, 2, \ldots 2m$ (cf. [8, 9]):

$$\frac{\partial c_i(t)}{\partial t} = A_1 c_i(t) + A_2 c_{i-1}(t), \text{ with } c_i(t^n) = c^n \tag{3.61}$$

$$\text{and } c_0(t^n) = c^n, \quad c_{-1} = 0.0,$$

$$\frac{\partial c_{i+1}(t)}{\partial t} = A_1 c_i(t) + A_2 c_{i+1}(t), \tag{3.62}$$

$$\text{with } c_{i+1}(t^n) = c^n,$$

where $c^n$ is the known split approximation at the time level $t = t^n$. The split approximation at the time level $t = t^{n+1}$ is defined as $c^{n+1} = c_{2m+1}(t^{n+1})$. (Clearly, the function $c_{i+1}(t)$ depends on the interval $[t^n, t^{n+1}]$, too, but, for the sake of simplicity, in our notation, we omit the dependence on $n$.)

In the following, we will analyse the convergence and the rate of convergence of the method (3.61) and (3.62) as $m$ tends to infinity for the linear operators $A_1, A_2 : \mathbf{X} \to \mathbf{X}$, where we assume that these operators and their sum are generators of $C_0$ semi-groups. We emphasize that these operators are not necessarily bounded, so the convergence is examined in a general Banach space setting.

The novelty of the convergence results are the reformulation in integral notation. Based on this, we can assume that we have bounded integral operators which can be estimated and given in a recursive form. Such formulations are known in the work of [10, 11], and estimations of the kernel part with the exponential operators are sufficient to estimate the recursive formulations.

## 3.4 Parallelization

The parallelization is important to accelerate the solver methods.

We distinguish between three different parallelization areas:

- Parallelization in Time, e.g. Parareal method: Decomposition of large time intervals to smaller time intervals
- Parallelization in Operators, e.g. Parallel operator splitting method
- Parallelization in Space, e.g. Schwartz waveform relaxation, Domain decomposition algorithms

The application of the different parallel methods are discussed in the following:

- Time parallelization: The large time interval is decomposed into smaller time intervals (time decomposition). The full equations can be handled in one processor, such that the memory effect is not too important. But the duration of the full time interval is very large such that it will take too long for one processor. Therefore, we decompose it to smaller time intervals and parallelize the large time interval, i.e. each time slot can be handled independently by one processor, see [12].
- Operator splitting methods or parallelization of the different operators: The problem is based on storing the full operator of the differential equation in one processor (this was the motivation of the earliest splitting schemes [13]). Therefore, we decompose the full operator into simpler operators and distribute the simpler operators, which can be stored into one processor, to various processors. The operators are coupled via the operator splitting scheme and can be computed in parallel. Such ideas allow to deal with modular coupling of program codes, e.g. different specialized codes for an E- and B-field (e.g. Maxwell equation) and a transport field (e.g. particle code), which can be computed on different PC clusters.

*Example 3.2* Reduction of the computational time via time parallelization.

We assume to have an effective parallel algorithm with about 20–50 %, see [12].

Therefore, we reduce the computational time for one processor, for example, of 48 [h], with 128 processors and an efficiency of about 20 % to 2–3 [h].

### *3.4.1 Time Parallelization: Parareal Algorithm as an Iterative Solver*

The original algorithm was introduced by [14]. The idea is to partition the time domain $\Omega_N = [0, T]$, which is large, into $N$ time subdomains:

$$\Omega_n \quad [T_{n-1}, T_n], n = 1, \ldots, N, \tag{3.63}$$

furthermore, we define the following solvers:

- coarse solver (coarse propagator): $G(T_n, T_{n-1}, x)$ and
- fine solver (fine propagator): $F(T_n, T_{n-1}, x)$

with both, we can approximate the underlying differential equation:

$$U'(t) = f(t, U(t)), U(T_{n-1}) = x. \tag{3.64}$$

Here, we assume the following:

1. The coarse integrator is computationally much faster, i.e. a lower order scheme, than the fine integrator.
2. The fine integrator is much more accurate, i.e. a higher order scheme, and much more time consuming, and therefore we need the benefit of parallelization.

We have the following steps:

- In the first iteration, we use the coarse integrator in a serial fashion to provide initial conditions to each time slice $\Omega_n$:

$$U_n^1 = G(T_n, T_{n-1}, U_{n-1}^1), \quad n = 1, 2, \ldots, N.$$

- In the second step, we use the fine propagator and integrate independently (i.e. in parallel) $N$ initial value problems $F(T_n, T_{n-1}, U_{n-1}^k)$ $(n = 1, 2, \ldots, N)$, yielding new approximations for the initial conditions on the following time slices.
- In each iteration $k$, the corrections are then again quickly propagated using the coarse integrator:

$$U_n^{k+1} = F(T_n, T_{n-1}, U_{n-1}^k) + G(T_n, T_{n-1}, U_{n-1}^{k+1}) - G(T_n, T_{n-1}, U_{n-1}^k), \tag{3.65}$$

*Example 3.3* We deal with a differential equation,

$$U' = AU + BU, U(0) = u(0), \tag{3.66}$$

with two operators $A$ and $B$.

We assume to have $F$ as a fine integrator and choose the iterative splitting method as a more accurate propagator. Further, we assume to have $G$ as a coarse integrator and choose the A–B splitting scheme as a lower order accurate propagator, see the example in [15].

The method can be compared be the so-called *Multiple Shooting Method*, see [16]. While we repeat each time slot with an improved approximation and if the error is small enough, we go on to the next time intervals, see also Fig. 3.1.

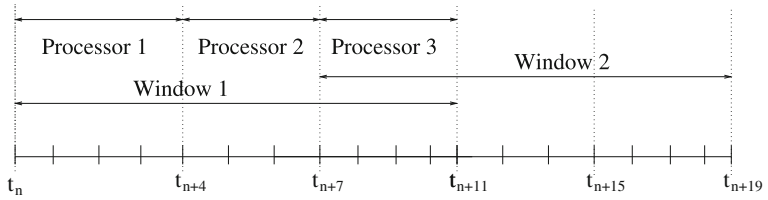



**Fig. 3.1** Parallelization with Parareal, windowing of the parallel process

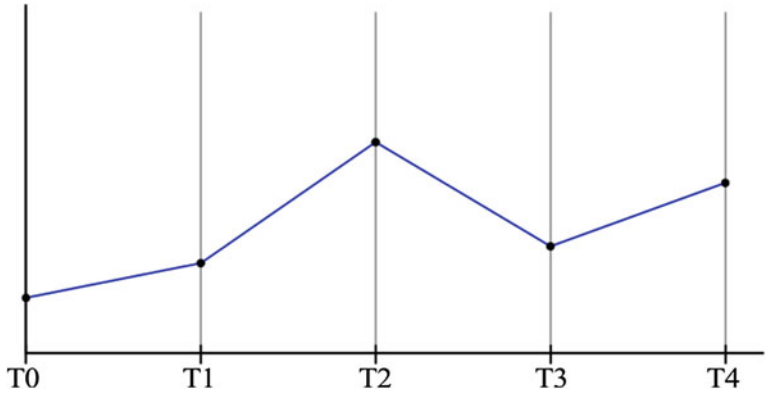



**Fig. 3.2** First initialization step in the algorithm

In the following, we discuss the different steps.

**Step 1:**

Coarse computation of one processor of the full interval with a fast-and low-order solver method, e.g. forward Euler scheme, see Fig. 3.2.

We propagate in the coarse method with

$$U_0^1 = u_0, \tag{3.67}$$

$$U_n^1 = G(T_n, T_{n-1}, U_{n-1}^1), n = 1, \ldots, N. \tag{3.68}$$

**Step 2:**

The next step is a fine propagator with $n$-processors, for each smaller time interval. The methods for the smaller time intervals are of higher order and expensive in time, see Fig. 3.3.

We propagate with the fine propagator, while the initial conditions are given for each subdomain $\Omega_n$:

$$U_{fine,n}^1 = F\left(T_n, T_{n-1}, U_{n-1}^1\right), n = 1, \ldots, N. \tag{3.69}$$

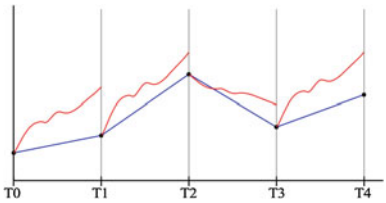



**Fig. 3.3** Second step of a fine propagator step done in parallel

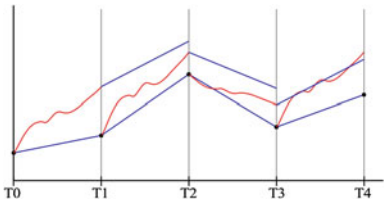



**Fig. 3.4** The third step is the corrector step, which coupled the coarse and fine step and go on with the initialization of the next timeframe

**Step 3:**

The next step is the corrector step to couple the coarse and fine steps together (coupling process). One processor computes the corrections between each time interval. Such time intervals which fit of the accuracy are finished and we step forward. The other intervals are computed via the first step and so on, see Fig. 3.4.

We apply the improved initial guess and propagate coarsely in a correction:

$$U_n^{k+1} = F\left(T_n, T_{n-1}, U_{n-1}^k\right) + G\left(T_n, T_{n-1}, U_{n-1}^{k+1}\right) - G\left(T_n, T_{n-1}, U_{n-1}^k\right). \quad (3.70)$$

If the error in the time slot is sufficient small, we shift the window to the next time slot and start with the step 1, till we are done. Otherwise, we go on with the next iterative step $k = k + 1$.

### *3.4.2 Operator Parallelization: Operator Splitting Method*

We deal with large operators in a differential equation, which is given as

$$\frac{dc(t)}{dt} = A_{\text{full}} c(t), \quad \text{for } t \in (t^n, T), \quad (3.71)$$

$$\frac{dc(t)}{dt} = \sum_{i=1}^{m} A_i \; c(t), \quad \text{for } t \in (t^n, T), \quad (3.72)$$

$$c(0) = c_0, \; \text{Initial-conditions}, \quad (3.73)$$

where we have the time intervals $t_1, t_2, \ldots, t_N$.

We assume that the full operator $A_{full}$ is partitioned into different smaller operators $A_j$, $j = 1, \ldots, m$. Furthermore, we have an appropriate Banach space with a vector and induced matrix norm $||\cdot||$, where $c \in \mathbf{X}$ and also the operators are given in $A_j \in \mathbf{X}^2$ for $j = 1, \ldots, m$.

We also assume that the operators include the boundary conditions and are derived of semi-discretizations, e.g. Finite Difference or Finite Element Methods. Based on their problems, they might have different physical behaviours, e.g. diffusion, convection or reaction operators, if we deal with a fluid flow problem, see [17].

We deal with the following problems:

- The full operator $A_{full}$ cannot be stored into one processor, and therefore we have to partition the problem to $A_j$, $j = 1, \ldots, m$ smaller operators.
- The physical problem allows to deal with different program codes, e.g. we have a code for the diffusion problem, a code for the reaction problem and so on. We only like to couple such problems via the splitting approach.

### *3.4.3 Sequential Operator Splitting Method*

Such a scheme can be applied to couple the different operators to the full operator equations (3.71). We deal with a successive computation of each operator in each time slot and couple via the initial conditions of each step, see [18].

We solve $m$ subproblem sequentially on the subintervals $[t^n, t^{n+1}]$, where $n = 0, 1, \ldots, N-1$, $t^0 = 0$ and $t^N = T$.

The subproblems are given in the following and coupled via the initial conditions:

$$\frac{\partial c_1(t)}{\partial t} = A_1 c_1(t), \quad \text{with } c_1(t^n) = c(t^n), \tag{3.74}$$

$$\frac{\partial c_2(t)}{\partial t} = A_2 c_2(t), \quad \text{with } c_2(t^n) = c_1(t^{n+1}), \tag{3.75}$$

$$\vdots \tag{3.76}$$

$$\frac{\partial c_m(t)}{\partial t} = A_m c_m(t), \quad \text{with } c_m(t^n) = c_{m-1}(t^{n+1}), \tag{3.77}$$

for $n = 0, 1, \ldots, N-1$ and $\tau = t^{n+1} - t^n$, where $c(t^{n+1}) = c_m(t^{m+1})$ is the approximated solution at the time point $t^{n+1}$.

The local splitting error of the sequential scheme is $\mathcal{O}(\tau^2)$ and the global splitting error of the sequential scheme is $\mathcal{O}(\tau)$, if we assume non-commutable operators. Otherwise, we are exact.

Here, we have to wait for the next initial condition, while we are dependent on the result of the previous step, such that the scheme is only interested to couple different codes, see [18].

### 3.4.4 Parallel Operator Splitting Method: Version 1

The following first parallel operator splitting method is also called splitting-up method, see [19, 20].

We also deal with $m$ sub-problems, which can be solved independently, i.e. parallel, while the initial conditions are given at time point $t^n$ for each sub-problem and independent of other sub-problems, see [21].

We have the subintervals $[t^n, t^{n+1}]$, where $n = 0, 1, \ldots, N-1$, $t^0 = 0$ and $t^N = T$. We deal with $m$ parallel sub-problem given as follows:

$$\frac{\partial c_1(t)}{\partial t} = A_1 c_1(t), \quad \text{with } c_1(t^n) = c(t^n), \tag{3.78}$$

$$\frac{\partial c_2(t)}{\partial t} = A_2 c_2(t), \quad \text{with } c_2(t^n) = c(t^n), \tag{3.79}$$

$$\vdots \tag{3.80}$$

$$\frac{\partial c_m(t)}{\partial t} = A_m c_m(t), \quad \text{with } c_m(t^n) = c(t^n), \tag{3.81}$$

and result in one additive step that couples the independent sub-steps:

$$c(t^{n+1}) = c(t^n) + \sum_{i=1}^{m} \Big( c_i(t^{n+1}) - c(t^n) \Big),$$
$$n = 1, 2, \ldots, N, \quad \text{where } c(0) = c_0.$$

The local splitting error of the parallel scheme is $\mathcal{O}(\tau)$ if we deal with non-commutable operators. Otherwise, we are exact.

Based on the low-order scheme, we introduce in the following a second-order scheme, which can also be applied in parallel, see [21].

### 3.4.5 Parallel Operator-Splitting Method: Version 2

The following second parallel operator-splitting method is also weighted sequential splitting method, see [21].

We obtain a second-order scheme, like the Strang splitting scheme, see [13], while we apply sequential splitting in both directions, i.e., $A_1 \to A_2 \to \cdots, \to A_m$ and $A_m \to A_{m-1} \to \cdots, \to A_1$.

We also deal with two sequential splitting problems, which can be handled parallel, while each splitting problem has $m$ sub-problems. These sub-problems are dependent and are done sequentially, see [21].

We have the two independent $m$ sequential problems in the subintervals $[t^n, t^{n+1}]$, where $n = 0, 1, \ldots, N-1$, $t^0 = 0$ and $t^N = T$.

We deal with the first $m$ sequential sub-problem given as

$$\frac{\partial c_1(t)}{\partial t} = A_1 c_1(t), \quad \text{with } c_1(t^n) = c(t^n), \tag{3.82}$$

$$\frac{\partial c_2(t)}{\partial t} = A_2 c_2(t), \quad \text{with } c_2(t^n) = c_1(t^{n+1}), \tag{3.83}$$

$$\vdots \tag{3.84}$$

$$\frac{\partial c_m(t)}{\partial t} = A_m c_m(t), \quad \text{with } c_m(t^n) = c_{m-1}(t^{n+1}), \tag{3.85}$$

and the second $m$ sequential sub-problem given as

$$\frac{\partial v_1(t)}{\partial t} = A_m v_1(t), \quad \text{with } v_1(t^n) = c(t^n), \tag{3.86}$$

$$\frac{\partial v_2(t)}{\partial t} = A_{m-1} v_2(t), \quad \text{with } v_2(t^n) = v_1(t^{n+1}), \tag{3.87}$$

$$\vdots \tag{3.88}$$

$$\frac{\partial v_m(t)}{\partial t} = A_1 v_m(t), \quad \text{with } v_m(t^n) = v_{m-1}(t^{n+1}). \tag{3.89}$$

We result in one additive step that couples the independent sub-problems:

$$c(t^{n+1}) = \frac{1}{2} c_m(t^{n+1}) + \frac{1}{2} v_m(t^{n+1}).$$

The local splitting error of the parallel scheme is $\mathcal{O}(\tau^2)$, and the global splitting error of the parallel scheme is $\mathcal{O}(\tau)$, if we deal with non-commutable operators. Otherwise, we are exact.

In the next subsection, we present an iterative splitting scheme, which deals with a parallelization of the exp-operators.

### *3.4.6 Iterative Splitting Scheme*

The iterative splitting scheme is based on a relaxation idea, see [18].

Here, we deal with exp-operators, which can also be applied independently. Based on the idea to relax only to the so-called dominant operators, see [18], we only apply multiplications via the non-dominant operators, see [22].

We assume that we are partitioned into two operators and deal with the following algorithm. The time interval is given as $[t^n, t^{n+1}]$ and we solve the following sub-problems with the iterative steps $i = 1, 2, \ldots, I$:

$$\frac{dc_i(t)}{dt} = Ac_i(t) \; + \; Bc_{i-1}(t), \; \text{with} \; c_i(t^n) = c_{\text{sp}}^n, \tag{3.90}$$

where $c_0(t)$ is an initialization for the iterative scheme, e.g. $c_0(t) = 0$.

The iterative schemes are solved in the following manner:

$$c_1(t) = \exp(At)c(t^n), \tag{3.91}$$

$$c_2(t) = c_1(t) + c_1(t) \int_0^t [B, \exp(sA)]ds, \tag{3.92}$$

where $[\cdot, \cdot]$ is the commutator.

Based on the $\exp(At)$ operators, we can decouple into $A$ and $B$ dependent terms, see [22].

*Remark 3.3* The iterative splitting schemes have the benefit of their modularization, i.e. we could add relaxed operators to the scheme. A drawback is the strong coupling in each iterative step, which means that the parallelization is more delicate, and compare also the waveform relaxation methods with Jacobian or Gauss–Seidel Schemes, see [4].

### *3.4.7 Spatial Parallelization Techniques*

#### Domain Decomposition

Traditional domain decomposition schemes, e.g. Schwarz waveform relaxation schemes, motivate with a different idea to decompose the domains into subdomains, such that the operator is only defined in subdomains.

*Example 3.4* We start to decompose the operator $A$ into operators defined at each subdomain. We have $\overline{\Omega} = \overline{\Omega_1} \cup \overline{\Omega_2}$; here, we obtain an *artificial* boundary with $\Omega_1 \cap \Omega_2$, which is not considered in an iterative operator splitting scheme.

The main advantage of such decomposition is the two decoupled independent equations on each domain:

$$A|_{\Omega_1} u|_{\Omega_1} = f|_{\Omega_1} \tag{3.93}$$

$$A|_{\Omega_2} u|_{\Omega_2} = f|_{\Omega_2} \tag{3.94}$$

where we assume that the boundary condition at the boundary $\partial\Omega \cap \Omega_1$ is included in $A_1$ and the boundary condition $\partial\Omega \cap \Omega_2$ is included in $A_2$.

To couple the two separate equations, we have to apply waveform relaxation methods with the *artificial* boundary condition, which can be given as

$$A|_{\Omega_1} u_i|_{\Omega_1} = f|_{\Omega_1} \tag{3.95}$$

$$B|_{\Omega_1 \cap \Omega_2} u_i|_{\Omega_1 \cap \Omega_2} = B|_{\Omega_2 \cap \Omega_1} \hat{u}_{i-1}|_{\Omega_1 \cap \Omega_2} \tag{3.96}$$

$$A|_{\Omega_2} u_i|_{\Omega_2} = f|_{\Omega_2} \tag{3.97}$$

$$B|_{\Omega_2 \cap \Omega_1} \hat{u}_i|_{\Omega_1 \cap \Omega_2} = B|_{\Omega_1 \cap \Omega_2} u_i|_{\Omega_1 \cap \Omega_2}, \tag{3.98}$$

where $i = 1, 2, \ldots, I$ and we start from an initial guess of $u_0|_{\Omega_2}$.

Here, we iterate via the two decoupled equations and achieve $u_i|_{\Omega_1 \cap \Omega_2} = \hat{u}_i|_{\Omega_1 \cap \Omega_2}$ for $i$ sufficient large.

At least we have to *double* the variables at the *artificial* boundary.

#### 3.4.7.1 Domain Decomposition Methods: Discussion

The motivation of domain decomposition methods arose to the fact of decomposing into smaller and simpler calculatable domains. We want to apply a standard solver code for each domain, based on the same model equations, see [23].

We can classify the following techniques:

- Non-iterative methods, e.g. FETI methods, Mortar element methods, [24, 25],
- Iterative methods, e.g. Schwarz waveform relaxation methods, see [26].

#### 3.4.7.2 Iterative Method: Schwarz Waveform Relaxation Method

The Schwarz waveform relaxation method deals with the idea to iterate over the decoupled domains. The model equation, the decomposition methods and the underlying software codes are discussed in the manuscript [27].

We deal with the following equations as

$$-\frac{\partial^2 u}{\partial x^2} + \eta u = f, \text{ in } \Omega = [0,1], \tag{3.99}$$

$$u(0) = g_g, \; u(1) = g_d, \tag{3.100}$$

and then separate them into the following equations and apply the iterative steps:

$$-\frac{\partial^2 u_1^{n+1}}{\partial x^2} + \eta u_1^{n+1} = f, \text{ in } \Omega_1 = [0,\beta], \tag{3.101}$$

$$u_1^{n+1}(0) = g_g,$$

$$u_1^{n+1}(\beta) = u_2^n(\beta),$$

$$-\frac{\partial^2 u_2^{n+1}}{\partial x^2} + \eta u_2^{n+1} = f, \text{ in } \Omega_2 = [\alpha,1], \tag{3.102}$$

$$u_2^{n+1}(0) = g_d,$$

$$u_2^{n+1}(\alpha) = u_1^{n+1}(\alpha).$$

We can deal with the different ideas to partition the domains, e.g. overlapping or non-overlapping, see Fig. 3.5.

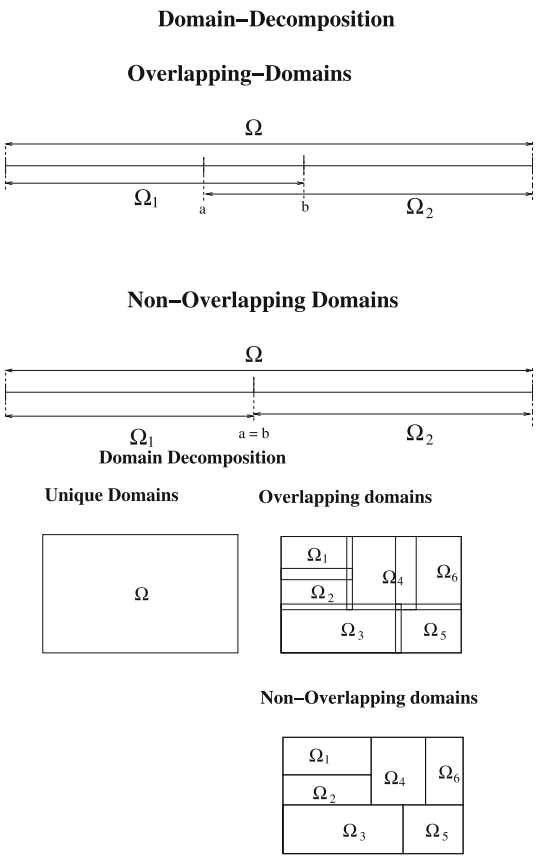



**Fig. 3.5** Domain decomposition with respect to overlapping and non-overlapping domains

Further, we discretize our decomposed equation and we deal with the following discretized equations as

$$-\frac{u_{j+1}-2u_j+u_{j-1}}{h^2}+\eta u_j=f_j,\ 1\le j\le J, \tag{3.103}$$

and then decompose into

$$-\frac{(u_1^{n+1})_{j+1}-2(u_1^{n+1})_j+(u_1^{n+1})_{j-1}}{h^2}+\eta(u_1^{n+1})_j=f_j,$$
$$1\le j\le b-1,\ (u_1^{n+1})_b=(u_2^n)_b, \tag{3.104}$$

$$-\frac{(u_2^{n+1})_{j+1}-2(u_2^{n+1})_j+(u_2^{n+1})_{j-1}}{h^2}+\eta(u_2^{n+1})_j=f_j,$$
$$a+1\le j\le J,\ (u_2^{n+1})_a=(u_1^{n+1})_a. \tag{3.105}$$

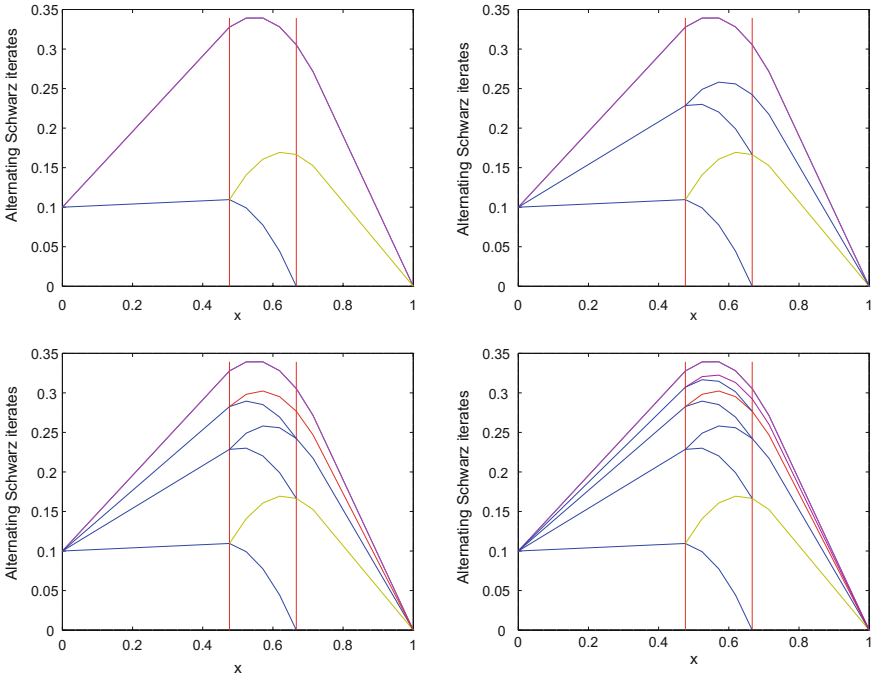



**Fig. 3.6** We start with the initialization (0th iteration, *upper left* figure), then we have the first iteration (*upper right* figure), then we conclude with the second iteration (*lower left* figure) and at least the third iteration (*lower right* figure) of the Schwartz waveform relaxation method

Then, we have the following Schwartz waveform iterative steps, where the exact solution of the decomposed equation is also in the different iterative solutions. The iterative steps are given in Fig. 3.6.

*Remark 3.4* Some ideas and motivations for using non-iterative or iterative domain decomposition methods are discussed below:

1. Iterative Method:
    a. Benefit: Simple to implement, and an inversion of a matrix is not necessary,
    b. Drawback: We deal with an iteration method, i.e. we need relaxation-steps to obtain the *correct* solution (additional time).
2. Non-iterative Method:
    a. Benefit: We obtain a direct solution, we have only one step.
    b. Drawback: Often delicate solver methods for the solutions are needed, e.g., Schur-complement methods for the coupled matrices (to solve such systems, it is necessary to apply iterative solvers).

*Remark 3.5* The benefits and drawbacks of the overlapping or non-overlapping decomposition methods are discussed below. Here are some ideas:

1. Overlapping Domain Decomposition:
   a. Benefit: Stronger coupling of the equation-parts, i.e. we achieve more stable methods, which converge faster.
   b. Drawback: Higher computational amount and more delicate to parallelize.
2. Non-overlapping Domain Decomposition:
   a. Benefit: Simpler to parallelize (it is stronger decoupled).
   b. Drawback: Solver amount is higher, while we need additional iterative steps to couple the equation parts. Often, we have a slower convergence of the method.

In the following Example 3.5, we explain an application of the Schwarz waveform relaxation method.

*Example 3.5* **Application to Convection-Diffusion-Reaction Equations**

The example is given in the author's paper [28]. Here, we conclude with some ideas and aspects of the decomposition method.

We consider the convection-diffusion-reaction equation, given by

$$u_t = Du_{xx} - vu_x - \lambda u, \tag{3.106}$$

defined on the domain $\Omega \times T$, where $\Omega = [0, L]$ and $T = [T_0, T_f]$, with the following boundary and initial conditions:

$$u(0,t) = f_1(t), \quad u(L,t) = f_2(t), \quad u(x,T_0) = u_0.$$

To solve the model problem using overlapping Schwarz waveform relaxation method, we subdivide the domain $\Omega$ into two overlapping subdomains $\Omega_1 = [0, L_2]$ and $\Omega_2 = [L_1, L]$, where $L_1 < L_2$ and $\Omega_1 \bigcap \Omega_2 = [L_1, L_2]$ are the overlapping regions for $\Omega_1$ and $\Omega_2$, respectively.

To start the waveform relaxation algorithm, we consider first the solution of the model problem (3.106) over $\Omega_1$ and $\Omega_2$ as follows:

$$\begin{aligned} v_t &= Dv_{xx} - vv_x - \lambda v \text{ over } \Omega_1, \quad t \in [T_0, T_f] \\ v(0,t) &= f_1(t), \quad t \in [T_0, T_f] \\ v(L_2,t) &= w(L_2,t), \quad t \in [T_0, T_f] \\ v(x,T_0) &= u_0 \quad x \in \Omega_1, \end{aligned} \tag{3.107}$$

$$\begin{aligned} w_t &= Dw_{xx} - vw_x - \lambda w \text{ over } \Omega_2, \quad t \in [T_0, T_f] \\ w(L_1,t) &= v(L_1,t), \quad t \in [T_0, T_f] \\ w(L,t) &= f_2(t), \quad t \in [T_0, T_f] \\ w(x,T_0) &= u_0 \quad x \in \Omega_2, \end{aligned} \tag{3.108}$$

where $v(x,t) = u(x,t)|_{\Omega_1}$ and $w(x,t) = u(x,t)|_{\Omega_2}$.

Then the Schwarz waveform relaxation is given by

$$\begin{aligned} v_t^{k+1} &= Dv_{xx}^{k+1} - \nu v_x^{k+1} - \lambda v^{k+1} \text{ over } \Omega_1, \quad t \in [T_0, T_f] \\ v^{k+1}(0,t) &= f_1(t), \quad t \in [T_0, T_f] \\ v^{k+1}(L_2,t) &= w^k(L_2,t), \quad t \in [T_0, T_f] \\ v^{k+1}(x,T_0) &= u_0 \quad x \in \Omega_1, \end{aligned} \tag{3.109}$$

$$\begin{aligned} w_t^{k+1} &= Dw_{xx}^{k+1} - \nu w_x^{k+1} - \lambda w^{k+1} \text{ over } \Omega_2, \quad t \in [T_0, T_f] \\ w^{k+1}(L_1,t) &= v^k(L_1,t), \quad t \in [T_0, T_f] \\ w^{k+1}(L,t) &= f_2(t), \quad t \in [T_0, T_f] \\ w^{k+1}(x,T_0) &= u_0 \quad x \in \Omega_2. \end{aligned} \tag{3.110}$$

We are interested in estimating the decay of the error of the solution over the overlapping subdomains obtained with the overlapping Schwarz waveform relaxation method over long time interval.

Let us assume that $e^{k+1}(x,t) = u(x,t) - v^{k+1}(x,t)$ and $d^{k+1}(x,t) = u(x,t) - w^{k+1}(x,t)$ are the errors of (3.109) and (3.110) over $\Omega_1$ and $\Omega_2$, respectively. The corresponding differential equations satisfied by $e^{k+1}(x,t)$ and $d^{k+1}(x,t)$ are

$$\begin{aligned} e_t^{k+1} &= De_{xx}^{k+1} - \nu e_x^{k+1} - \lambda e^{k+1} \text{over } \Omega_1, \quad t \in [T_0, T_f] \\ e^{k+1}(0,t) &= 0, \quad t \in [T_0, T_f] \\ e^{k+1}(L_2,t) &= d^k(L_2,t), \quad t \in [T_0, T_f] \\ e^{k+1}(x,T_0) &= 0 \quad x \in \Omega_1, \end{aligned} \tag{3.111}$$

$$\begin{aligned} d_t^{k+1} &= Dd_{xx}^{k+1} - \nu d_x^{k+1} - \lambda d^{k+1} \text{over } \Omega_2, \quad t \in [T_0, T_f] \\ d^{k+1}(L_1,t) &= e^k(L_1,t), \quad t \in [T_0, T_f] \\ d^{k+1}(L,t) &= 0, \quad t \in [T_0, T_f] \\ d^{k+1}(x,T_0) &= 0, \quad x \in \Omega_2. \end{aligned} \tag{3.112}$$

We define for bounded functions $h(x,t) : \Omega \times [T_0, T_f] \to \mathbf{R}$ the norm

$$||h(.,.)||_\infty := \sup_{x \in \Omega, t \in [T_0, T_f]} |h(x,t)|.$$

The theory behind our error estimates is based on the positivity lemma by Pao (or the maximum principle theorem), which is introduced as follows.

**Lemma 3.1** *Let $u \in C(\overline{\Omega}_T) \cap C^{1,2}(\Omega_T)$, where $\Omega_T = \Omega \times (0,T]$ and $\partial\Omega_T = \partial\Omega \times (0,T]$, be such that*

$$u_t - D\, u_{xx} + \nu\, u_x + c\, u \geq 0, \quad \textit{in } \Omega_T \tag{3.113}$$

$$\alpha_0\, \partial u \partial \nu + \beta_0\, u \geq 0, \quad \textit{on } \partial\Omega_T \tag{3.114}$$

$$u(x,0) \geq 0, \quad \textit{in } \Omega \tag{3.115}$$

*where* $\alpha_0 \geq 0$, $\beta_0 \geq 0$, $\alpha_0 + \beta_0 > 0$ *on* $\partial\Omega_T$, *and* $c \equiv c(x,t)$ *is a bounded function in* $\Omega_T$, *Then* $u(x,t) \geq 0$ *in* $\Omega_T$.

The convergence and error estimates of $e^{k+1}$ and $d^{k+1}$ given by (3.111) and (3.112), respectively, are presented in the following theorem.

**Theorem 3.5** *Let* $e^{k+1}$ *and* $d^{k+1}$ *be the errors from the solution of the sub-problems (3.107) and (3.108) by Schwarz waveform relaxation over* $\Omega_1$ *and* $\Omega_2$, *respectively, then*

$$||e^{k+2}(L_1,t)||_\infty \leq \gamma ||e^k(L_1,t)||_\infty,$$

*and*

$$||d^{k+2}(L_2,t)||_\infty \leq \gamma ||d^k(L_1,t)||_\infty,$$

*where*

$$\gamma = \frac{\sinh(\beta L_1)}{\sinh(\beta L_2)} \frac{\sinh(\beta(L_2 - L))}{\sinh(\beta(L_1 - L))} < 1,$$

*with* $\beta = \frac{\sqrt{v^2 + 4D\lambda}}{2D}$.

*Proof* The proof is given in [28].

## References


1. K. Dekker, J.G. Verwer, *Stability of Runge-Kutta Methods for Stiff Nonlinear Differential Equations* (North-Holland Elsevier Science Publishers, Amsterdam, 1984)
2. B. Sportisse, An analysis of operator splitting techniques in the stiff case. J. Comput. Phys. **161**, 140–168 (2000)
3. E. Hairer, G. Wanner, *Solving Ordinary Differential Equations II.*, SCM, vol. 14 (Springer, Heidelberg, 1996)
4. S. Vandewalle, *Parallel Multigrid Waveform Relaxation for Parabolic Problems* (B.G. Teubner Stuttgart, Teubner Skripten zur Numerik, 1993)
5. M.J. Gander, A.M. Stuart, Space-time continuous analysis of waveform relaxation for the heat equation. SIAM J. Sci. Comput. **19**(6), 2014–2031 (1998)
6. P. Vabishchevich, Additive schemes (splitting schemes) for some systems of evolutionary equations. Math. Comput. **83**(290), 2787–2797 (2014)
7. E.J. Davison, A high order Crank-Nicholson technique for solving differential equations. Comput. J. **10**(2), 195–197 (1967)
8. R. Glowinski, in *Numerical Methods for Fluids*. Handbook of Numerical Analysis, vol. IX, ed. by P.G. Ciarlet, J. Lions (North-Holland Elsevier, Amsterdam, 2003)
9. J. Kanney, C. Miller, C.T. Kelley, Convergence of iterative split-operator approaches for approximating nonlinear reactive transport problems. Adv. Water Res. **26**, 247–261 (2003)
10. E. Hansen, A. Ostermann, Exponential splitting for unbounded operators. Math. Comput. **78**, 1485–1496 (2009)
11. T. Jahnke, C. Lubich, Error bounds for exponential operator splittings. BIT Numer. Math. **40**(4), 735–745 (2000)
12. M.J. Gander, S. Vanderwalle, Analysis of the parareal time-parallel time-integration method. SIAM J. Sci. Comput. **29**(2), 556–578 (2007)

13. G. Strang, On the construction and comparison of difference schemes. SIAM J. Numer. Anal. **5**, 506–517 (1968)
14. J.L. Lions, Y. Maday, G. Turincini, A "parareal" in time discretization of PDEs. C. R. Acad. Sci. Paris Sér. I Math. **332**, 661–668 (2001)
15. J. Geiser, St. Guettel, Coupling methods for heat-transfer and heat-flow: operator splitting and the parareal algorithm. J. Math. Anal. Appl. **388**(2), 873–887 (2012)
16. H.G. Bock, K.J. Plitt, A multiple shooting algorithm for direct solution of optimal control problems, in *Proceedings 9th IFAC World Congress Budapest* (Pergamon Press, 1984), pp. 243–247
17. J. Geiser, in *Coupled Systems: Theory, Models and Applications in Engineering*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.H. Lai (CRC Press, Boca Raton, 2014)
18. J. Geiser, in *Iterative Splitting Methods for Differential Equations*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.H. Lai (Chapman & Hall/CRC, Boca Raton, 2011)
19. A.R. Gourlay, Proc. Conf. Univ. York, *Splitting methods for time-dependent partial differential equations*, The state of art in numerical analysis (Academic Press, London, 1976). Heslington 1997
20. T. Lu, P. Neittaanmaki, X.-C. Tai, A parallel splitting-up method for partial differential equations and its applications to Navier-Stokes equations. RAIRO Model. Math. Anal. Numer. **26**, 673–708 (1992)
21. P. Csomós, I. Faragó, A. Havasi, Weighted sequential splittings and their analysis. Comput. Math. Appl. **50**(7), 1017–1031 (2005)
22. J. Geiser, Computing exponential for iterative splitting methods. J. Appl. Math. Hindawi Publishing Corp., New York, Article ID 193781, 27 pp. (2011)
23. A. Quarteroni, A. Valli, *Domain Decomposition Methods for Partial Differential Equations*, Series: Numerical Mathematics and Scientific Computation (Clarendon Press, Oxford, 1999)
24. A. Toselli, O. Widlund, *Domain Decomposition Methods-Algorithms and Theory*. Springer Series in Computational Mathematics, vol. 34 (Springer, Berlin, 2004)
25. C. Farhat, J. Mandel, F.-X. Roux, Optimal convergence properties of the FETI domain decomposition method. Comput. Methods Appl. Mech. Eng. **115**, 365–385 (1994)
26. M.J. Gander, Y.-L. Jiang, R.-J. Li, Parareal Schwarz Waveform Relaxation methods. *Domain Decomposition Methods in Science and Engineering XX*. Lecture Notes in Computational Science and Engineering, vol. 91 (Springer, New York, 2013), pp. 451–458
27. M.J. Gander, L. Halpern, *Methodes de decomposition de domainess*. Lecture Notes, Section de Mathematiques, Universite de Geneve, Switzerland, 26, April 2012
28. D. Daoud, J. Geiser, Overlapping Schwarz wave form relaxation for the solution of coupled and decoupled system of convection diffusion reaction equation. Appl. Math. Comput. **190**(1), 946–964 (2007)

# Chapter 4
# Models and Applications

**Abstract** In this section, we discuss the different multicomponent and multiscale models, which are later applied in simulations. We focus on the coupling of microscopic and macroscopic models, while the microscopic model is related on finer spatial and time scales and the macroscopic model is related to the coarser spatial and time scales. We discuss exemplary engineering problems in the field of electronic application and transport reaction applications in Plasma models. Here, the models and their underlying multiscale and multicomponent methods are discussed. Based on the aligned methods, we see the data flow between the disparate scales and can estimate the accuracy in each micro- and macroscopic model, such that we obtained truly working multiscale and multicomponent approaches.

We deal with the following characterization based on the different spatial and time scales of the models, where we decompose the models into the following, see [1]:

- Microscopic Models: Multicomponent Kinetics (discrete treatment) and
- Macroscopic Models: Multicomponent Fluids (continuous treatment).

Further, we deal with multiscale models, which covered the different microscopic and macroscopic scales and applied methods to overcome the large-scale differences, see [2].

*Remark 4.1* We concentrate on multiscale models, which describes different models —e.g. a microscopic and macroscopic model—and also only macroscopic models but with embedded microscopic scales to resolve material properties—e.g. electromagnetic behaviour of a magnetizable fluid, see [3, 4].

## 4.1 Multicomponent Fluids

**Abstract** In this section, we discuss the models and applications based on the different multicomponent fluid models. Here, we assume to have a macroscopic scale, i.e. we can upscale the microscopic behaviour into the macroscopic scales. We deal with a continuum description and discuss some models based on the multicomponent



J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5_4

fluid problems. Here, standard splitting and multiscale methods are modified with respect to the requirements of the applications. Then, we can close the gap between pure theoretical treatment of numerical methods and their numerical analysis and the necessary adaptation of such standard numerical schemes to engineering applications with the relation to the model problems.

## *4.1.1 Multicomponent Transport Model for Atmospheric Plasma: Modelling, Simulation and Application*

### 4.1.1.1 Introduction

In the following, we discuss a multicomponent transport model for atmospheric (normal pressure) plasma applications.

In such models, it is important to take into account the mixture of the plasma species.

We are motivated to understand atmospheric plasmas within non-thermal equilibrium, which are applied in etching, deposition and sterilization applications, see [5, 6]), and further in emission filtering processes.

We deal with weakly ionized gas mixtures and chemical reactions in room temperature. Each behaviour of a single species and the mixture is complex and needs additional mixture terms that extend the standard models, see [7–10].

Furthermore, the motivation arose of different applications in the so-called jet stream plasma apparatus, for example [11–13]. In such applications, the understanding of the flow and reaction of the species are important.

We assume to deal with a modelling in a time- and spatial- scale, which we can decompose into heavy particles (molecules, atoms, ions) and light particles (electrons)—i.e. we have $Kn \ll 1.0$ where $Kn$, Knudsen number, is the ratio of the molecular mean free path length to a representative physical length scale—e.g. length of the apparatus, and therefore, we can apply a macroscopic model.

In the following, we discuss the so-called macroscopic models, also called fluid models, for the plasma model, which is discussed in [14, 15].

We present the special models with respect to their benefits, starting from a two-component fluid model till a multicomponent fluid model with Stefan–Maxwell equation for the mixture of the species. With such a complex model, we achieve an optimal mixture model, which represents the individual single heavy particle.

The underlying conservation laws result in the equations of mass, momentum and energy and additional with conditions related to the Stefan–Maxwell equation, e.g. summation of the mass rates is 1 ($\sum_{i=1} w_i = 1$) and summation of the mass fluxes is 0 ($\sum_{i=1} j_i = 0$).

Such equations with additional conditions are quasilinear, strong coupled parabolic differential equations, see [16].

Such equations need a larger computational amount based on the nonlinearities in the diffusion part. Standard models, based on the Fickian's approach, compared the ideas in [17], are much more simpler to solve and the extended model has taken into account singularities and nonlinear behaviours, see [16, 18, 19].

In the following, we discuss step-by-step approach of the novel models and the development of the underlying solver methods.

#### 4.1.1.2 Introduction and Overview

Since recent years, the application in normal pressure plasmas arose important and therefore the understanding of the reactive chemical species in the plasma and during its mixture is necessary. For such delicate problems, the standard models which are known in the literature have to be extended by the reactive parts of the mixture. Such an important detail can be modelled by the diffusion operator, and the Stefan–Maxwell equation is a possibility to take into account such mixture behaviours, see [16].

In such reactive plasmas, we obtain due to the typical known processes, as ionization and collision, and additional processes, the so-called chemical reactions.

Such chemical processes are dominant for normal pressure plasmas and they are used in the plasma medicine technology.

While they applied air as a plasma background, we have the highly reactive elements oxygen $O^2$ and nitrogen $N^2$ in the complex gas mixtures.

Therefore, it is important to extend the standard modelling and simulation techniques, see [20, 21], and embed the nonlinear structures of the Stefan–Maxwell approach.

The diffusive processes are modelled by the so-called *multicomponent diffusion*, which are more and more studied in the Stefan–Maxwell approaches in fluid-dynamical models, see [22].

We obtain an improvement of the so-called binary diffusion processes in the transport reaction models, if we have no dominant species, e.g. only minor at species, which means we do not have a dominant background matrix. Such observations made it necessary to deal with a more detailed modelling, see [17, 23–25].

In comparison to pure fluid-dynamical models, see porous media models [26] or elementary modelling [27], or so-called neutral fluids, in macroscopical plasma models, we have additional terms, for example electric fields. We assume additional to deal with weak-ionized particles that such weak-ionized heavy particles can be modelled by a multicomponent fluid model, vgl. [14].

In modelling plasmas, we deal with a so-called scaling, which allows to distinguish between macroscopic plasma models and microscopic plasma models, confer Table 4.1 and Fig. 4.1.

**Table 4.1** Parameters for the macro- and microplasma and their applications

| Mean free path length electrons | Pressure | Temperature | Length of the reactor | |
|---|---|---|---|---|
| | | | Plasma | Neutral gas |
| *MacroPlasma (CCP, ICP: Etching and Deposition)* | | | | |
| 0.01–1 (cm) | 1–100 (Pa) | 300–500 (K) | ≈10 (cm) | ≈100 (cm) |
| *MicroPlasma (Plasmajets, DBD: Deposition and Sterilization)* | | | | |
| 1 (μm) | $10^5$ (Pa) | 300–500 (K) | 0.1–1 (mm) | 1–10 (cm) |

Both plasmas have the same characteristics in the Knudsen number, i.e. $Kn \ll 1$ and can be treated and simulated as macroscopic models

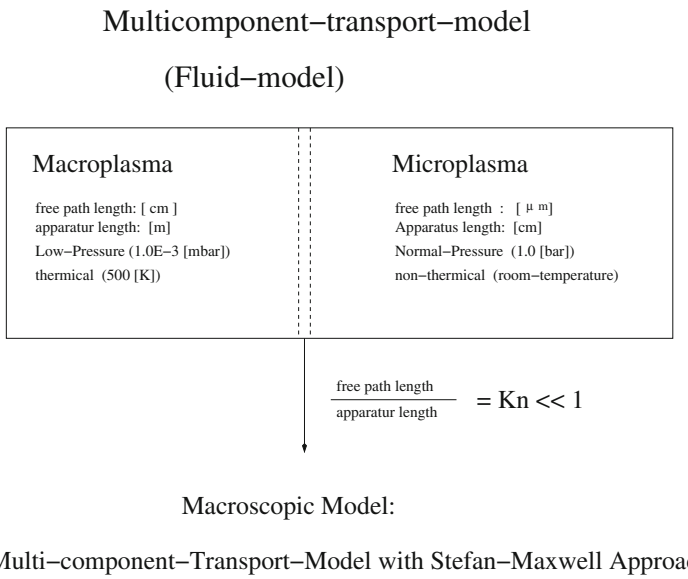



**Fig. 4.1** Macroscopic plasma models

We discuss the following steps in the next sections:

- In Sect. 4.1.1.3, we discuss the derivation of the multicomponent transport models. We begin with a simple model (two-component fluid model) and end up with a delicate multicomponent transport model (multifluid flow model).
- In Sect. 4.1.1.4, we discuss the mathematical classification and the numerical treatment of such delicate transport models with embedded Stefan–Maxwell approximations.
- The conclusions are discussed in Sect. 4.1.1.5.

#### 4.1.1.3 Discussion of the Multicomponent Transport Models for Normal Pressure Plasmas

We deal in the following with the so-called hierarchical model equations, see [28, 29], which approximate the behaviour of the normal pressure plasmas.

For the first start, we can simply deal with a two-fluid formulation, where we decouple heavy particles (ions, molecules, atoms) into light particles (electrons). Furthermore, a more appropriate model is done with the multifluid formulation, where we can apply for each heavy particle species (e.g. we distinguish between the different ions and atoms of $O, N, \ldots$) and apply an individual distribution function.

Furthermore, we extend the transport equations with the Stefan–Maxwell equation, see the ideas in [25].

As a start point to derive the hierarchical equations with heavy and light particles in the plasma bulk, we use the Boltzmann equation:

$$\frac{\partial}{\partial t} f + \mathbf{v} \cdot \nabla_{\mathbf{x}} f + \frac{q}{m} (\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}} f = \langle f \rangle, \tag{4.1}$$

- $f$: Density function of the a general particle species;
- $\mathbf{v}$: General velocity in the bulk;
- $q$: Particle charge in general;
- $m$: Mass of the species;
- $\langle f \rangle$: Collision term in general;
- $\mathbf{E}$: Electrical field vector; and
- $\mathbf{B}$: Magnetical field vector.

For the heavy particle in general and electrons, we derive the fluid model with the help of the velocity moments to obtain the macroscopic quantities, see [30].

**Two-Component Fluid Model**

In the following, we assume a simple description of the all heavy particles $i$ (i.e. all ions and neutrons) and all electrons $e$.

We have the following Assumption 4.1:

**Assumption 4.1**
- We concentrate on the density function of the heavy particles (we neglect the electrons, based on their relative small mass compared to the ions and neutrons).
- We assume that we do not have mixture of the different species and we only have to model the pure transport of one particle species.
- An exact distribution function is not necessary for such regimes, while we do not consider a kinetic behaviour.
- The extension between electrons and heavy particles (e.g. scattering) is sufficient by a approximated collision term, see also [15].

By applying the velocity momentums, we obtain the conservation equations of the heavy particles $i$ and the electrons $e$, in the following equation with $\alpha = \{i, e\}$, see also [14]:

$$\frac{\partial \rho_\alpha}{\partial t} + \nabla_{\mathbf{x}} \cdot (\rho_\alpha \mathbf{u}) = m_\alpha Q_n^{(\alpha)}, \tag{4.2}$$

$$\frac{\partial}{\partial t} \rho_\alpha \mathbf{u}_\alpha + \nabla_{\mathbf{x}} \cdot \left( \rho_\alpha \mathbf{u}_\alpha \mathbf{u}_\alpha + nT \underline{I} - \underline{\tau}^* \right) = q_\alpha n_\alpha (\mathbf{E} + \mathbf{u}_\alpha \times \mathbf{B}) - Q_m^e, \tag{4.3}$$

$$\frac{\partial}{\partial t} E_{total}^* + \nabla_{\mathbf{x}} \cdot \left( E_{total}^* \mathbf{u} + \mathbf{q}^* + nT\mathbf{u} - \underline{\tau}^* \cdot \mathbf{u} \right) = q_\alpha n_\alpha \mathbf{E} - Q_\varepsilon^{(e)}, \tag{4.4}$$

- $\rho_\alpha$: Mass density of the species $\alpha$;
- $\mathbf{u}_\alpha$: Averaged velocity of the species $\alpha$;
- $Q_n^\alpha$, $Q_m^e$, $Q_\varepsilon^e$: Collision integral based on the mass, momentum and energy conservation;
- $q_\alpha$: Heat flow of the species $\alpha$;
- $n_\alpha$: Density of species $\alpha$;
- $\mathbf{E}$: Electrical field vector;
- $\mathbf{B}$: Magnetical field vector; and
- $E_{total}^*$: Total energy of all species.

Furthermore, we have to add the Maxwell equations for the electro-magnetic field, see [15].

**Multicomponent Fluid Model with Fickian's Approach without Stefan–Maxwell Approach)**
In the following, we apply a first multicomponent model based on the work of [9, 14], where all the heavy particles are described. The Fickian's approach is used and we assume to have dominant species, e.g. majorant species, which can be applied as a matrix background such that binary diffusion is sufficient, see [25].

We have therefore the following Assumption 4.2.

**Assumption 4.2** The assumptions for the Fickian's approach are given as follows:

- Each heavy particle species is described with an individual density function.
- We apply only a simple summation of the transport parameters, which results in a phenomenological result (not the derivation with Stefan–Maxwell equations).
- The electrons are modelled in the same manner as in the two-component fluid model, see [15].

We have the following notation and constraints of the heavy particle.
The notation for the multicomponent formulation is given as follows:

- $N$: Number of species;
- $n_s$: Particle density of species $s$, $s = 1, \ldots, N$;
- $n = \sum_{s=1}^{N} n_i$: Total particle density;
- $T$: Particle energy of all heavy particles, e.g. $T = k_B T_{gas}$;
- $\rho = \sum_{s=1}^{N} \rho_s$: Mass density of all particles with $\rho_s$ as the mass particle density of species $s$;

- $\rho_s = m_s n_s$, $n_s$, $m_s$: Mass of species $s$;
- $c_s = u_s - u$, $c_s$: Difference or diffusion velocity of species $s$;
- $u_s$: Drift velocity of species $s$; and
- $u$: Drift velocity of the total system and given as $u = \frac{1}{\rho} \sum_{s=1}^{N} \rho_s u_s$.

The model equation with the binary diffusion coefficients, see in the paper of Senega/Brinkmann [14], is given for the heavy particles $s \in \{1, \ldots, N\}$:

$$\frac{\partial}{\partial t} n_s + \nabla_{\mathbf{x}} \cdot (n_s \mathbf{u}_s + n_s \mathbf{c}_s) = Q_n^{(s)}, \tag{4.5}$$

$$\frac{\partial}{\partial t} \rho \mathbf{u} + \nabla_{\mathbf{x}} \cdot \left( \rho \mathbf{u}\mathbf{u} + nT \underline{I} - \underline{\tau}^* \right) = \sum_{s=1}^{N} q_s n_s \langle E \rangle, \tag{4.6}$$

$$\begin{aligned} \frac{\partial}{\partial t} E_{total}^* + \nabla_{\mathbf{x}} \cdot \left( E_{total}^* \mathbf{u} + \mathbf{q}^* + nT\mathbf{u} - \underline{\tau}^* \cdot \mathbf{u} \right) \\ = \sum_{s=1}^{N} q_s n_s (\mathbf{u} + \mathbf{c}_s) \cdot \langle E \rangle - Q_{inel,ST}^{(e)}, \end{aligned} \tag{4.7}$$

where

$$E_{total}^* = \sum_{s=1}^{N} \frac{1}{2} \rho_s \mathbf{c}_s^2 + \frac{1}{2} \rho \mathbf{u}^2 + \frac{3}{2} nT + \sum_{s=1}^{N} \rho_s \Delta h_{f,s}^0, \tag{4.8}$$

see also in paper [14].

An improvement of the standard derivation of such models is obtained with the individual density functions for all different heavy particle species, such that we obtain the following representation for the values $\mathbf{c}_s$, $\mathbf{q}^*$ and $\underline{\tau}^*$ with

$$\mathbf{c}_s = -d_T^{(s)} \nabla_{\mathbf{x}} T - \sum_{\alpha=1}^{N} D_n^{(\alpha,s)} \frac{1}{n_s} \nabla_{\mathbf{x}} n_\alpha, \tag{4.9}$$

$$\mathbf{q}^* = \lambda_E \langle E \rangle - \lambda \nabla_{\mathbf{x}} T - \sum_{s=1}^{N} \sum_{\alpha=1}^{N} \lambda_n^{(\alpha,s)} \frac{1}{n_s} \nabla_{\mathbf{x}} n_\alpha, \tag{4.10}$$

$$\underline{\tau}^* = -\eta \left( \nabla_{\mathbf{x}} \mathbf{u} + (\nabla_{\mathbf{x}} \mathbf{u})^T - \frac{2}{3} (\nabla_{\mathbf{x}} \cdot \mathbf{u}) \underline{I} \right). \tag{4.11}$$

The production terms (e.g. collision terms, reaction terms) are approximated in the following operators:

$$Q_n^{(e)} = \int_{v_s} \langle f_s \rangle d^3 v_s = \sum_r a_{sign,r} k_{\alpha,r} n_\alpha n_r, \tag{4.12}$$

$$Q_n^{(s)} = \int_{v_s} \langle f_s \rangle d^3 v_s = \sum_r a_{sign,r} k_{\alpha,r} n_\alpha n_r, \tag{4.13}$$

where $k_{\alpha,r}$ is the parameter of the averaged collision rates, see [14] and $a_{sign,r}$ is the signum function, $a_{sign,r} = 1$ is a source term and $a_{sign,r} = -1$ is a sink term.

**Multicomponent Fluid Model with Stefan–Maxwell Equation**

In the following, we discuss the extended multicomponent description, which is generalized via the Stefan–Maxwell approach.

The Stefan–Maxwell equation allows a systematical derivation of the diffusion processes, where the mixture of the different species is considered, such that we can also discuss counter diffusion, which is possible in ternary diffusion processes. Also, the thermodynamical behaviour is discussed accurately without heuristic assumptions as in the Fickian's approach.

We discuss in the following the extension of the transport parameters with respect to the Stefan–Maxwell equation, see [16].

We assume the following:

- Each heavy particle species can be described with an individual density function.

Our notations are used as in the section "Multicomponent Fluid model with Stefan–Maxwell Equation".

We apply the transport equation:

$$\frac{\partial}{\partial t} n_s + \nabla_{\mathbf{x}} \cdot (n_s \mathbf{u}_s + n_s \mathbf{c}_s) = Q_n^{(s)}, \tag{4.14}$$

with the diffusion velocity:

$$\mathbf{c}_s = -d_T^{(s)} \nabla_{\mathbf{x}} T - \sum_{\alpha=1}^{N} D_n^{(\alpha,s)} \frac{1}{n_s} \nabla_{\mathbf{x}} n_\alpha, \tag{4.15}$$

which is extended in the following with the Stefan–Maxwell equation.

We decompose into two fluxes:

$$\mathbf{c}_s = \mathbf{c}_{s,1} + \mathbf{c}_{s,2}, \tag{4.16}$$

where $c_{s,1}$ is the thermal flux and $c_{s,2}$ is the diffusive flux

$$\mathbf{c}_{s,1} = -d_T^{(s)} \nabla_{\mathbf{x}} T, \tag{4.17}$$

$$\mathbf{c}_{s,2} = j_s, \tag{4.18}$$

where $j_s$ is the so-called driving force of the species $s$.

In our case, we restrict us to the chemical potential as driving force:

$$j_s = n_s \nabla_{\mathbf{x}} \mu_s, \tag{4.19}$$

where $\mu_s = \log(\gamma_s n_s)$ and $\gamma_s$ is the so-called activation constant ($\gamma_s > 0$) and we obtain

$$j_s = \nabla_{\mathbf{x}} n_s, \tag{4.20}$$

where $\sum_{s=1}^{N} j_s = 0$, i.e. the sum of all fluxes is equal to 0 and the also the sum of the mass rates is zero

$$\sum_{s=1}^{N} y_s = 0, \tag{4.21}$$

where $y_s = \frac{\rho_s}{\rho}$.

The Stefan–Maxwell equation is given as

$$j_s = \left( \sum_{j=1}^{N} \frac{1}{\tilde{D}_{sj}} (y_s j_j - y_j j_s) \right). \tag{4.22}$$

We can compute the flux matrix $j = (j_1, \ldots, j_N)^T \in \mathbb{R}^{N \times N}$, where $j^s$ is the column vector of $j$ with $M = \mathrm{diag}(m_s)$, $e = [1, \ldots, 1]^T$, $P(y) = I - y \otimes e = I - (\cdot, |e)y$ (where $\otimes$ is the dyadic product), and we obtain the equation

$$\begin{cases} B(y) j^{\alpha} = P(y) M^{-1} \partial_{x_\alpha} y, \ \alpha = 1, \ldots, n, \\ B(y) = [b_{ij}(y)], \ b_{ij}(y) = f_{ij} y_i, \\ \text{fur } i \neq j, b_{ii}(y) = -\sum_{l=1}^{N} f_{il} y_l, \ i, j = 1, \ldots, N, \end{cases} \tag{4.23}$$

furthermore, $\tilde{D}_{ij} = f_{ij}, \ i, j = 1, \ldots, N$ is the multidiffusion coefficient and $n$ is the number of spatial dimensions, e.g. $n = 2$ or $n = 3$, see [19].

#### 4.1.1.4 Solver Ideas for the Multicomponent System with the Stefan–Maxwell Equation

We can apply different numerical schemes to solve the multicomponent system with Stefan–Maxwell equation. Some are discussed in the following:

- Implicit Ideas: Solve the coupled nonlinear transport equation with relaxation methods.
- Explicit Ideas: Direct solving of Stefan–Maxwell equations, where we apply the overdetermined equation system and solve analytically the parameters (such a

analytical method is very delicate and only applicable to binary or ternary diffusion operators, see [31]).
- Variational formulations: We apply an additional Poisson's equation to solve the constraint of the Stefan–Maxwell equation. We obtain a saddle point problem, which can be solved by standard mixed finite element methods.

**Implicit Method**

We apply an implicit method with iteration scheme and rewrite the full equation system into a quasilinear, strong coupled parabolic differential equation, where we consider for simplicity only the mass conservation:

$$\rho \frac{\partial}{\partial t} y + \mathrm{Div}_x(A(y)P(y)M^{-1}[\nabla_x y]^T)) = Q_n, \text{ in } \Omega, \ t > 0, \tag{4.24}$$

$$\frac{\partial}{\partial n} y = 0, \text{ auf } \partial\Omega, \ t > 0, \tag{4.25}$$

$$y(0) = y_0, \text{ in } \Omega, \tag{4.26}$$

$y = (y_1, \ldots, y_N)$ and $Q_n = \rho(m_1, Q_n^1, \ldots, m_N, Q_n^N)$. Furthermore, we have $A(y) = (B(y)|_{Ext})$, where the matrix $B$ is extended to an invertable matrix. We have $\nabla_x y = [\partial_\alpha y_j] \in \mathbb{R}^{n \times N}$ and $\mathrm{Div}_x$ is the divergence in each row of the matrix.

We can show, under some conditions, that we have a existing solution, see [19].

Based on the existing solution, we can apply the following iterative scheme, for the time intervals $n = 0, \ldots, N$ and iterative steps $i = 0, \ldots, I$:

$$U'_{i+1} = A_1(U_i)U_{i+1} + A_2(U_i)U_i, \ t \in [t^n, t^{n+1}], \tag{4.27}$$

$$U_i(t^n) = U(t^n), \tag{4.28}$$

where $U(t^n)$ is the approximated solution of the last iterative cycle and $U_0(t)$ is an estimated initial or starting solution for the next cycle, e.g. $U_0(t) = U(t^n)$. The stopping criterion is given as $||U_{i+1}(t^{n+1}) - U_i(t^{n+1})|| \leq err$ or the limit of the number of iterative steps $i = I$. Furthermore, the operator $A_1$ is the convection part and $A_2$ is the diffusion part of the transport equation.

The iterative method is convergent with the assumption of the existence of the solution and boundedness of the operators, see [20, 21].

**Explicit Method**

Here, we have the benefit of a fast and direct solver for small systems, e.g. binary or ternary systems.

A main drawback is the application to larger systems of quarternary or higher mixtures, while it is hard to find the explicit equations.

We show the method based on a three-component system, given as

$$\partial_t \xi_i + \nabla \cdot N_i = 0, \ 1 \le i \le 3, \tag{4.29}$$

$$\sum_{j=1}^{3} N_j = 0, \tag{4.30}$$

$$\frac{\xi_2 N_1 - \xi_1 N_2}{D_{12}} + \frac{\xi_3 N_1 - \xi_1 N_3}{D_{13}} = -\nabla \xi_1, \tag{4.31}$$

$$\frac{\xi_1 N_2 - \xi_2 N_1}{D_{12}} + \frac{\xi_3 N_2 - \xi_2 N_3}{D_{23}} = -\nabla \xi_2, \tag{4.32}$$

where we have $\Omega \in \mathbb{R}^d$, $d \in \mathbb{N}^+$ mit $\xi_i \in C^2$.

We can simplify to

$$\partial_t \xi_i + \nabla \cdot N_i = 0, \ 1 \le i \le 2, \tag{4.33}$$

$$\frac{1}{D_{13}} N_1 + \alpha N_1 \xi_2 - \alpha N_2 \xi_1 = -\nabla \xi_1, \tag{4.34}$$

$$\frac{1}{D_{23}} N_2 - \beta N_1 \xi_2 + \beta N_2 \xi_1 = -\nabla \xi_2, \tag{4.35}$$

where $\alpha = \left(\frac{1}{D_{12}} - \frac{1}{D_{13}}\right)$, $\beta = \left(\frac{1}{D_{12}} - \frac{1}{D_{23}}\right)$.

We obtain the explicit solvable Stefan–Maxwell equation with the multidiffusion coefficients:

$$D_{12} = \tilde{D}_{12} \left[ 1 + \frac{\frac{w_3}{M_3}\left(\frac{M_3}{M_2}\tilde{D}_{13} - \tilde{D}_{12}\right)}{\frac{w_1}{D_1}\tilde{D}_{23} + \frac{w_2}{D_2}\tilde{D}_{13}} + \frac{w_3}{D_3}\tilde{D}_{12} \right], \tag{4.36}$$

where $\tilde{D}_{ij}$ are the binary diffusion coefficients, $M_i$ is the molar mass of the species $i$ and $w_i$ is the mass rate of the species $i$, see also the derivation in the paper [18, 32].

**Variational Formulation**

Here, we can apply standard software codes, which are done in the direction of the Poisson's equation.

A drawback of the method is that we have to solve a saddle point problem, which needs iterative solver methods, which are expensive and apply special solver schemes, e.g. Lagrangian multipliers.

Formulation with respect to the Poisson's equation is

$$-\Delta u = r, \ \text{in } \Omega, \tag{4.37}$$

$$u = f, \ \text{auf } \partial\Omega, \tag{4.38}$$

where $\partial\Omega$ is the boundary of the domain $\Omega$.

A solution of the problem equation is given via a mixed formulation as a saddle point problem:

$$p - \nabla u = 0, \text{ in } \Omega, \tag{4.39}$$

$$\nabla \cdot p = r, \text{ in } \Omega, \tag{4.40}$$

which means that we find a solution for the mixed formulation of $(p, u) \in Q \times V$:

$$\int_{\Omega} (pq + u\nabla \cdot q) = \int_{\partial\Omega} fq \cdot nds, \tag{4.41}$$

$$\int_{\Omega} v\nabla \cdot pdx = -\int_{\Omega} rvdx, \tag{4.42}$$

where $n$ is the outer normal vector of $\partial\Omega$.

The variational formulation of the Stefan–Maxwell equation is given as

$$\int_{\Omega} (-\nabla \xi_i q_i) dx = \int_{\Omega} \frac{1}{c_{tot}} \left( \sum_{i=1}^{N} \alpha_{ij} (\xi_i J_j q_i - \xi_j J_i q_i) \right) dx, \tag{4.43}$$

$$\int_{\Omega} v_i \nabla \cdot J_i dx = \int_{\Omega} r_i v_i dx, \tag{4.44}$$

where $r_i$ is the reaction rate (e.g. collision term), $J_i$ is the flux, and $\xi_i$ is the molar rate of species $i$. $c_{tot}$ is the total concentration of the mixture.

**Regularization Method: Regularization of the Transport Model with Stefan–Maxwell Equation**

There exist several more methods; a well-known idea is the regularization method.

We start with the macroscopic model and extend the Stefan–Maxwell equation to a regular and solvable system.

The flux term is given as

$$c_s = \frac{1}{\rho_s} \nabla j_s, \tag{4.45}$$

where $j_s$ are the mass flux densities with the following constraints:

- $\sum_{s=1}^{N} j_s = 0$ (i.e. all fluxes are zero), and
- $\sum_{s=1}^{N} w_s = 1$ (i.e. all mass rates are 1),

where $x_s = w_s \frac{\tilde{M}}{M_s}$ and $x_s$ are the molar rates, $M_s$ is the molar mass of species $s$, $\tilde{M}$ is the molar mass of the mixture and further the density of the mixture is given as $\tilde{\rho} = (1 - \sum_{s=1}^{N-1} w_s)\rho_N + \sum_{s=1}^{N-1} w_s \rho_s$.

The Stefan–Maxwell approach is the equibalance of the molar rates for each individual diffusive flux:

$$-\nabla x_s = \frac{\tilde{M}}{\tilde{\rho}} \left( \sum_{j=1}^{N} \frac{1}{\tilde{D}_{sj}} \left( x_s \frac{j_j}{M_j} - x_j \frac{j_s}{M_s} \right) \right), \tag{4.46}$$

and we obtain the equation system

$$FV = -d, \tag{4.47}$$

where $d = (\nabla x_1, \ldots, \nabla x_M)^t$, $V = (j_1, \ldots, j_M)^t$ and $F$ is a singular matrix of the equation system (4.46).

The next step is the regularization of the singular equation system and we obtain the novel diffusion matrix:

$$\tilde{F} = F + \alpha y \otimes y, \tag{4.48}$$

where $y = (n_1, \ldots, n_M)^t$, $\alpha$ is a parameter for the solver method and $\otimes$ is the dyadic product.

Based on this regularisation, we can apply a standard iterative method and solve the Stefan-Maxwell equation and also the heavy-particle equations together in a large linear equation system. Such a combination allows to apply fast linear equation solvers, e.g. SuperILU solvers.

#### 4.1.1.5 Conclusion

The extension of the known standard heavy particle model with an improved diffusion part can be done with the Stefan–Maxwell equation.

The former summation approach is replaced by the balance approach, see [16, 18, 19], which is done with the Stefan–Maxwell equation.

The former modelling approaches are extended and the solver methods can be applied. But we have to extend also the analytical or numerical methods for the singular perturbed novel equation system.

Therefore, we have to modify the simulation packages with respect to the novel diffusion part.

While explicit methods to solve the Stefan–Maxwell equations are fast and simple to implement, they lack with larger systems and larger species in the mixture. Implicit methods are more flexible and also resolve higher mixtures but are more time-consuming in the computations, while we apply iterative schemes.

At the end, it is an approach how large the systems and the mixtures are, while for small systems, we apply an explicit method and for large systems we have to apply an implicit approach.

### 4.1.2 Multicomponent Fluid Transport Model for Groundwater Flow

We concentrate on such models, which deal with the transport behaviour of fluids in the porous media, see [26].

Such models arose of the background to understand flow of water in aquifers, transport of pollutants in aquifers or underlying rocks and propagation of stresses, see [26, 33].

We concentrate on introducing the mathematical models, see also [34–36] and discuss possible solver methods to simulate such models.

#### 4.1.2.1 Introduction and Mathematical Model

We consider a steady-state groundwater flow that is described by a given velocity field $\boldsymbol{v} = \boldsymbol{v}(x)$ for $x \in \Omega \subset R^d$ for $d = 2$ or $d = 3$. In the groundwater, several radionuclides (or some other chemical species) are dissolved.

We suppose that these nuclides take part in irreversible, first-order chemical reactions. Particularly, each nuclide (a "mother") can decay only to a single component (to a "daughter"), but each nuclide can be produced by several reactions, i.e. each daughter can have several mothers, see [34].

Moreover, the radionuclides can be adsorbed to the soil matrix. If equilibrium linear sorption is assumed with different sorption constants for each component, the advective–dispersive transport of each component is slowed down by a different retardation factor.

Summarizing, the mathematical model can be written in the form [33, 34]

$$R^{(i)}\phi\left(\partial_t c^{(i)} + \lambda^{(ij)} c^{(i)}\right) + \nabla \cdot \left(\boldsymbol{v} c^{(i)} - D^{(i)} \nabla c^{(i)}\right) = \sum_k R^{(k)} \phi\, \lambda^{(ki)} c^{(k)}, \quad (4.49)$$

where $i = 1, \ldots, I_c$. The integer $I_c$ denotes the total number of involved radionuclides. A stationary groundwater is supposed by considering only divergence-free velocity field, i.e.

$$\nabla \cdot \boldsymbol{v}(x) = 0, \quad x \in \Omega. \quad (4.50)$$

The unknown functions $c^{(i)} = c^{(i)}(t, x)$ denote the concentrations of radionuclides, where the space and time variables $(t, x)$ are considered as $t \geq 0$ and $x \in \Omega$. The constant reaction rate $\lambda^{(ij)} \geq 0$ determines the decay (sink) term $\lambda^{(ij)} c^{(i)}$ for the concentration $c^{(i)}$ and the production (source) term for the concentration $c^{(j)}$. In general, the $j$th radionuclide need not to be included in the system (4.49), i.e. $j > I_c$. The indices $k$ in the right- hand side of (4.49) run through all mothers of the $i$th radionuclide.

The remaining parameters in (4.49) include the diffusion–dispersion tensors $D^{(i)} = D^{(i)}(x, \boldsymbol{v})$ [33], the retardation factors $R^{(i)} = R^{(i)}(x) \geq 1$ and the porosity of medium $\phi = \phi(x) > 0$.

For the modelling of processes on the boundary $\partial\Omega$ of the domain $\Omega$, we apply standard inflow and outflow boundary conditions. Particularly, we neglect the diffusive–dispersive flux at the outflow (and "noflow") boundary $\partial^{out}\Omega := \{x \in \partial\Omega,\ \boldsymbol{n} \cdot \boldsymbol{v} \geq 0\}$,

$$\boldsymbol{n} \cdot D^{(i)} \nabla c^{(i)}(t, \gamma) = 0, \quad t > 0, \quad \gamma \in \partial\Omega, \tag{4.51}$$

where $\boldsymbol{n}$ is the normal unit vector with respect to $\partial\Omega$. For the case of inflow boundary $\partial^{in}\Omega := \{x \in \partial\Omega,\ \boldsymbol{n} \cdot \boldsymbol{v} < 0\}$, we assume that the concentrations are prescribed by Dirichlet boundary conditions:

$$c^{(i)}(t, \gamma) = C^{(i)}(t, \gamma), \quad t > 0, \quad \gamma \in \partial^{in}\Omega. \tag{4.52}$$

The functions $C^{(i)}$ can describe decay reactions in a waste site (e.g. a nuclear waste repository), and, in such a way, they shall be related to each other, see, e.g. [37].

The initial conditions are considered in a general form:

$$c^{(i)}(0, x) = C^{(i)}(0, x), \quad x \in \Omega. \tag{4.53}$$

#### 4.1.2.2 Solver Ideas for the Multicomponent Fluid Transport Model

If we assume simple domains, e.g. one-dimensional problems and special boundary and initial conditions, for the problem (4.49), we could derive analytical solutions, see for example [37].

Such analytical solutions solve the multicomponent behaviour analytically in an explicit equation.

For more general applications, e.g. multidimensions and general boundary and initial conditions, it is necessary to deal with a discretized equation.

Here, we have the following methods to discretize the spatial operators, for example:

- Finite element methods, see [38] and
- Finite-volume methods, see [39].

We concentrate on the finite-volume scheme, which allow to deal with the conservation equation and apply geometrically the derivation of the convection and diffusion term, see [40].

The finite-volume discretization method, see [42], allows to deal with a general velocity $\boldsymbol{v} = \boldsymbol{v}(x)$ and general boundary and initial conditions (4.51)–(4.53).

We have the following ideas:

- We apply analytical solutions for locally one-dimensional advection-reaction problems on boundaries between two finite volumes, see also Godunov algorithm [40]; and
- We split the diffusion part of (4.49) using operator splitting procedure and apply finite-volume method, see [41].

If we have nonlinearities, we apply a linearization method, e.g. fixpoint scheme or Newton's method. Based on the linearized equations in (4.49), linear splitting schemes can be applied and decoupled to several simpler problems. Applying afterwards the principle of superposition, one can obtain the solution of (4.49) by summing the solutions of such simpler problems.

#### 4.1.2.3 Splitting Method for the Multicomponent Fluid Transport Model

We decompose the multicomponent fluid transport equation into a convection-reaction part and a diffusion part.

While the convection-reaction part is solved exactly with one-dimensional solutions and Godunov's scheme is applied, the diffusion part is solved in the spatial operators with finite-volume discretization scheme and in the time operator with implicit time discretization.

**Convection-Reaction Part**

We apply the following convection-reaction equation:

$$\partial_t \left( R^{(l)} \phi u^{(l)} \right) + \nabla \cdot \left( v u^{(l)} \right) + \lambda^{(l)} R^{(l)} \phi u^{(l)} = \lambda^{(l-1)} R^{(l-1)} \phi u^{(l-1)}. \quad (4.54)$$

We apply the Godunov's method which means the solution of the one-dimensional convection-reaction equations, which are embedded as mass transfer to the finite-volume scheme, see [42].

So we solve for each underlying one-dimensional $\Omega_i$ and the mass concentration to the out-flowing cell $j \in out(i)$, a one-dimensional convection-reaction equations for each species $l = 1, \ldots, I$:

$$R_i^{(l)} \phi_i \partial_t u_i^{(l)} + v_{ij} \, \partial_x u_i^{(l)} + \lambda^{(l)} R_i^{(l)} \phi_i u_i^{(l)} = \lambda^{(l-1)} R_i^{(l-1)} \phi_i u_i^{(l-1)}. \quad (4.55)$$

We transform to a directly solvable convection-reaction system, with the following as

$$c_i^{(l)} := R_i^{(l)} \phi_i u_i^{(l)}, \; \tilde{v}_{ij} = \frac{v_{ij}}{R_i^{(l)} \phi_i}, \quad (4.56)$$

and we obtain

$$\partial_t c_i^{(l)} + \tilde{v}_{ij} \, \partial_x c_i^{(l)} + \lambda^{(l)} c_i^{(l)} = \lambda^{(l-1)} c_i^{(l-1)}. \quad (4.57)$$

For each cell, we compute the total outflow fluxes

$$\tau_{l,i} = \frac{V_i \; R^{(l)}}{v_i}, \quad v_i = v_{ij} \; , \; j = out(i).$$

Based on the restriction of the local time, we have the minimum over all possible cell time steps:

$$\tau^n \leq \min_{\substack{l=1,\ldots,m \\ i=1,\ldots,I}} \tau_{l,i},$$

and we obtain a velocity of the finite-volume cell:

$$v_{l,i} = \frac{1}{\tau_{l,i}}.$$

Then, we can calculate the mass, which is important to embed into the FV discretization:

$$\begin{aligned} m_{ij,rest}^{(l),n} &= m_1^{(l),n}(a, b, \tau^n, v_{1,i}, \ldots, v_{l,i}, R^{(1)}, \ldots, R^{(l)}, \lambda^{(1)}, \ldots, \lambda^{(l)}), \\ m_{ij,out}^{(l),n} &= m_2^{(l)}(a, b, \tau^n, v_{1,i}, \ldots, v_{l,i}, R^{(1)}, \ldots, R^{(l)}, \lambda^{(1)}, \ldots, \lambda^{(l)}), \end{aligned}$$

where $a = V_i R^{(l)}(c_{ij}^{(l),n} - c_{ij'}^{(l),n})$, $b = V_i R^{(l)} c_{ij'}^{(l),n}$ and $m_i^{(l),n} = V_i R^{(l)} c_i^{(l),n}$ are the parameters and $j = out(i)$, $j' = in(i)$.

The discretization with the embedded analytical mass is given by

$$m_i^{(l),n+1} = m_{ij,rest}^{(l),n} \; + \; m_{j'i,out}^{(l),n},$$

where $m_{ij,rest}^{(l),n} = m_i^{(l),n} - m_{ij,out}^{(l),n}$ is rest mass coming from the total mass and the outflown mass, see [42].

**Diffusion Part**

We discretize the diffusion part with the finite-volume methods. We can concentrate on the following equation:

$$\partial_t R \; c - \nabla \cdot (D \nabla c) = 0, \tag{4.58}$$

where $c = c(x, t)$ with $x \in \Omega$ and $t \geq 0$. The diffusion is given as $D \in \mathbb{R}^+$ and the retardation factor is $R > 0.0$.

The equation is integrated over time and space (implicit time and mass averaging in space):

$$\int_{\Omega_j}\int_{t^n}^{t^{n+1}} \partial_t R(c)\; dt\; dx = \int_{\Omega_j}\int_{t^n}^{t^{n+1}} \nabla\cdot(D\nabla c)\; dt\; dx. \tag{4.59}$$

After applying Green's formula and the approximation in the finite cells (i.e. $\Gamma_j$ is the boundary of the finite-volume cell $\Omega_j$), we have for one finite cell

$$V_j R(c_j^{n+1}) - V_j R(c_j^n) = \tau^n \sum_{e\in\Lambda_j}\sum_{k\in\Lambda_j^e} |\Gamma_{jk}^e| \mathbf{n}_{jk}^e \cdot D_{jk}^e \nabla c_{jk}^{e,n+1}, \tag{4.60}$$

where $|\Gamma_{jk}^e|$ is the length of the boundary element $\Gamma_{jk}^e$.

We calculate the gradients via piecewise finite element function $\phi_l$ and obtain

$$\nabla c_{jk}^{e,n+1} = \sum_{l\in\Lambda^e} c_l^{n+1} \nabla\phi_l(\mathbf{x}_{jk}^e). \tag{4.61}$$

Then, we obtain the finite-volume discretization for the diffusion part:

$$\begin{aligned} &V_j R(c_j^{n+1}) - V_j R(c_j^n) \\ &\quad = \tau^n \sum_{e\in\Lambda_j}\sum_{l\in\Lambda^e\setminus\{j\}} \Big(\sum_{k\in\Lambda_j^e} |\Gamma_{jk}^e| \mathbf{n}_{jk}^e \cdot D_{jk}^e \nabla\phi_l(\mathbf{x}_{jk}^e)\Big)(c_j^{n+1} - c_l^{n+1}), \end{aligned} \tag{4.62}$$

where the finite cells are given as $j = 1, \ldots, m$.

For such a discretization, we can embed the convection-reaction part via a splitting approach, which is given in the following.

**Coupling Part**

The different parts of the full equations are coupled via a operator splitting method.

We apply the following splitting approach:

$$c^*(t^{n+1}) = c(t^n) + \tau_n A c(t^n) \tag{4.63}$$

$$c^{**}(t^{n+1}) = c^*(t^{n+1}) + \tau_n B c^{**}(t^{n+1}), \tag{4.64}$$

where the time step is $\tau^n = t^{n+1} - t^n$ and $n = 1, \ldots, N$ are the number of time steps. The operator $A$ is the convection-reaction operator, which can be resolved in the equation analytically. The operator $B$ is the diffusion operator, which is solved via FV methods and implicit Euler method.

Based on the analytical resolution of the convection-reaction part, we have the following splitting approach:

$$c^{**}(t^{n+1}) = (I - \tau_n B)^{-1}\; c^*(t^{n+1}), \tag{4.65}$$

where $c^*(t^{n+1})$ is the analytical solution of the convection-reaction part.

The splitting error is of the first order based on the non-commuting operators, see [42].

*Remark 4.2* Based on the analytical embedding of the convection-reaction equation, we can speed up the solver scheme and concentrate on solving the diffusion part. Here, based on the first-order splitting scheme, we can see the method as following: The diffusion equation is only perturbed by a convection-reaction part, see [43].

### *4.1.3 Conclusion*

For the multicomponent fluid transport model, it is important to decompose into simpler and faster solvable equation-parts. Each equation-part, e.g. convection-reaction part or diffusion part, can be solved with more adequate schemes, which are more effective and faster as a full equation solver. We have applied fast solver methods for the convection-reaction part, e.g. modified Godunov's method embedded to finite volume schemes, and for the diffusion part, e.g. finite volume schemes to discretize the spatial operators. The parts are coupled with fast operator splitting schemes, which allow to concentrate on the diffusion solver, while the convection-reaction part can be embedded as on explicit solved part. Such effective methods allow to solve the multicomponent fluid transport model with high accuracy and acceleration. In future, an extension of multicomponent fluid transport models with respect to additional equation-parts, e.g. multiphase parts or growth parts, are possible and the splitting schemes can be modified to such additional parts.

## 4.2 Multicomponent Kinetics

**Abstract** In this section, we discuss the models and applications based on the different multicomponent kinetic models. Here, we assume to have a microscopic scale, i.e. we deal with the fine resolution in the atomic scale. So we have a discrete description and discuss some models based on the multicomponent kinetics problems.

### *4.2.1 Multicomponent Langevin-Like Equations*

The idea is to apply an alternative model based on the Coulomb collision in plasma to reduce the computational in particle simulations.

The alternative models are based on Langevin equations, which are coupled nonlinear stochastic differential equations, see [44].

Historically, we have two ideas for algorithms for Coulomb collisions in particle simulations:

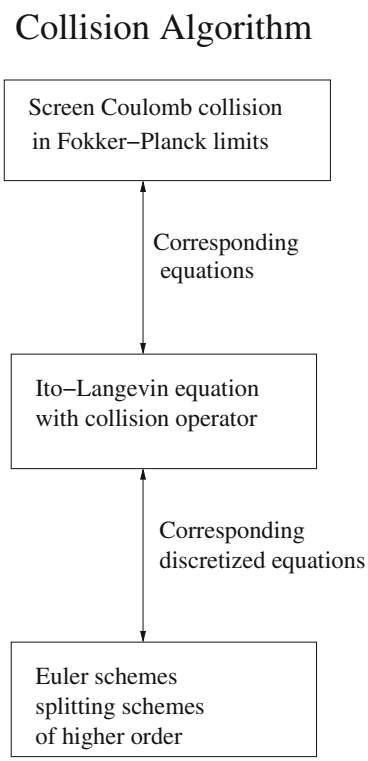



**Fig. 4.2** Screen Coulomb collision in the Fokker–Planck limit

- Binary algorithm: Particles in a finite cell, see particle in cell, are organized into discrete pairs (therefore binary algorithm) of interacting particles. The collision is based on oulomb collision of two particles, see [45].
- Test particle algorithm: The collisions are modelled by defining a dual particles (test particles) and primary particles (field particles). The velocity of the test particle is modelled by Langevin equation, which is deposited on the space mesh [46].

The idea of the alternative approach is given in Fig. 4.2.

The main contribution to deal with the stochastic model is based on the following Remark 4.3 of the Coulomb collision approach:

*Remark 4.3* Coulomb collisions can be approximated via defining test and field particles. The test particle velocity is subjected to drag and diffusion in three velocity dimensions using Langevin equations, see [47].

### 4.2.2 Introduction to the Model Equations

We are motivated to develop fast algorithms to solve Fokker–Planck equation with Coulomb collisions in plasma simulations.

The Fokker–Planck equations are given as

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} - E(x)\frac{\partial f}{\partial v} = \frac{\partial}{\partial v}\left(-\gamma v f + \beta^{-1}\gamma\frac{\partial f}{\partial v}\right), \tag{4.66}$$

where we could decouple such a FP equation into the PIC (particle in cell) part and the SDE part.

- PIC part

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} - E(x)\frac{\partial f}{\partial v} = 0, \tag{4.67}$$

- SDE part

$$\frac{\partial f}{\partial t} = \frac{\partial}{\partial v}\left(-\gamma v f + \beta^{-1}\gamma\frac{\partial f}{\partial v}\right), \tag{4.68}$$

where we solve the characteristics.

- PIC part

$$\frac{dx}{dt} = v, \tag{4.69}$$

$$\frac{dv}{dt} = -E(x) = \frac{\partial U}{\partial x}, \tag{4.70}$$

  where $U$ is the potential.
- SDE part

$$\frac{dx}{dt} = 0, \tag{4.71}$$

$$dv = -\gamma v dt + \sqrt{2\beta^{-1}\gamma}dW. \tag{4.72}$$

We apply the following nonlinear SDE problem:

$$\frac{dx}{dt} = v, \tag{4.73}$$

$$dv(t) = \frac{\partial}{\partial x}U(x) - \gamma v dt + \sqrt{2\beta^{-1}\gamma}dW, \tag{4.74}$$

where $W$ is a Wiener process, $\gamma$ is the thermostat parameter and $\beta$ is the inverse temperature.

A long solution to the SDE is distributed according to a probability measure with density $\pi$ satisfying

$$\pi(x, v) = C^{-1} \exp\left(-\beta\left(\frac{v^2}{2} + U(x)\right)\right), \tag{4.75}$$

where $x > 0.0, v \in \mathbb{R}$.

### 4.2.3 Analytical Methods for Mixed Deterministic–Stochastic Ordinary Differential Equations

In the following, we present an algorithm, which is based on solving the mixture of deterministic and stochastic ordinary differential equations.

The idea is based on the deterministic variation of constants to embed perturbed right-hand sides.

We deal with the following equations:

$$\frac{dX}{dt} = V, \tag{4.76}$$

$$dV = -E(x)dt - AVdt + BdW,$$

$$\text{with } X(0) = X_0,\ V(0) = V_0, \tag{4.77}$$

where $W$ is a Wiener process with the $N(0, \sqrt{\Delta})$ distributed.

We rewrite to a linear operator and a nonlinear and stochastic function.

$$\frac{d\mathbf{X}}{dt} = \tilde{A}\mathbf{X} + \mathbf{E}(\mathbf{X}) + \frac{d\mathbf{W}}{dt},$$

$$\text{with } \mathbf{X}_0 = (X_0, V_0)^t, \tag{4.78}$$

where $\mathbf{X} = (X, V)^t$ is the solution vector, $\mathbf{X}_0 = (X_0, V_0)^t$ is the initial vector, the matrix is $\tilde{A} = \begin{pmatrix} 0 & 1 \\ 0 & -A \end{pmatrix}$, the nonlinear function is $\mathbf{E} = \begin{pmatrix} 0 \\ -E(X) \end{pmatrix}$ and the stochastic function is $\frac{d\mathbf{W}}{dt} = \begin{pmatrix} 0 \\ B\frac{dW}{dt} \end{pmatrix}$.

The analytical solution is given with the exact integration of the $\exp(\tilde{A}s)$ (variation of constants):

$$\begin{aligned} \mathbf{X}(t^{n+1}) &= \exp(\tilde{A}\Delta t)\mathbf{X}_0 + \int_{t^n}^{t^{n+1}} \exp(\tilde{A}(t^{n+1} - s))\ \mathbf{E}(\mathbf{X}(s))\ ds \\ &\quad + \int_{t^n}^{t^{n+1}} \exp(\tilde{A}(t^{n+1} - s))\ d\mathbf{W}_s, \\ \mathbf{X}(t^{n+1}) &= \exp(\tilde{A}\Delta t)\mathbf{X}_0 + \tilde{\mathbf{E}}(\mathbf{X}_0) + \tilde{\mathbf{W}}(\mathbf{X}_0), \end{aligned} \tag{4.79}$$

where is the electric field integral is computed with a higher order exponential Runge–Kutta method.

Integration of the E-field function with fourth-order Runge–Kutta method is as follows:

$$\mathbf{k}_1 = \Delta t \mathbf{E}(\mathbf{X}^n), \tag{4.80}$$

$$\mathbf{k}_2 = \Delta t (\mathbf{E}(\exp(\tilde{A}\Delta t/2)\mathbf{X}^n + \frac{1}{2}\exp(\tilde{A}\Delta t/2)\mathbf{k}_1)), \tag{4.81}$$

$$\mathbf{k}_3 = \Delta t (\mathbf{E}(\exp(\tilde{A}\Delta t/2)\mathbf{X}^n + \frac{1}{2}\mathbf{k}_2)), \tag{4.82}$$

$$\mathbf{k}_4 = \Delta t (\mathbf{E}(\exp(\tilde{A}\Delta t)\mathbf{X}^n + \exp(\tilde{A}\Delta t/2)\mathbf{k}_2)), \tag{4.83}$$

$$\tilde{\mathbf{E}}(\mathbf{X}^n) = \frac{1}{6}\Big(\exp(\tilde{A}\Delta t)k_1 + 2\exp(\tilde{A}\Delta t/2)(\mathbf{k}_2 + \mathbf{k}_3) + \mathbf{k}_4)\Big), \tag{4.84}$$

and the stochastic integral is computed as

$$\begin{aligned}\tilde{\mathbf{W}}(\mathbf{X}^n) &= \int_{t^n}^{t^{n+1}} \exp(\tilde{A}(t^{n+1} - s)d\mathbf{W}_s \\ &= \sum_{j=0}^{N-1} \exp\left(\tilde{A}\left(\frac{t^{n,j} + t^{n,j+1}}{2}\right)\right)\ (\mathbf{W}(t^{n,j+1}) - \mathbf{W}(t^{n,j})),\end{aligned} \tag{4.85}$$

$$\Delta t = (t^{n+1} - t^n)/N,\ t^{n,j} = \Delta t + t^{n,j-1},\ t^{n,0} = t^n. \tag{4.86}$$

*Remark 4.4* Based on the perturbation and finer time scales, the stochastic integral is resolved with finer time steps as the non-stochastic parts. Therefore, we have applied an adaptive numerical integration method that allows to apply additional smaller time intervals with more integration points. We obtained more accurate numerical results of the stochastic integral and reduce the numerical error of the full scheme.

### *4.2.4 Conclusion*

For the multicomponent kinetics, we have additional stochastic equation-parts. Therefore, it is important to resolve such stochastic parts with high accurate stochastic solvers. We have highly perturbed and finer time scale to resolve such multiscale parts. In our case, we proposed analytical methods, which solved the stoachstic part with semi-analytical methods and embedded directly the results to the deterministic (non-stochastic) part. Here, we obtain high accurate results, while we could concentrate on the deterministic solver parts. In the future, an extension of multicomponent kinetics to many particle applications, e.g. plasma dynamics, is important and we can apply the idea of the analytical embedding of the stochastic part to the deterministic parts to reduce the computational time.

## 4.3 Additive Operator Splitting with Finite-Difference Time-Domain Method: Multiscale Algorithms

**Abstract** We discuss numerical methods based on additive operator splitting schemes, which are used to solve Maxwell equations, see [48]. The discretization schemes are given with Finite-Difference Time-Domain (FDTD) methods, which apply finite differences in time and space and allow to conserve the physical behaviour of the equations, see [49]. Because of the 3D Maxwell equations, we result into large semi-discretized equation systems, i.e. we have to deal with large systems of ordinary differential equations. Therefore, we are motivated to optimize 3D computations of electro-magnetic fields with decomposition methods, which decompose into different time and spatial scales. Here, we discuss additive operator splitting schemes, which allow to decompose into several independent solvable smaller equation systems, see [2]. We embed the FDTD schemes into the additive splitting and result into a multiscale approach into each spatial dimension.

### *4.3.1 Introduction*

We are motivated to split large semi-discretized equation systems, e.g. resulted from FDTD schemes, see [3], with additive operator splitting schemes, which allow to concentrate on each individual dimension and each time and spatial scale of the underlying reduced equation systems, see [2].

In Fig. 4.3, we present the multiscale splitting with the FDTD discretization scheme and the AOS (additive operator splitting scheme) as a multiscale splitting approach.

While explicit time-discretization schemes have restriction with respect to their CFL (Courant–Friedrichs–Lewy) condition, we also discuss implicit time-discretization schemes based on modified FDTD and AOS schemes, which overcome such restrictions, see [50, 51].

### *4.3.2 Introduction FDTD Schemes*

One of the simplest FDTD schemes is the Yee's algorithm, see [49]. The ideas are given in the following:

- We combine time and space discretization on a time–space grid. Using central difference schemes for both time and space, we obtain second-order methods with respect to the CFL condition of the discretization schemes.
- A staggered grid is necessary to obtain for both time and space second-order schemes and obtain a stable discretization scheme, see [49].

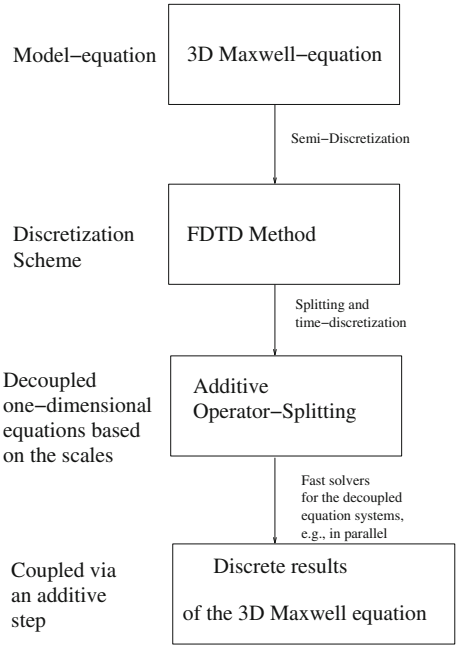



**Fig. 4.3** Splitting approach based on an FDTD discretized Maxwell equation

In the following, we present the so-called Yee's cells in 2D and 3D, see Figs. 4.5 and 4.4.

Such cells are applied with respect to the time and spatial discretization and their staggered behaviours allow to achieve a second-order scheme.

In the following example, we discuss a first-order FDTD method, see Example 4.1.

*Example 4.1* We have the following preparations to achieve the higher order scheme:

- We discretize both time and spackle with a central difference, which is a second-order scheme.
- We decompose into a primary and dual grid, i.e. we apply a staggered grid for the magnetic and electric field equations.
- We step forward in time.

We start with the following 1D equations:

$$\frac{\partial E_x}{\partial t} = -\frac{1}{\varepsilon_0}\frac{\partial H_y}{\partial z}, \tag{4.87}$$

$$\frac{\partial H_y}{\partial t} = -\frac{1}{\mu_0}\frac{\partial E_x}{\partial z}, \tag{4.88}$$

where we have an initial condition of the impulse and adsorbing boundary conditions. We deal with a wave-front solution in the z direction.

We apply the 1D FDTD method as follows:

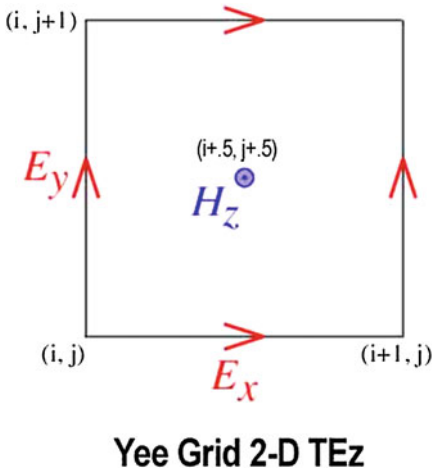



**Fig. 4.4** Staggered grid: 2D

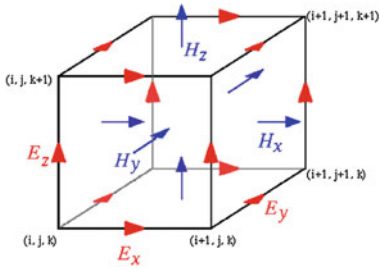



**Fig. 4.5** Staggered Grid für FDTD Methden für 3D

- The simplest 1D FDTD Schema is the Yee's method.
- We stagger $E_x$ and $H_y$ in time and space with a half-time and half-spatial step.
- We apply the central difference scheme for the time and space coordinates.

We obtain the 1D equation as

$$\frac{E_x^{n+1/2}(k) - E_x^{n-1/2}(k)}{\Delta t} = -\frac{1}{\varepsilon_0}\frac{H_y^n(k+1/2) - H_y^n(k-1/2)}{\Delta z}, \tag{4.89}$$

$$\frac{H_y^{n+1}(k+1/2) - H_y^n(k-1/2)}{\Delta t} = -\frac{1}{\mu_0}\frac{E_x^{n+1/2}(k+1) - E_x^{n+1/2}(k)}{\Delta z}, \tag{4.90}$$

where we have a so-called *leap-frog* algorithm, i.e. first we apply $E_x^{n+1/2}$ for all spatial points and then we apply $H_y^{n+1}$ for all spatial points. We step forward in time. For the discretization points of a 1D Yee's algorithm, see Fig. 4.6.

Based on the explicit method, i.e. we step forward in time, we have restrictions for the stability in the time step. The CFL condition for the simple 1D Maxwell equation is given as

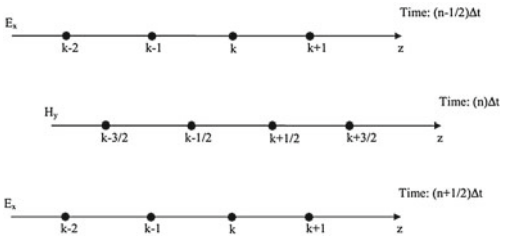



**Fig. 4.6** Staggered grid: 1D

$$\Delta t \leq \frac{\Delta z}{c_0} \tag{4.91}$$

where $c_0$ is the light speed.

*Remark 4.5* For the explicit higher dimensional FDTD methods, we have also the same restriction as for the 1D methods. We have also to restrict our time step in 2D and 3D, as follows, see also [49, 52]:

$$\Delta t \leq \frac{\min_{i=1}^{3}\{\Delta x_i\}}{c_0\sqrt{d}} \tag{4.92}$$

where $\Delta x_i$, $i = 1, \ldots, d$ are the spatial steps, and $c_0$ is the light speed and $d = 2, 3$.

### *4.3.3 Additive Operator Splitting Schemes*

The additive operator splitting scheme can be applied with respect to the different spatial dimensions. Based on their different scales, we can also apply the AOS scheme as a multiscale approach, see [2].

In the following, we discuss additive operator splitting schemes, see also [3].

We describe traditional operator splitting methods and focus our attention to the case of two linear operators, i.e. we consider the Cauchy problem,

$$\partial_t c(t) = \sum_{i=1}^{m} A_m(c) \quad t \in (0, T); \quad c(0) = c_0 \tag{4.93}$$

whereby the initial function $c_0$ is given, and $A_1, \ldots, A_m$ are assumed to be bounded nonlinear operators. (In many applications, they denote the spatially discretized operators, e.g. they correspond to the discretized in space convection and diffusion operators (matrices). Hence, they can be considered as bounded operators.)

We discuss the following schemes:

- AOS (explicit):

$$c^{n+1} = \left( I + t \sum_{i=1}^{m} A_i(c^n) \right) c^n, \tag{4.94}$$

while the method is closely related to the idea of the multiplicative splitting (A–B Splitting) in the explicit form:

$$\exp((A_1(c^n) + \cdots + A_m(c^n))t) = \exp(A_1(c^n)t) \cdot \cdots \cdot \exp(A_m(c^n)t), \tag{4.95}$$

if one apply the explicit Euler to Eq. (4.93) and scheme, you neglect the second-order term $\mathcal{O}(t^2)$.
The scheme can be additively applied as

$$c_i^{n+1} = B_i(c^n)c^n, \ i = 1, \ldots, m, \tag{4.96}$$

$$c^{n+1} = c^n + \sum_{i=1}^{m} c_i^{n+1}, \tag{4.97}$$

with the operators $B_i(c^n) := t \ A_i(c^n))$.

- AOS (semi-implicit):

$$c^{n+1} = \left( I - t \sum_{i=1}^{m} A_i(c^n) \right)^{-1} c^n, \tag{4.98}$$

and further

$$c^{n+1} = \frac{1}{m} \left( \sum_{i=1}^{m} (I - m \ t \ A_i(c^n) \right)^{-1} ) c^n, \tag{4.99}$$

with the operators $B_i(c^n) := \frac{1}{m}(I - m \ t \ A_i(c^n))$
The scheme can be additively applied as

$$c_i^{n+1} = B_i(c^n)^{-1} c^n, \ i = 1, \ldots, m, \tag{4.100}$$

$$c^{n+1} = \sum_{i=1}^{m} c_i^{n+1}. \tag{4.101}$$

### 4.3.4 Application to the Maxwell Equations

We have the following Maxwell equation:

$$\frac{\partial \boldsymbol{E}}{\partial t} = \frac{1}{\varepsilon}\nabla \times \boldsymbol{H} - \frac{1}{\varepsilon}\sigma \boldsymbol{E}, \tag{4.102}$$

$$\frac{\partial \boldsymbol{H}}{\partial t} = -\frac{1}{\mu}\nabla \times \boldsymbol{E}, \tag{4.103}$$

where the operators are $c = \boldsymbol{E}$, $v = \boldsymbol{H}$ and we have the abstract formulation:

$$\frac{\partial \mathbf{c}}{\partial t} = \mathscr{A}\mathbf{c} + \mathscr{A}_4\mathbf{c}, \tag{4.104}$$

with $\mathbf{c} = (c, v)^t$, $\mathscr{A} = \mathscr{A}_1 + \mathscr{A}_2 + \mathscr{A}_3 = \begin{pmatrix} 0 & \frac{1}{\varepsilon}A \\ -\frac{1}{\mu}A^* & 0 \end{pmatrix}$, $\mathscr{A}_4 = \begin{pmatrix} -\frac{\sigma}{\varepsilon}I_{3\times 3} & 0_{3\times 3} \\ 0_{3\times 3} & 0_{3\times 3} \end{pmatrix}$, where we have $\mathscr{A}, \mathscr{A}_1, \mathscr{A}_2, \mathscr{A}_3, \mathscr{A}_4 \in \mathbb{R}^{6\times 6}$ and $A, A_1, A_2, A_3, I_{3\times 3}, 0_{3\times 3} \in \mathbb{R}^{3\times 3}$ with $I_{3\times 3}$ as the identity matrix and $0_{3\times 3}$ as the zero matrix.

The decomposition is given in the following steps. Each full $A = A_1 + A_2 + A_3$ is divided into a single dimension as

$$A_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -\frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial x} & 0 \end{pmatrix}, A_2 = \begin{pmatrix} 0 & 0 & \frac{\partial}{\partial y} \\ 0 & 0 & 0 \\ -\frac{\partial}{\partial y} & 0 & 0 \end{pmatrix}, A_3 = \begin{pmatrix} 0 & -\frac{\partial}{\partial z} & 0 \\ \frac{\partial}{\partial z} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

Here, we have to apply an AOS scheme with four operators.

The full version is given as

$$\frac{\partial \mathbf{c}}{\partial t} = \begin{pmatrix} -\frac{\sigma}{\varepsilon} & 0 & 0 & 0 & -\frac{\partial}{\partial z} & \frac{\partial}{\partial y} \\ 0 & -\frac{\sigma}{\varepsilon} & 0 & \frac{1}{\varepsilon}\frac{\partial}{\partial z} & 0 & -\frac{1}{\varepsilon}\frac{\partial}{\partial x} \\ 0 & 0 & -\frac{\sigma}{\varepsilon} & -\frac{1}{\varepsilon}\frac{\partial}{\partial y} & \frac{1}{\varepsilon}\frac{\partial}{\partial x} & 0 \\ 0 & \frac{1}{\mu}\frac{\partial}{\partial z} & -\frac{1}{\mu}\frac{\partial}{\partial y} & 0 & 0 & 0 \\ -\frac{1}{\mu}\frac{\partial}{\partial z} & 0 & \frac{1}{\mu}\frac{\partial}{\partial x} & 0 & 0 & 0 \\ \frac{1}{\mu}\frac{\partial}{\partial y} & -\frac{1}{\mu}\frac{\partial}{\partial x} & 0 & 0 & 0 & 0 \end{pmatrix} \mathbf{c}, \tag{4.105}$$

based on the equations, and when we apply AOS, we split into the following six matrices:

$$\mathscr{A}_{11} = \begin{pmatrix} -\frac{\sigma}{\varepsilon} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\varepsilon}\frac{\partial}{\partial z} & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{\varepsilon}\frac{\partial}{\partial y} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{4.106}$$

$$\mathscr{A}_{21} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \frac{\partial}{\partial y} \\ 0 & -\frac{\sigma}{\varepsilon} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\varepsilon}\frac{\partial}{\partial x} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{4.107}$$

$$\mathscr{A}_{31} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & \frac{\partial}{\partial y} \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{\varepsilon}\frac{\partial}{\partial x} \\ 0 & 0 & -\frac{\sigma}{\varepsilon} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{4.108}$$

$$\mathscr{A}_{12} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{\mu}\frac{\partial}{\partial z} & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{\mu}\frac{\partial}{\partial y} & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{4.109}$$

$$\mathscr{A}_{22} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{\mu}\frac{\partial}{\partial z} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{\mu}\frac{\partial}{\partial x} & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{4.110}$$

$$\mathscr{A}_{32} = \begin{pmatrix} 0\,0 & 0 & 0\,0\,0 \\ 0\,0 & 0 & 0\,0\,0 \\ 0\,0 & 0 & 0\,0\,0 \\ 0\,0 & -\frac{1}{\mu}\frac{\partial}{\partial y} & 0\,0\,0 \\ 0\,0 & \frac{1}{\mu}\frac{\partial}{\partial x} & 0\,0\,0 \\ 0\,0 & 0 & 0\,0\,0 \end{pmatrix}, \tag{4.111}$$

The splitting is based as follows:

$$\mathbf{c}^{n+1} = \frac{1}{6}\left(\sum_{i=1}^{3}\sum_{j=1}^{2}(I - 6\ \Delta t\ \mathscr{A}_{i,j})^{-1}\right)\mathbf{c}^{n}, \tag{4.112}$$

for example, the first operator is given as

$$\mathbf{c}_1^{n+1} = \frac{1}{6}(I - 6\ \Delta t\ \mathscr{A}_{11})^{-1}\ \mathbf{c}^{n}. \tag{4.113}$$

If we apply the finite difference discretization of a structured grid, we obtain the following matrices:

- We assume to have $N \times N \times N$ grid points, i.e. $H_x, H_y, H_z, E_x, E_y, E_z \in \Omega \in \mathbb{R}^N \times \mathbb{R}^N \times \mathbb{R}^N = \mathbb{R}^{N^3}$.
- The matrices are given as $\mathscr{A}_{i,j} \in 6\mathbb{R}^{N^3} \times 6\mathbb{R}^{N^3}$, where $i = 1, 2, 3$ and $j = 1, 2$.
- For the discretization, we apply the following submatrices: $I \in \mathbb{R}^N \times \mathbb{R}^N$ is the identity matrix, $0 \in \mathbb{R}^N \times \mathbb{R}^N$ is the zero matrix and $M \in \mathbb{R}^N \times \mathbb{R}^N$ which is needed for the difference matrices and given as

$$M = \begin{pmatrix} 0 & 0 & \dots & \dots & 0 \\ -1 & 0 & 0 & \dots & 0 \\ 0 & -1 & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & -1 & 0 \end{pmatrix},$$

- The difference matrices for $M_x, M_y, M_z \in \mathbb{R}^{N^3} \times \mathbb{R}^{N^3}$ are given as

$$M_x = \tfrac{1}{\Delta x}\begin{pmatrix} I+M & 0 & \dots & \dots & 0 \\ 0 & I+M & 0 & \dots & 0 \\ 0 & 0 & I+M & \dots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & I+M \end{pmatrix},$$

$$M_y = \frac{1}{\Delta y}\begin{pmatrix} I & 0 & \dots & \dots & 0 \\ M & I & 0 & \dots & 0 \\ 0 & M & I & \dots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & M & I \end{pmatrix},$$

$$M_z = \frac{1}{\Delta z}\begin{pmatrix} \tilde{I} & 0 & \dots & \dots & 0 \\ \tilde{M} & \tilde{I} & 0 & \dots & 0 \\ 0 & \tilde{M} & \tilde{I} & \dots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \tilde{M} & \tilde{I} \end{pmatrix},$$

where $\tilde{I} \in \mathbb{R}^{N^2} \times \mathbb{R}^{N^2}$ is the identity matrix and $\tilde{M} \in \mathbb{R}^{N^2} \times \mathbb{R}^{N^2}$ is given as

$$\tilde{M} = \begin{pmatrix} M & 0 & \dots & \dots & 0 \\ 0 & M & 0 & \dots & 0 \\ 0 & 0 & M & \dots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & 0 & M \end{pmatrix}.$$

Then for example the first operator is discretized as

$$\mathbf{C}_1^{n+1} = \frac{1}{6}(I_{Disc} - 6\ \Delta t\ \mathscr{A}_{11,Disc})^{-1}\ \mathbf{C}^n, \tag{4.114}$$

where $I_{\mathscr{A}} \in \mathbb{R}^{N^3} \times \mathbb{R}^{N^3}$ is the identity matrix, $0_{\mathscr{A}} \in \mathbb{R}^{N^3} \times \mathbb{R}^{N^3}$ is the zero matrix and we have

$$I_{Disc} = \begin{pmatrix} I_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & I_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & I_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & I_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & I_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & I_{\mathscr{A}} \end{pmatrix}, \tag{4.115}$$

$$\mathscr{A}_{11,Disc} = \begin{pmatrix} \frac{\sigma}{\varepsilon} I_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & \frac{1}{\varepsilon} M_z & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & -\frac{1}{\varepsilon} M_y & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} \\ 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} & 0_{\mathscr{A}} \end{pmatrix}, \tag{4.116}$$

furthermore, we have $\mathbf{C}^{n+1} = (\mathbf{E}_{x,disc}, \mathbf{E}_{y,disc}, \mathbf{E}_{z,disc}, \mathbf{H}_{x,disc}, \mathbf{H}_{y,disc}, \mathbf{H}_{z,disc})^T$ and all $\mathbf{E}_{x,disc}, \mathbf{E}_{y,disc}, \mathbf{E}_{z,disc}, \mathbf{H}_{x,disc}, \mathbf{H}_{y,disc}, \mathbf{H}_{z,disc} \in \mathbb{R}^{N^3}$.

*Remark 4.6* Here, we have an application of a semi-implicit AOS scheme, while the nonlinearity in Eq. (4.93), i.e. $A_i(c^{n+1})$, is approximated via $A_i(c^n)$, which means that we restrict us to the linearization of the previous time point $t^n$ and therefore, we embed also a CFL condition.

### 4.3.5 Practical Formulation of the 3D-FDTD Method

For more practical reasons, we consider on a simpler scheme based on the staggered time step method, such that we apply semi-implicit schemes.

Maxwell's equations in lossy and frequency independent materials are given as

$$\nabla \times \boldsymbol{E} = -\mu \frac{\partial \boldsymbol{H}}{\partial t}, \tag{4.117}$$

$$\nabla \times \boldsymbol{H} = \frac{\partial \boldsymbol{D}}{\partial t}, \tag{4.118}$$

$$\frac{\partial \boldsymbol{D}}{\partial t} = \sigma \boldsymbol{E} + \varepsilon_0 \varepsilon_r \frac{\partial \boldsymbol{E}}{\partial t} \tag{4.119}$$

where $\sigma$ is the conductivity, $\mu$ is the permeability, $\varepsilon_0$ is the vacuum permittivity, $\varepsilon_r$ is the relative permittivity, $\boldsymbol{E}$ is the electric field, $\boldsymbol{D}$ is the electric flux density and $\boldsymbol{H}$ is the magnetic field. Equation (4.118) is Maxwell–Ampere equation without free currents.

We apply the operator $\nabla\times$ to the equations

$$\begin{aligned}\nabla \times \boldsymbol{E} &= \left(\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z}\right)\boldsymbol{i}_x + \left(\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x}\right)\boldsymbol{i}_y + \left(\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y}\right)\boldsymbol{i}_z \\ &= -\mu \frac{\partial (H_x \boldsymbol{i}_x + H_y \boldsymbol{i}_y + H_z \boldsymbol{i}_z)}{\partial t},\end{aligned} \tag{4.120}$$

$$\begin{aligned}\nabla \times \boldsymbol{H} &= \left(\frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z}\right)\boldsymbol{i}_x + \left(\frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x}\right)\boldsymbol{i}_y + \left(\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y}\right)\boldsymbol{i}_z \\ &= \frac{\partial (D_x \boldsymbol{i}_x + D_y \boldsymbol{i}_y + D_z \boldsymbol{i}_z)}{\partial t},\end{aligned} \tag{4.121}$$

$$\begin{aligned}\frac{\partial \boldsymbol{D}}{\partial t} &= \frac{\partial (D_x \boldsymbol{i}_x + D_y \boldsymbol{i}_y + D_z \boldsymbol{i}_z)}{\partial t} \\ &= \sigma (E_x \boldsymbol{i}_x + E_y \boldsymbol{i}_y + E_z \boldsymbol{i}_z) + \varepsilon_0 \varepsilon_r \frac{\partial (E_x \boldsymbol{i}_x + E_y \boldsymbol{i}_y + E_z \boldsymbol{i}_z)}{\partial t}.\end{aligned} \tag{4.122}$$

where $\boldsymbol{i}_x$, $\boldsymbol{i}_y$ and $\boldsymbol{i}_z$ are the unit vectors in $x$, $y$ and $z$ directions. Then Eqs. (4.120),

(4.121) and (4.122) are expressed in a scalar manner as

$$\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} = -\mu \frac{\partial H_x}{\partial t}, \tag{4.123}$$

$$\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} = -\mu \frac{\partial H_y}{\partial t}, \tag{4.124}$$

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} = -\mu \frac{\partial H_z}{\partial t}, \tag{4.125}$$

$$\frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} = \frac{\partial D_x}{\partial t}, \tag{4.126}$$

$$\frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} = \frac{\partial D_y}{\partial t}, \tag{4.127}$$

$$\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} = \frac{\partial D_z}{\partial t}, \tag{4.128}$$

$$\frac{\partial D_x}{\partial t} = \sigma E_x + \varepsilon_0 \varepsilon_r \frac{\partial E_x}{\partial t}, \tag{4.129}$$

$$\frac{\partial D_y}{\partial t} = \sigma E_y + \varepsilon_0 \varepsilon_r \frac{\partial E_y}{\partial t}, \tag{4.130}$$

$$\frac{\partial D_z}{\partial t} = \sigma E_z + \varepsilon_0 \varepsilon_r \frac{\partial E_z}{\partial t}. \tag{4.131}$$

In the following, we apply the semi-implicit version of an additive splitting approach to our equation.

### 4.3.6 Explicit Discretization

Here, the time and space derivatives are discretized by centred differences and the fields affected by the curl operators and staggered in time.

First, we discretize the conductivity term:

$$\frac{D_x^{n+1/2}(i,j,k) - D_x^{n-1/2}(i,j,k)}{\Delta t} = \sigma E_x^{n+1/2}(i,j,k) + \varepsilon_0 \varepsilon_r \frac{E_x^{n+1/2}(i,j,k) - E_x^{n-1/2}(i,j,k)}{\Delta t}, \tag{4.132}$$

$$\frac{D_y^{n+1/2}(i,j,k) - D_y^{n-1/2}(i,j,k)}{\Delta t} = \sigma E_y^{n+1/2}(i,j,k) + \varepsilon_0 \varepsilon_r \frac{E_y^{n+1/2}(i,j,k) - E_y^{n-1/2}(i,j,k)}{\Delta t}, \tag{4.133}$$

$$\frac{D_z^{n+3/2}(i,j,k) - D_z^{n+1/2}(i,j,k)}{\Delta t} = \sigma E_z^{n+1/2}(i,j,k) + \varepsilon_0 \varepsilon_r \frac{E_z^{n+1/2}(i,j,k) - E_z^{n-1/2}(i,j,k)}{\Delta t}. \tag{4.134}$$

Then we discretize the magnetic part:

$$\left\{\frac{H_z^n(i,j,k)-H_z^n(i,j-1,k)}{\Delta y}-\frac{H_y^n(i,j,k)-H_y^n(i,j,k-1)}{\Delta z}\right\} = \frac{D_x^{n+1/2}(i,j,k)-D_x^{n-1/2}(i,j,k)}{\Delta t}, \tag{4.135}$$

$$\left\{\frac{H_x^n(i,j,k)-H_x^n(i,j,k-1)}{\Delta z}-\frac{H_z^n(i,j,k)-H_z^n(i-1,j,k)}{\Delta x}\right\} = \frac{D_y^{n+1/2}(i,j,k)-D_y^{n-1/2}(i,j,k)}{\Delta t}, \tag{4.136}$$

$$\left\{\frac{H_y^n(i,j,k)-H_y^n(i-1,j,k)}{\Delta x}-\frac{H_x^n(i,j,k)-H_x^n(i,j-1,k)}{\Delta y}\right\} = \frac{D_z^{n+1/2}(i,j,k)-D_z^{n-1/2}(i,j,k)}{\Delta t}. \tag{4.137}$$

The last step is to discretize the electric part of the equation:

$$\left\{\frac{E_z^{n+1/2}(i,j+1,k)-E_z^{n+1/2}(i,j,k)}{\Delta y}-\frac{E_y^{n+1/2}(i,j,k+1)-E_y^{n+1/2}(i,j,k)}{\Delta z}\right\} = -\mu\frac{H_x^{n+1}(i,j,k)-H_x^n(i,j,k)}{\Delta t}, \tag{4.138}$$

$$\left\{\frac{E_x^{n+1/2}(i,j,k+1)-E_x^{n+1/2}(i,j,k)}{\Delta z}-\frac{E_z^{n+1/2}(i+1,j,k)-E_z^{n+1/2}(i,j,k)}{\Delta x}\right\} = -\mu\frac{H_y^{n+1}(i,j,k)-H_y^n(i,j,k)}{\Delta t}, \tag{4.139}$$

$$\left\{\frac{E_y^{n+1/2}(i+1,j,k)-E_y^{n+1/2}(i,j,k)}{\Delta x}-\frac{E_x^{n+1/2}(i,j+1,k)-E_x^{n+1/2}(i,j,k)}{\Delta y}\right\} = -\mu\frac{H_z^{n+1}(i,j,k)-H_z^n(i,j,k)}{\Delta t}, \tag{4.140}$$

*Remark 4.7* We follow forward stepping $H^n \to E^{n+1/2} \to H^{n+1}$. Based on the staggered grid, we can follow such a forward staggering in time and space.

Furthermore, the spatial parts of the equations can be splitted by applying the explicit AOS scheme.

### 4.3.7 Combination: Discretization and Splitting

In the following, we discuss the combination of discretization and splitting. For example, (4.138) is split into the $y$ direction part and the $z$ direction part.

Therefore, we can apply the additive operator splitting scheme, where we decompose the electric field into a $z$- and $y$-part.

We have

$$-\mu\frac{\partial H_x}{\partial t} = \frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z},\ H_x^n = H_x(t^n),\ \Delta t = t^{n+1} - t^n, \tag{4.141}$$

and split into the two steps

$$-\mu\frac{\partial H_x^1}{\partial t} = \frac{\partial E_z}{\partial y},\ H_x^{n,1} = H_x(t^n),\ \Delta t = t^{n+1} - t^n, \tag{4.142}$$

$$-\mu\frac{\partial H_x}{\partial t} = -\frac{\partial E_y}{\partial z},\ H_x^n = H_x^{n+1,1} = H_x^1(t^{n+1}),\ \Delta t = t^{n+1} - t^n, \tag{4.143}$$

where the initial condition of the second equation is coupled by the solution of the first equation, see also A–B splitting, see [53].

The discretized version of the two steps is given as

$$\begin{aligned}&\frac{E_z^{n+1/2}(i, J+1, K) - E_z^{n+1/2}(i, j, k)}{\Delta y}\\ &\quad = -\mu\frac{H_x^{n+1,1}(i, j, k) - H_x^{n,1}(i, j, k)}{\Delta t}\end{aligned} \tag{4.144}$$

where $H_x^{n,1}(i, j, k) = H_x^n(i, j, k)$, and the $z$ direction part is

$$\begin{aligned}&-\frac{E_y^{n+1/2}(i, j, k+1) - E_y^{n+1/2}(i, j, k)}{\Delta z})\\ &\quad = -\mu\frac{H_x^{n+1}(i, j, k) - H_x^{n+1,1}(i, j, k)}{\Delta t}.\end{aligned} \tag{4.145}$$

*Remark 4.8* Here, we have an explicit AOS splitting scheme combined with a FDTD method. The discretization scheme is based on the staggered grid idea, while the splitting method is an explicit version.

### 4.3.8 Practical Formulation of the 3D-AOS-FDTD Method

For more practical reasons, we formulate Eq. (4.99) as

$$B_i(c^n)c_i^{n+1} = c^n,\ i = 1, \ldots, m, \tag{4.146}$$

$$c^{n+1} = \sum_{i=1}^{m} c_i^{n+1}. \tag{4.147}$$

The Maxwell's equation is given as in Eqs. (4.117)–(4.119). Furthermore, the operator $\nabla\times$ is applied to the equations and we obtain Eqs. (4.120)–(4.122). The equations can be presented in the scalar notation, which is given as in Eqs. (4.123)–(4.131).

In the following, we apply the additive splitting approach to our magnetic field equation, which are derived in the AOS scheme as follows:

- For the scalar field $H_x$, we have

$$\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} = -\mu\frac{\partial H_x}{\partial t}, \tag{4.148}$$

and the AOS scheme is given as

$$\frac{\partial {H_x}^*}{\partial t} = -\frac{1}{\mu}\frac{\partial E_z}{\partial y},\ {H_x}^*(t^n) = H_x(t^n), \tag{4.149}$$

$$\frac{\partial {H_x}^{**}}{\partial t} = \frac{1}{\mu}\frac{\partial E_y}{\partial z},\ {H_x}^{**}(t^n) = {H_x}^*(t^{n+1}), \tag{4.150}$$

where $H_x(t^{n+1}) = {H_x}^{**}(t^{n+1})$.

- For the scalar field $H_y$, we have

$$\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} = -\mu\frac{\partial H_y}{\partial t}, \tag{4.151}$$

and the AOS scheme is given as

$$\frac{\partial H_y{}^*}{\partial t} = -\frac{1}{\mu}\frac{\partial E_x}{\partial y},\ H_y{}^*(t^n) = H_y(t^n), \tag{4.152}$$

$$\frac{\partial H_y{}^{**}}{\partial t} = \frac{1}{\mu}\frac{\partial E_z}{\partial z},\ H_y{}^{**}(t^n) = H_y{}^*(t^{n+1}), \tag{4.153}$$

where $H_y(t^{n+1}) = H_y{}^{**}(t^{n+1})$.

- For the scalar field $H_z$, we have

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} = -\mu\frac{\partial H_z}{\partial t}, \tag{4.154}$$

and the AOS scheme is given as

$$\frac{\partial H_z{}^*}{\partial t} = -\frac{1}{\mu}\frac{\partial E_y}{\partial y},\ H_z{}^*(t^n) = H_z(t^n), \tag{4.155}$$

$$\frac{\partial H_x{}^{**}}{\partial t} = \frac{1}{\mu}\frac{\partial E_x}{\partial z},\ H_z{}^{**}(t^n) = H_z{}^*(t^{n+1}), \tag{4.156}$$

where $H_z(t^{n+1}) = H_z{}^{**}(t^{n+1})$.

### 4.3.9 Discretization of the Equations with the AOS

Here, the time and space derivatives are discretized by centred differences and the fields affected by the curl operators are averaged in time. We apply $\theta$-schemes, i.e. the combination of an explicit and implicit time discretization, and can apply such a scheme to the AOS.

For example, we apply AOS Eqs. (4.149)–(4.150) and we have

$$\theta\frac{E_z^{n+1}(i,j+1,k) - E_z^{n+1}(i,j,k)}{\Delta y} + (1-\theta)\frac{E_z^{n}(i,j+1,k) - E_z^{n}(i,j,k)}{\Delta y} = -\mu\frac{H_x^{*,n+1}(i,j,k) - H_x^{n}(i,j,k)}{\Delta t}, \tag{4.157}$$

$$-\theta\frac{E_y^{n+1}(i,j,k+1) - E_y^{n+1}(i,j,k)}{\Delta z} - (1-\theta)\frac{E_y^{n}(i,j,k+1) - E_y^{n}(i,j,k)}{\Delta z} = -\mu\frac{H_x^{**,n+1}(i,j,k) - H_x^{*,n+1}(i,j,k)}{\Delta t}, \tag{4.158}$$

where the results are given as $H_x^{n+1}(i,j,k) = H_x^{**,n+1}(i,j,k)$.

For all Eqs. (4.123)–(4.131) applied to the AOS and the $\theta$-scheme, we have the discretized equations as

$$\theta\left\{\frac{E_z^{n+1}(i,j+1,k)-E_z^{n+1}(i,j,k)}{\Delta y}-\frac{E_y^{n+1}(i,j,k+1)-E_y^{n+1}(i,j,k)}{\Delta z}\right\}$$
$$+(1-\theta)\left\{\frac{E_z^{n}(i,j+1,k)-E_z^{n}(i,j,k)}{\Delta y}-\frac{E_y^{n}(i,j,k+1)-E_y^{n}(i,j,k)}{\Delta z}\right\}$$
$$=-\mu\frac{H_x^{n+1}(i,j,k)-H_x^{n}(i,j,k)}{\Delta t}, \tag{4.159}$$

$$\theta\left\{\frac{E_x^{n+1}(i,j,k+1)-E_x^{n+1}(i,j,k)}{\Delta z}-\frac{E_z^{n+1}(i+1,j,k)-E_z^{n+1}(i,j,k)}{\Delta x}\right\}$$
$$+(1-\theta)\left\{\frac{E_x^{n}(i,j,k+1)-E_x^{n}(i,j,k)}{\Delta z}-\frac{E_z^{n}(i+1,j,k)-E_z^{n}(i,j,k)}{\Delta x}\right\}$$
$$=-\mu\frac{H_y^{n+1}(i,j,k)-H_y^{n}(i,j,k)}{\Delta t}, \tag{4.160}$$

$$\theta\left\{\frac{E_y^{n+1}(i+1,j,k)-E_y^{n+1}(i,j,k)}{\Delta x}-\frac{E_x^{n+1}(i,j+1,k)-E_x^{n+1}(i,j,k)}{\Delta y}\right\}$$
$$+(1-\theta)\left\{\frac{E_y^{n}(i+1,j,k)-E_y^{n}(i,j,k)}{\Delta x}-\frac{E_x^{n}(i,j+1,k)-E_x^{n}(i,j,k)}{\Delta y}\right\}$$
$$=-\mu\frac{H_z^{n+1}(i,j,k)-H_z^{n}(i,j,k)}{\Delta t}, \tag{4.161}$$

$$\theta\left\{\frac{H_z^{n+1}(i,j,k)-H_z^{n+1}(i,j-1,k)}{\Delta y}-\frac{H_y^{n+1}(i,j,k)-H_y^{n+1}(i,j,k-1)}{\Delta z}\right\}$$
$$+(1-\theta)\left\{\frac{H_z^{n}(i,j,k)-H_z^{n}(i,j-1,k)}{\Delta y}-\frac{H_y^{n}(i,j,k)-H_y^{n}(i,j,k-1)}{\Delta z}\right\}$$
$$=\frac{D_x^{n+1}(i,j,k)-D_x^{n}(i,j,k)}{\Delta t}, \tag{4.162}$$

$$\theta\left\{\frac{H_x^{n+1}(i,j,k)-H_x^{n+1}(i,j,k-1)}{\Delta z}-\frac{H_z^{n+1}(i,j,k)-H_z^{n+1}(i-1,j,k)}{\Delta x}\right\}$$
$$+(1-\theta)\left\{\frac{H_x^{n}(i,j,k)-H_x^{n}(i,j,k-1)}{\Delta z}-\frac{H_z^{n}(i,j,k)-H_z^{n}(i-1,j,k)}{\Delta x}\right\}$$
$$=\frac{D_y^{n+1}(i,j,k)-D_y^{n}(i,j,k)}{\Delta t}, \tag{4.163}$$

$$\theta\left\{\frac{H_y^{n+1}(i,j,k)-H_y^{n+1}(i-1,j,k)}{\Delta x}-\frac{H_x^{n+1}(i,j,k)-H_x^{n+1}(i,j-1,k)}{\Delta y}\right\}$$
$$+(1-\theta)\left\{\frac{H_y^{n}(i,j,k)-H_y^{n}(i-1,j,k)}{\Delta x}-\frac{H_x^{n}(i,j,k)-H_x^{n}(i,j-1,k)}{\Delta y}\right\}$$
$$=\frac{D_z^{n+1}(i,j,k)-D_z^{n}(i,j,k)}{\Delta t}, \tag{4.164}$$

where $\theta=[0,1]$.

*Remark 4.9* If we apply the conductivity as an operator, we have taken into account the averaging of the electric field $\boldsymbol{E}$ term. Such an idea is done by $\theta$ method and afterwards, we can apply the additive operator splitting.

Further, we discretize Eqs. (4.129), (4.130) and (4.131), while the conductivity term and $\boldsymbol{E}$ term are averaged in time.

We apply then

$$\frac{D_x^{n+1}(i,j,k)-D_x^{n}(i,j,k)}{\Delta t}$$
$$=\sigma(\theta E_x^{n+1}(i,j,k)+(1-\theta)E_x^{n}(i,j,k))+\varepsilon_0\varepsilon_r\frac{E_x^{n+1}(i,j,k)-E_x^{n}(i,j,k)}{\Delta t}, \tag{4.165}$$
$$\frac{D_y^{n+1}(i,j,k)-D_y^{n}(i,j,k)}{\Delta t}$$
$$=\sigma(\theta E_y^{n+1}(i,j,k)+(1-\theta)E_y^{n}(i,j,k))+\varepsilon_0\varepsilon_r\frac{E_y^{n+1}(i,j,k)-E_y^{n}(i,j,k)}{\Delta t}, \tag{4.166}$$
$$\frac{D_z^{n+1}(i,j,k)-D_z^{n}(i,j,k)}{\Delta t}$$
$$=\sigma(\theta E_z^{n+1}(i,j,k)+(1-\theta)E_z^{n}(i,j,k))+\varepsilon_0\varepsilon_r\frac{E_z^{n+1}(i,j,k)-E_z^{n}(i,j,k)}{\Delta t}, \tag{4.167}$$

where $\theta\in[0,1]$, i.e. $\theta=1$ is implicit, $\theta=0$ is explicit and $\theta=1/2$ is semi-implicit

Then, (4.159)–(4.170) were split into the three direction parts.

For the pure implicit version, which conform with the AOS method, we have $\theta=1$ and obtain

$$\frac{D_x^{n+1}(i,j,k) - D_x^n(i,j,k)}{\Delta t} = \sigma E_x^{n+1}(i,j,k) + \varepsilon_0 \varepsilon_r \frac{E_x^{n+1}(i,j,k) - E_x^n(i,j,k)}{\Delta t}, \quad (4.168)$$

$$\frac{D_y^{n+1}(i,j,k) - D_y^n(i,j,k)}{\Delta t} = \sigma E_y^{n+1}(i,j,k) + \varepsilon_0 \varepsilon_r \frac{E_y^{n+1}(i,j,k) - E_y^n(i,j,k)}{\Delta t}, \quad (4.169)$$

$$\frac{D_z^{n+1}(i,j,k) - D_z^n(i,j,k)}{\Delta t} = \sigma E_z^{n+1}(i,j,k) + \varepsilon_0 \varepsilon_r \frac{E_z^{n+1}(i,j,k) - E_z^n(i,j,k)}{\Delta t}. \quad (4.170)$$

Also this part can be splitted by applying the AOS scheme for the two operators.

*Remark 4.10* At least, the AOS scheme is flexible and we could extend to the implicit version, i.e. $\theta = 1$. Here, we have to deal additional with inversion of the underlying equation system, which is more delicate, but we can skip the CFL conditions as time step conditions. Further additional steps are necessary and are computed implicitly.

### 4.3.10 Transport Equation Coupled with an Electro-magnetic Field Equations

The following example is discussed in [3], and concluded some of the important multiscale results.

We deal with the two-dimensional advection–diffusion equation and electric field equation:

$$\partial_t u = -v_x(E_z(x,y))\frac{\partial u}{\partial x} - v_y \frac{\partial u}{\partial y} + D\frac{\partial^2 u}{\partial x^2} + D\frac{\partial^2 u}{\partial y^2}, \quad (4.171)$$

$$(x,y,t) \in \Omega \times (0,T),$$

$$u(x,y,t_0) = u_0(x,y), \quad (4.172)$$

$$\frac{\partial H_x(x,y)}{\partial t} = -\frac{\partial E_z}{\partial y}, \ (x,y,t) \in \Omega \times (0,T), \quad (4.173)$$

$$\frac{\partial H_y(x,y)}{\partial t} = \frac{\partial E_z}{\partial x}, \ (x,y,t) \in \Omega \times (0,T), \quad (4.174)$$

$$\frac{\partial E_z(x,y)}{\partial t} = \frac{1}{\varepsilon}\left(\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y}\right) - J_{source}, \ (x,y,t) \in \Omega \times (0,T), \quad (4.175)$$

where we have the initial function:

$$u(\mathbf{x}, t_0) = u_a(\mathbf{x}, t_0) = \frac{1}{t_0}\exp\left(-\frac{(\mathbf{x} - \mathbf{v}t_0)^2}{4Dt_0}\right),$$

where $\mathbf{x} = (x,y)^t$ and $\mathbf{v} = (v_x, v_y)^t$, and we have

$$\begin{cases} v_x(E_z(x,y)) = 1, v_y = 1.0, & \text{for } t \in (0, t_0), \\ v_x(E_z(x,y)) = \alpha E_z(x,y), v_y = 0.0, & \text{for } t \geq t_0, \end{cases} \tag{4.176}$$

with $\alpha = 0.001$, $t_0 = 10.0$. The spatial domain is given as $\Omega = [0,1] \times [0,1]$.

The electric field $E_z(x,y)$ has the following line source:

$J_{source}(x,y) = \sin(t)$ where $x = 0$, $y \in (0, 100)$.

The control of the particle transport is given by the electric field shown in Fig. 4.7.

In the following, we have the line sources with the results given in Fig. 4.8:

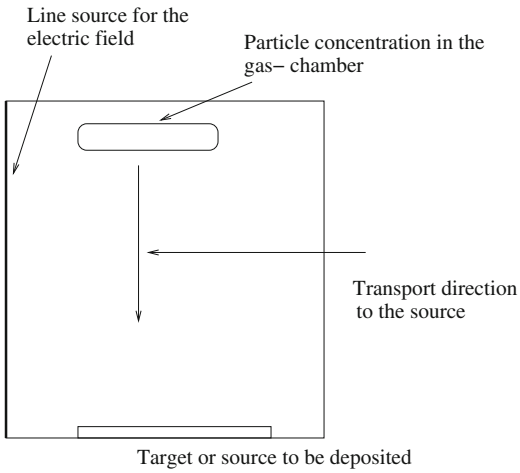



**Fig. 4.7** Electric field in the apparatus

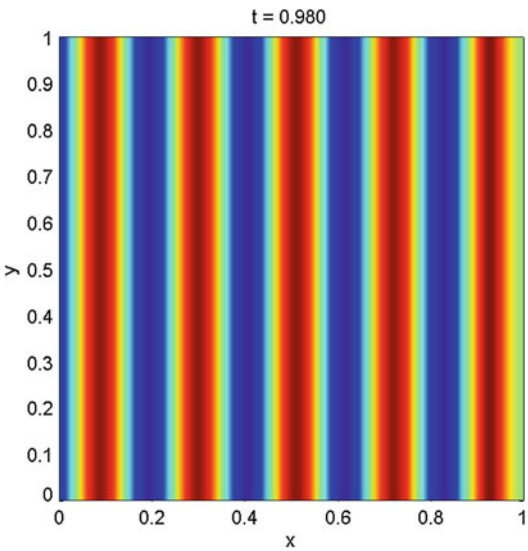



**Fig. 4.8** Line source of the electric field in the apparatus

Numerically, we solve the equation, as in the following explicit AOS Algorithm 4.3:

**Algorithm 4.3** We have coupled the equations by the following algorithm:
(1) Initialize convection–diffusion equation, till $t_0$.
(2) Solve the electric field equation with $t_{start}$ and obtain $E_z(x, y)$ for $t_0$
(3) Solve convection–diffusion equation with $t_0 + \Delta t$ and use $E_z(x, y)$ for $t_{start}$ for the unknown.
(4) Do $t_0 = t_0 + \Delta t$ and go to (2) till $t_0 = t_{end}$

The following Figs. 4.9 and 4.10 show the developing concentration under the influence of the electric field, where $\alpha = 0.07$, $t_{start} = 0.5$ and $v_y = 0$ for $t \geq t_{start}$.

*Remark 4.11* For spatial and time discretization, it is important to balance such schemes. If we apply an explicit AOS method and assume to have finite difference schemes in time and space, we have taken into account the CFL (Courant-Friedrichs-Levy) condition.

The condition for the explicit scheme is given as

$$\sqrt{\varepsilon}\Delta x \geq \Delta t, \tag{4.177}$$

where $\Delta x$ and $\Delta t$ are the spatial and time steps.

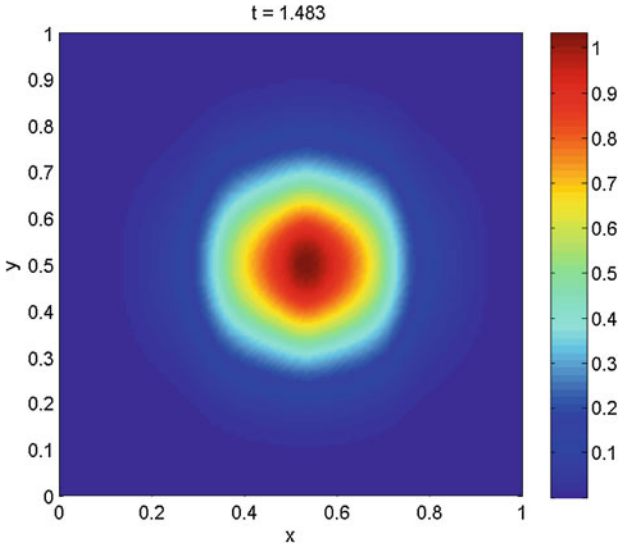



**Fig. 4.9** Concentration density of the plasma specie, influenced by the electromagnetic field, in the apparatus at time $t = 1.483$ (the concentration flows from the left lower corner to the center)

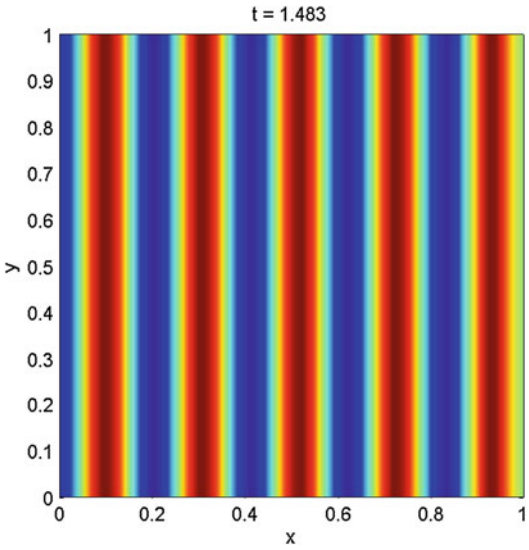



**Fig. 4.10** Electric field in the apparatus at time $t = 1.483$

*Remark 4.12* Another idea is based on the following implicit AOS Algorithm 4.4, while Eqs. (4.171)–(4.175) are discretized as

$$u^{n+1}(i,j) = u^n(i,j) + \Delta t\Big(-v_x(E_z^{n+1}(i,j))\frac{u^(n)(i+1,j)-u^n(i,j)}{\Delta x} - v_y\frac{u^n(i,j+1)-u^n(i,j)}{\Delta y} + D\frac{u^n(i+1,j)-2u^n(i,j)+u^n(i-1,j)}{\Delta x^2} + D\frac{u^n(i,j+1)-2u^n(i,j)+u^n(i,j-1)}{\Delta y^2}\Big), \tag{4.178}$$

$$\frac{H_x^{n+1}(i,j)-H_x^n(i,j)}{\Delta t} = -\frac{E_z^{n+1}(i,j+1)-E_z^{n+1}(i,j)}{\Delta y}, \tag{4.179}$$

$$\frac{H_y^{n+1}(i,j)-H_y^n(i,j)}{\Delta t} = \frac{E_z^{n+1}(i+1,j)-E_z^{n+1}(i,j)}{\Delta x}, \tag{4.180}$$

$$\frac{E_z^{n+1,*}(i,j)-E_z^n(i,j)}{\Delta t} = \frac{1}{\varepsilon}\Big(\frac{H_y^{n+1}(i+1,j)-H_y^{n+1}(i,j)}{\Delta x}\Big) - 0.5J_{source}(i,j), \tag{4.181}$$

$$\frac{E_z^{n+1}(i,j)-E_z^{n+1,*}(i,j)}{\Delta t} = \frac{1}{\varepsilon}\Big(-\frac{H_x^{n+1}(i,j+1)-H_x^{n+1}(i,j)}{\Delta y}\Big) - 0.5J_{source}(i,j), \tag{4.182}$$

where $i, j = 1, \ldots, I$ are the spatial discretization points with $\Delta x$, and $\Delta y$ are the spatial steps. Furthermore, $\Delta t$ is the time step with $n = 0, 1, \ldots, N$, which are the time points.

Then the equation system is given as

$$\mathscr{U}^{n+1} = (I - \Delta t A)^{-1}\mathscr{U}^n, \tag{4.183}$$

$$U^{n+1} = U^n + \Delta t B(v_x(\mathscr{E}_z^n), v_y, D)U^n, \tag{4.184}$$

where $U^{n+1} = (u^{n+1}(1,1), \ldots, u^{n+1}(I,I))^t$ is the discretized solution of the transport system, $\mathscr{U}^{n+1} = (\mathscr{H}_x^{n+1}, \mathscr{H}_y^{n+1}, \mathscr{E}_z^{n+1})^t$ is the discretized solution of the electro-magnetic field, with $\mathscr{H}_x^{n+1} = (H_x^{n+1}(1,1), \ldots, H_x^{n+1}(I,I))^t$, $\mathscr{H}_y^{n+1} = (H_y^{n+1}(1,1), \ldots, H_y^{n+1}(I,I))^t$ and $\mathscr{E}_z^{n+1} = (E_z^{n+1}(1,1), \ldots, E_z^{n+1}(I,I))^t$. Furthermore, the matrices $A \in \mathbb{R}^{I\times I}$ and $B \in \mathbb{R}^{3I\times 3I}$ are given and have embedded the boundary conditions.

The algorithmic idea 4.4 is given as follows.

**Algorithm 4.4** We have coupled the equations by the following algorithm:
(1) Initialize convection–diffusion equation, till $t_0$ and $n = 0$.
(2) Solve implicitly the electro-magnetic field equation with the time step $\Delta t$ and obtain $\mathscr{E}_z^{n+1}$ for $t_{n+1}$.
(3) Solve explicitly the convection–diffusion equation with $\Delta t$ and use $\mathscr{E}_z^{n+1}$ for $t_{n+1}$ for the unknown and obtain $U^{n+1}$.
(4) Do $t_{n+1} = t_n + \Delta t$ and go to (2) till $t_{n+1} = t_{end}$.

Here, we have the benefit that we are not restricted to the time step of the electro-magnetic field and we could apply the large time step, which is also applied for the convection–diffusion equation.

## 4.4 Extensions of Particle in Cell Methods for Nonuniform Grids: Multiscale Ideas and Algorithms

**Abstract** In this section, we discuss ideas to extend uniform particle in cell (PIC) method to nonuniform PIC methods. The ideas are based to modify the so-called PIC cycle parts, which decouple grid-free (particle methods), grid-based (field methods) and couple the parts with interpolation methods. The methodological idea of the PIC method can be seen as a multiscale method, while we deal with different underlying modelling scales, e.g. micro- and macroscopic scale. The different parts of the PIC method can be applied in different scales, e.g. a microscale (particle solver) and a macroscale (field solver). So we have a multiscale behaviour, while the transfer between the micro- and macro-model is done via interpolation or restriction, which is applied in the PIC method as spline approximations, see [54]. Another aspect results of the physical constraints mean that we have to fulfil the mass, momentum and energy conservation of the problem, see [54]. Such a problem can only be fulfilled for a uniform grid steps, while we deal with a primary grid. A modification to an adaptive or nonuniform grid needs to extend the freedom degrees of the underlying

grid, and therefore we have to deal with an additional so-called dual grid. On dual grid or in the logical space, we can extend the uniform grid into an adaptive grid and such a modification allows us to conserve the constraints, e.g. mass, momentum and energy conservation, see also [55, 56]. Both interpolation schemes (particle to grid and grid to particle) and solver methods (macrosolver: Poisson solver and microsolver: time integrator) have to be combined such that the physical constraints are fulfilled and the numerical errors are at least second order, see [57]. Here, we discuss the ideas to develop step-by-step multiscale extension of the PIC cycle. We modify shape function to adaptive shape functions and fit them to the adaptive discretization schemes such that the interpolations are of the same order as in the uniform case. Furthermore, we present some extensions to 2D and deal with simple 1D examples.

### *4.4.1 Introduction of the Problem*

The motivation of the modification arose of a practical application in a propulsion problem. While in the inner or ion thruster part we deal with high density of particle and the outer or plume region, it has only a very low density of particles, see [58].

If we apply uniform PIC methods, we have taken into one spatial step for the full region, i.e. the very small spatial step of the inner region, and we have the problem of very long computational times.

In Fig. 4.11, we present the different spatial scales of the motivation.

*Remark 4.13* The multiscale problem is given by the restriction of the time and spatial steps for a fine resolution of the inner part (restriction by the Debye length $\lambda_D$, where $\Delta x \leq \lambda_D$ and $\Delta x$ is the spatial step size of the uniform grid. The Debye length is the distance scale over which significant charge densities can spontaneously exist,

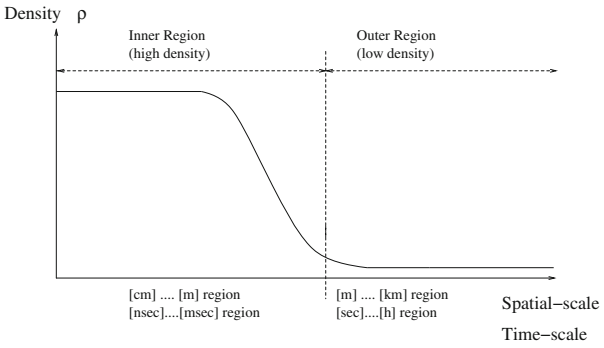



**Fig. 4.11** The model problem with *inner* and *outer* region of different spatial and time scales

see [30]. It is therefore the largest scale, which can be resolved by the PIC method, see [54]. Moreover, if we deal with the multiscale problem of the test problem, we have to obtain very small spatial step sizes.

### *4.4.2 Introduction of the Extended Particle in Cell Method*

The Particle in Cell (PIC) method is the well-known method over the last decades. The concept of coupling grid and grid-free methods are applied to accelerate the solver process. While parts of the equations are solved on a grid, e.g. Poisson equation, the transport of particles is done grid-free by computing the trajectories with fast time integrators, see [54, 59].

In recent applications, the flexibility of PIC schemes, with respect to higher order schemes and nonuniform grids, is important (Fig. 4.12).

In the following, we discussed a possible flexibilization of the PIC cycles based on improving all parts of the cycle, see Fig. 4.2.

The following three parts of the PIC can be improved:

- Shape function (higher order spline functions, which fulfil the constraints, e.g. TSC or higher, see [54]).

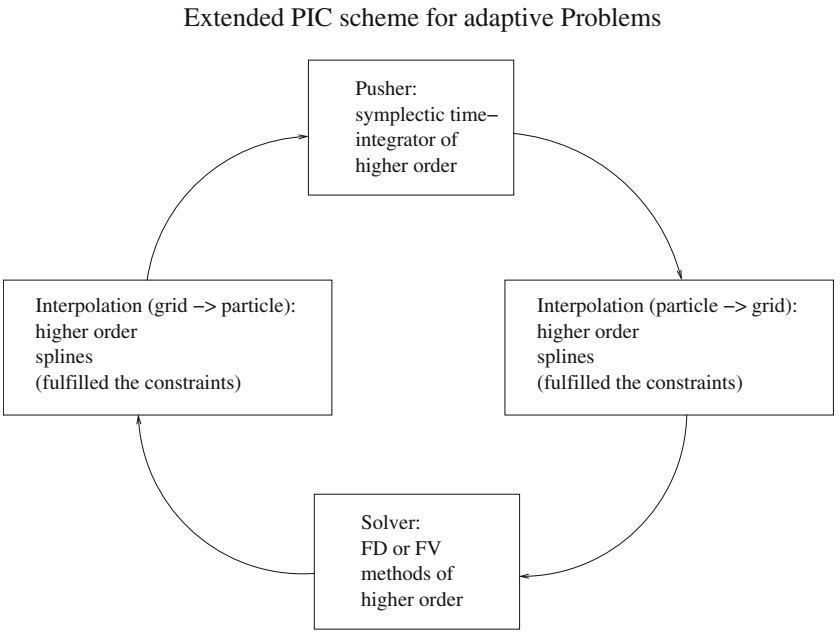



**Fig. 4.12** Improved PIC cycles for adaptive PIC

- Solver (higher order discretization schemes, e.g. fourth-order finite difference schemes, see [60]).
- Pusher (higher order symplectic time integrators, e.g. fourth-order symplectic schemes, see [61]).

*Remark 4.14* Before improving one part of the PIC cycle, we have to be careful to fulfil the physical constraints of the problem, such that it might be possible, which we have to update all the parts of the PIC cycle for such an extension, see the discussion of an adaptive PIC code in [62].

### *4.4.3 Mathematical Model*

In the following, we discuss the mathematical model, which is based on the Vlasov–Poisson equation, which describe an ideal plasma model.

The Vlasov equation describes the electron distribution $f$

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} + \frac{\mathbf{F}}{m} \cdot \frac{\partial f}{\partial \mathbf{v}} = 0, \tag{4.185}$$

and the Poisson equation describes the potential to the electrons in the electric field $\mathbf{E}$:

$$\nabla^2 \phi = -\frac{\rho}{\varepsilon}, \tag{4.186}$$

$$\mathbf{F} = q\mathbf{E} = -q\nabla\phi, \tag{4.187}$$

The positive ions are used as a fixed, neutralizing, background charge density $\rho_0$ and the total charge density $\rho$ is given as

$$\rho(\mathbf{x}) = q \int f \, d\mathbf{v} + \rho_0, \tag{4.188}$$

We apply the following assumption to the model in the linear case:

- Plasma frequency: $\omega_P = \sqrt{\frac{nq^2}{\varepsilon_0 m_e}}$ .
- Debye length: $\lambda_D = \sqrt{\frac{\varepsilon_0 k_B T}{nq^2}}$ .

These lengths are important for the explicit numerical schemes, i.e. we have restrictions of the time and spatial step sizes:

- Time step size $\Delta t \ll \frac{2}{\omega_P}$ .
- Spatial step size: $\Delta x \leq \lambda_D$ .

Furthermore, we have some more conditions:

- Restriction to the length of apparatus $L$: $\lambda_D \ll L$ .
- Number of particle: $N_P \lambda_D \gg L$,

such that we have a sufficient large length of the test apparatus and also to fulfil the number of particle per Debye length which is sufficiently large, where we have from the statistical point of view sufficient dates for the methods, see [54].

### *4.4.4 Discretization of the Model*

To compute the model, we have to apply the idea of a super particle, which allows to decouple into an equation of motion (transport of the particles) and the potential equation (forces to the particles).

We assume that the $x-v$ phase space is divided into a regular array of infinitesimal cells of volume $d\tau = dxdv$, where $d\tau$ is sufficiently small so that only one electron is in it. Then $f(x, v, t)d\tau$ gives the probability that the cell at $(x, v)$ is occupied at time $t$. We assume that the electron is then shifted to time $t'$ to the cell $(x', v')$. Due to this assumption, it is also used in the characteristics schemes, see [63].

We have to solve the equation of motions:

$$\frac{dx}{dt} = v, \tag{4.189}$$

$$\frac{dv}{dt} = \frac{qE}{m}, \tag{4.190}$$

and in the time integral form

$$x' = x + \int_t^{t'} v dt, \tag{4.191}$$

$$v' = v + \int_t^{t'} \frac{qE}{m} dt, \tag{4.192}$$

and we can show in general for such a shift: $f(x', v', t') = f(x, v, t)$.

To speed up the computations, we take a sample of points (super particle) $\{x_i, v_i, i = 1, \ldots, N_s\}$ and an element $i$ of the phase fluid is corresponding to

$$N_s = \int_i f \; dxdv, \tag{4.193}$$

The characteristics to the phase space of the super particle points are given by

$$\frac{dx_i}{dt} = v_i, \tag{4.194}$$

$$\frac{dv_i}{dt} = \frac{F(x_i)}{M}, \tag{4.195}$$

$M = N_s m_e$ and $m_e$ is the electron mass.

In the following, we discuss the different extensions to the adaptive PIC methods.

#### 4.4.4.1 1D Adaptive PIC

To understand the parts of the adaptive PIC method, we discuss in the first steps the one-dimensional case. In the following, we describe the different tools for the 1D adaptive PIC:

- 1 D adaptive finite difference (FD) method,
- 1 D adaptive Shape function, and
- Fitting scheme at the interface.

While the 1D FD methods are applied to the micro- and macro-model, we apply also adaptive interpolation/restriction methods, i.e. shape functions, to apply the data transfer between the different scales. Furthermore, we have to deal with a fitting scheme at the interface to fulfil the constraints of the scheme, e.g. conserve the first moments of the shape functions, see [62].

**1D Adaptive Finite Difference Discretization for the Poisson Equation**

In the following, we have the adaptive scheme, which are based on weighting the central difference scheme for the underlying model problem, i.e. here the Poisson's equation.

We discuss the adaptive grid of finite difference schemes, see [64], for the Poisson equation in one dimension:

$$\frac{d^2\phi}{dx^2} = -\frac{1}{\varepsilon_0}\rho(x_i), \;\; x_i \in [0, L], \tag{4.196}$$

$$\phi(0) = 0, \phi(L) = 0, \tag{4.197}$$

where $x_i$ give the coordinates of a super particle $i$.

The finite difference scheme after Shortly and Weller [65], which is given with a three-point stencil, see also [66], and the difference quotient are given as

$$-D_{\Delta x}^2\phi = \frac{2}{\Delta x^2}\left[\frac{1}{s_r(s_r+s_l)}\phi(x+s_r\Delta x) + \frac{1}{s_l(s_r+s_l)}\phi(x-s_l\Delta x) - \frac{1}{s_r s_l}\phi(x)\right], \tag{4.198}$$

where $\Delta x$ is the mesh size of the grid and $s_r, s_l \in (0, 1]$ are the scaled factors of the finer grid. Furthermore, $D_{\Delta x}^2 = \partial_{s_r\Delta x}^+ \partial_{s_l\Delta x}^-$ is the difference quotient with

$$\partial_{s_l\Delta x}^+\phi = \frac{\phi(x)-\phi(x-s_l\Delta x)}{s_l\Delta x}, \tag{4.199}$$

$$\partial_{s_r\Delta x}^-\phi = \frac{\phi(x+s_r\Delta x)-\phi(x)}{s_r\Delta x}. \tag{4.200}$$

The consistency error is given for the boundary points also as a second-order method, see [66]:

$$||\phi(x)-\phi_{\Delta x}(x)|| \leq \Delta x^2\left(\frac{1}{48}d^2||\phi||_{C^{3,1}(\overline{\Omega})} + \frac{2}{3}||\phi||_{C^{2,1}(\overline{\Omega})}\right), \tag{4.201}$$

where $d \leq \Delta x$.

*Remark 4.15* For a different notation, we apply

$$\begin{aligned} D_{\Delta x}D_{\Delta\tilde{x}}\phi \; &= -\frac{2}{\Delta x(\Delta x \; + \; \Delta\tilde{x})}\phi(x+\Delta x) + \frac{2}{\Delta x \; \Delta\tilde{x}}\phi(x) \\ &\quad -\frac{2}{\Delta\tilde{x}(\Delta x \; + \; \Delta\tilde{x})}\phi(x-\Delta\tilde{x}), \end{aligned} \tag{4.202}$$

while $\Delta x$ is the grid size on the left-hand side and $\Delta\tilde{x}$ is the grid size on the right-hand side.

### Adaptive Shape Functions

In the following, we derive adaptive higher order shape functions.

We have the following underlying steps for the construction:

1. 1D Interpolation and Shape functions.
2. 1D uniform shape functions.
3. Adaptive Linear Splines (adaptive CIC).
4. Construction of higher order Splines.

The steps are discussed in the following outlined points.

1. *1D Interpolation and Shape functions*:

In the following, we discuss the shape functions that are need to map the charge densities on a grid.

We deal with CIC (Cloud in Cell) shape functions, see [54], which are the linear shape functions $S(x_i - X_j)$, where $X_j$ implements the grid point and $x_i$ is the position of the particle $i$.
The density at the grid point of the particles is weighted by the weighting function:

$$\rho_j = \sum_{i=1}^{N} q_i S(x_i - X_j), \tag{4.203}$$

where $q_i$ is the $i$th charge.

In standard application, this function is symmetry and fulfils the isotropy of space, charge conservation and condition to avoid self forces, see [54, 67].

For the consistency of the uniform and nonuniform shape functions, we have the following restriction:

$$\sum_{i=1}^{N} S(x - X_i) = 1, \tag{4.204}$$

where all the weights are $q_i = 1$ and $x$ is the position of the particle and $X_i$ the grid point at position $i$.

In the following, we see the construction on a non-symmetric mesh, see Fig. 4.13.

2. *1D uniform shape function*

We deal with the following uniform shape functions:

- NGP: nearest grid point and
- CIC: Cloud in Cell.

The NGP uniform shape functions is given as

$$S(x - X) = \begin{cases} 1, & \text{when } |x - X| \leq \frac{\Delta x}{2}, \\ 0, & \text{else}, \end{cases} \tag{4.205}$$

where we have a uniform grid size of $\Delta x$ in the domain $\Omega = [0, L]$.

The CIC uniform shape functions is given as

$$S(x - X) = \begin{cases} 1 - \frac{|x-X|}{\Delta x}, & \text{when } |x - X| < \Delta x, \\ 0, & \text{else}, \end{cases} \tag{4.206}$$

where we have a uniform grid size of $\Delta x$ in the domain $\Omega = [0, L]$.

For the uniform mesh function, we have to fulfil the consistency (mass conservation) (4.204).

**Theorem 4.5** *For the uniform shape function (4.206), we fulfil the consistency (4.204).*

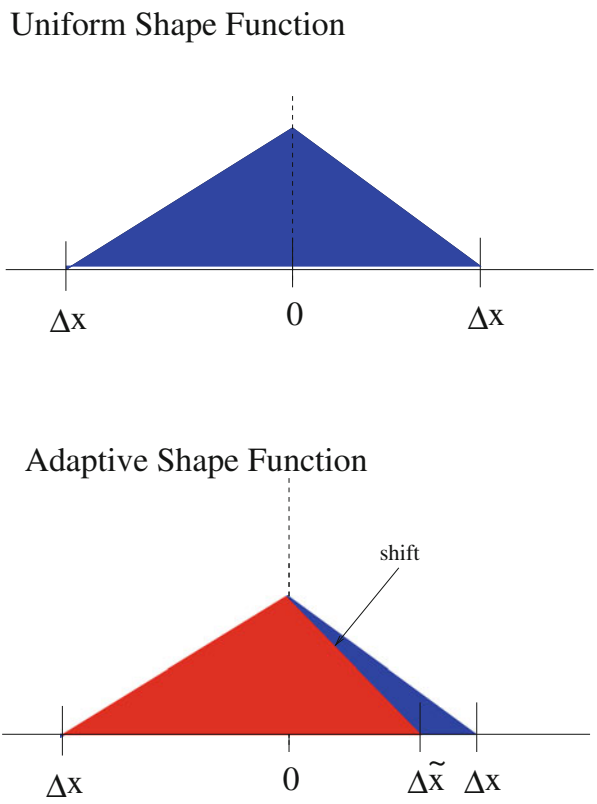



**Fig. 4.13** Adaptive shape function

*Proof* It is sufficient to proof the function for the following situation for one particle $x$ and the two grid points $X$ and $X - \Delta x$, based on the symmetry, and one can do it for all particles:

$$1 - \frac{x - (X - \Delta x)}{\Delta x} + 1 - \frac{(X - x)}{\Delta x} = 1, \tag{4.207}$$

$$2 - \frac{x}{\Delta x} - \frac{X}{\Delta x} - 1 - \frac{X}{\Delta x} + \frac{x}{\Delta x} = 1, \tag{4.208}$$

this is fulfilled.

3. *Adaptive Linear Splines (adaptive CIC)*

In the following, we discuss the adaptive shape functions.

We assume the domain $\Omega = [0, L]$ and $\Delta x$ is operating in the domain $[0, L_1]$, while $\Delta\tilde{x}$ is operating in the domain $[L_1, L]$.

The grid point $X$ is not at the boundary $L_1$, i.e. $x < L_1 - \Delta x$ or $x > L_1 + \Delta\tilde{x}$:

$$S(x-X)=\begin{cases}1-\frac{|x-X|}{\Delta x}, & \text{when } |x-X|<\Delta x, \text{ and } x\in[0,L_1],\\ 1-\frac{|x-X|}{\Delta \tilde{x}}, & \text{when } |x-X|<\Delta \tilde{x}, \text{ and } x\in[L_1,L],\\ 0, & \text{else},\end{cases} \tag{4.209}$$

where we assume to have a nonuniform grid size, while of $\Delta x$ is the domain $\Omega = [0, L_1]$ and $\Delta\tilde{x}$ is the domain $\Omega = [L_1, L]$.

For the nonuniform mesh function, we have to fulfil the consistency (mass conservation) (4.204).

**Theorem 4.6** *For the nonuniform shape function (4.285), we fulfil the consistency (4.204).*

*Proof* It is sufficient to prove that the shape functions based on each different domain fulfil the condition.

For domain $\Omega_1 = [0, L_1]$, we have

$$1-\frac{x-(X-\Delta x)}{\Delta x}+1-\frac{(X-x)}{\Delta x}=1, \tag{4.210}$$

and when it is fulfilled also for domain $\Omega_2 = [L_1, L]$, we have

$$1-\frac{x-(X-\Delta \tilde{x})}{\Delta \tilde{x}}+1-\frac{(X-x)}{\Delta \tilde{x}}=1. \tag{4.211}$$

*Remark 4.16* The idea of the adaptive shape functions can also be extended to higher order shape functions, e.g. [54]. An example is given in the appendix.

4. *Construction of the higher order Spline*

We have the following situation of the shape functions, see in Fig. 4.14.

Following [54], we have the following constraints to derive the higher shape functions:

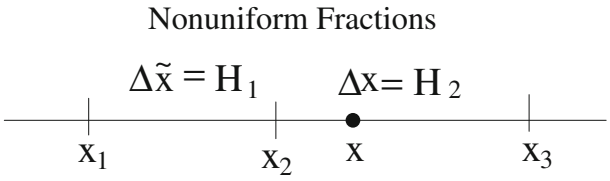



**Fig. 4.14** Nonuniform fractions for the shape functions (nonuniform TSC function)

$$\sum_{p=1}^{m} W_p(x) = 1, \tag{4.212}$$

$$\sum_{p=1}^{m} W_p(x)(x - x_p)^n = const. \tag{4.213}$$

The additional obtained freedom degrees can be used to approximate to the correct potential $\phi_c$.

We improve the interpolation by the fact that

$$\phi(x') = G(x' - x) + \frac{C}{2}\frac{d^2 G(x' - x)}{dx^2} + O(\Delta^3), \tag{4.214}$$

where $G$ is the Greens function and $\phi(x')$ is the correct potential at $x'$.

Later, we could apply the freedom degree with $C$ to the spline fitting of the adaptive grids.

Due to the fact that $\phi$ and $G$ are even functions, we have the following restriction of our constraint:

$$\sum_{p=1}^{m} W_p(x)(x - x_p)^n = \begin{cases} 0 & n \text{ odd} \\ const, & n \text{ even} \end{cases}, \tag{4.215}$$

where

$$W_p = 0, \;\; p \neq 1, 2, \ldots, n. \tag{4.216}$$

Furthermore, the displacement invariance property is given as

$$W_p(x) = W(x - x_p). \tag{4.217}$$

*Example 4.2* In the following, we derive the uniform and nonuniform shape functions.

1. *Uniform Case*

We derive the case of $n = 2$ and $n = 3$.

For $n = 2$, we have three constraint equations:

$$W_1 + W_2 + W_3 = 1, \tag{4.218}$$

$$W_1 x_1 + W_2 x_2 + W_3 x_3 = x, \tag{4.219}$$

$$W_1 x_1^2 + W_2 x_2^2 + W_3 x_3^2 = C + x^2, \tag{4.220}$$

$$W_p = 0 \text{ for } p \neq 1, 2, 3, \tag{4.221}$$

additional, we have to apply to derive the constant $C$:

$$\phi(x') = G(x' - x) + \frac{C}{2}\frac{d^2 G(x' - x)}{dx^2} + O(\Delta^3), \tag{4.222}$$

where $G$ is the Greens function and $\phi(x')$ is the correct potential at $x'$.
For solving the linear equation system, we applied program-code Maxima [68] and we obtain

$$W_1 = \frac{C + (x_2 - x)\ x_3 - x\, x_2 + x^2}{(x_2 - x_1)\ x_3 - x_1\, x_2 + {x_1}^2}, \tag{4.223}$$

$$W_2 = -\frac{C + (x_1 - x)\ x_3 - x\, x_1 + x^2}{(x_2 - x_1)\ x_3 - {x_2}^2 + x_1\, x_2}, \tag{4.224}$$

$$W_3 = \frac{C + (x_1 - x)\ x_2 - x\, x_1 + x^2}{{x_3}^2 + (-x_2 - x_1)\ x_3 + x_1\, x_2}. \tag{4.225}$$

Using the displacement invariance property (4.217) and Eq. (4.216), we obtain

$$W(x) = \begin{cases} \frac{x^2+3\ H\ x+2\ H^2+C}{2H^2}, & -\frac{3}{2}H \le x < -\frac{1}{2}H, \\ \frac{1-(x^2+C)}{H^2}, & -\frac{1}{2}H \le x < \frac{1}{2}H, \\ \frac{x^2-3\ H\ x+2\ H^2+C}{2H^2}, & \frac{H}{2} < x < \frac{3H}{2}, \\ 0, & \text{else.} \end{cases} \tag{4.226}$$

2. For $n = 3$, we have three constraint equations:

$$W_1 + W_2 + W_3 + W_4 = 1, \tag{4.227}$$

$$W_1 x_1 + W_2 x_2 + W_3 x_3 + W_4 x_4 = x, \tag{4.228}$$

$$W_1 x_1^2 + W_2 x_2^2 + W_3 x_3^2 + W_4 x_4^2 = C + x^2, \tag{4.229}$$

$$W_1 x_1^3 + W_2 x_2^3 + W_3 x_3^3 + W_4 x_4^3 = 3xC + x^3, \tag{4.230}$$

$$W_p = 0 \text{ for } p \neq 1, 2, 3, 4, \tag{4.231}$$

additionally, we have to apply to derive the constant $C$:

$$\phi(x') = G(x' - x) + \frac{C}{2}\frac{d^2 G(x' - x)}{dx^2} + O(\Delta^3), \tag{4.232}$$

where $G$ is the Greens function and $\phi(x')$ is the correct potential at $x'$.

For solving the linear equation system, we applied program-code Maxima [68] and we obtain

$$W_1 = \frac{x_4\,\left(C+(x_2-x)\,x_3-x\,x_2+x^2\right)+x_3\,\left(C-x\,x_2+x^2\right)+x_2\,\left(C+x^2\right)-3\,x\,C-x^3}{\left((x_2-x_1)\,x_3-x_1\,x_2+{x_1}^2\right)\,x_4+\left({x_1}^2-x_1\,x_2\right)\,x_3+{x_1}^2\,x_2-{x_1}^3},$$

$$W_2 = -\frac{x_4\,\left(C+(x_1-x)\,x_3-x\,x_1+x^2\right)+x_3\,\left(C-x\,x_1+x^2\right)+x_1\,\left(C+x^2\right)-3\,x\,C-x^3}{\left((x_2-x_1)\,x_3-{x_2}^2+x_1\,x_2\right)\,x_4+\left(x_1\,x_2-{x_2}^2\right)\,x_3+{x_2}^3-x_1\,{x_2}^2},$$

$$W_3 = \frac{x_4\,\left(C+(x_1-x)\,x_2-x\,x_1+x^2\right)+x_2\,\left(C-x\,x_1+x^2\right)+x_1\,\left(C+x^2\right)-3\,x\,C-x^3}{\left({x_3}^2+(-x_2-x_1)\,x_3+x_1\,x_2\right)\,x_4-{x_3}^3+(x_2+x_1)\,{x_3}^2-x_1\,x_2\,x_3},$$

$$W_4 = -\frac{x_3\,\left(C+(x_1-x)\,x_2-x\,x_1+x^2\right)+x_2\,\left(C-x\,x_1+x^2\right)+x_1\,\left(C+x^2\right)-3\,x\,C-x^3}{{x_4}^3+(-x_3-x_2-x_1)\,{x_4}^2+((x_2+x_1)\,x_3+x_1\,x_2)\,x_4-x_1\,x_2\,x_3}.$$

Using the displacement invariance property (4.217) and Eq. (4.216), we obtain

$$W(x) = \begin{cases} \frac{-x^3+6\,H(x^2+C)-11H^2x-3Cx+6H^3}{6H^3}, & -2H \le x < -H, \\ \frac{x^3-2H(x^2+C)-H^2x+3Cx+2H^3}{2H^3}, & -H \le x < 0, \\ \frac{-x^3-2H(x^2+C)+H^2x-3Cx+2H^3}{2H^3}, & 0 \le x < H, \\ \frac{x^3+6\,H(x^2+C)+11H^2x+3Cx+6H^3}{6H^3}, & H \le x < 2H, \\ 0, & \text{else.} \end{cases} \tag{4.233}$$

A simpler notation because of the symmetry is given as

$$W(x) = \begin{cases} \frac{-|x|^3-2H(x^2+C)+H^2|x|-3C|x|+2H^3}{2H^3}, & |x| < H, \\ \frac{|x|^3+6\,H(x^2+C)+11H^2|x|+3C|x|+6H^3}{6H^3}, & H \le |x| < 2H, \\ 0, & \text{else.} \end{cases} \tag{4.234}$$

Such shape function can be applied for the higher order interpolations between grid-free (pusher) and grid parts (solver).

2. *Nonuniform Case*

We derive the cases for $n = 2$, and the same idea is also applied for $n = 3$.

For $n = 2$, we have three constraint equations:

$$W_1 + W_2 + W_3 = 1, \tag{4.235}$$
$$W_1 x_1 + W_2 x_2 + W_3 x_3 = x, \tag{4.236}$$
$$W_1 x_1^2 + W_2 x_2^2 + W_3 x_3^2 = C + x^2, \tag{4.237}$$
$$W_p = 0 \text{ for } p \neq 1, 2, 3, \tag{4.238}$$

additionally, we have to apply to derive the constant $C$:

$$\phi(x') = G(x' - x) + \frac{C}{2} \frac{d^2 G(x' - x)}{dx d\tilde{x}} + O(\Delta^3), \tag{4.239}$$

where $G$ is the Greens function and $\phi(x')$ is the correct potential at $x'$. The adaptive Laplacian is given as $\frac{d^2}{dx d\tilde{x}}$, as given in Eq. (4.202).

We deal with the discussion of the smoothness constraint, which we have as an upper bound of our $C$. For the uniform grid, the discussion is done in [54].

The effects of the charge assignment are given as

$$\phi_p = \frac{1}{8} \frac{2H_1}{H_1 + H_2} G_{p+H_2} + \frac{3}{4} G_p + \frac{1}{8} \frac{2H_2}{H_1 + H_2} G_{p-H_1}, \tag{4.240}$$

and we obtain

$$\phi_p = G_p + \frac{H_1 H_2}{8} \left( \frac{2H_2}{H_1 + H_2} G_{p+H_2} - \frac{2}{H_1 H_2} G_p + \frac{2H_1}{H_1 + H_2} G_{p-H_1} \right),$$
$$\phi_p = G_p + \frac{C}{2} \frac{d}{dx} \frac{d}{d\tilde{x}} G_p, \tag{4.241}$$

and we obtain $C = \frac{H_1 H_2}{4}$.

Next, we solve the linear equation system that we applied program-code Maxima [68] and we obtain

$$W_1 = \frac{C + (x_2 - x)\ x_3 - x\ x_2 + x^2}{(x_2 - x_1)\ x_3 - x_1\ x_2 + {x_1}^2}, \tag{4.242}$$
$$W_2 = -\frac{C + (x_1 - x)\ x_3 - x\ x_1 + x^2}{(x_2 - x_1)\ x_3 - {x_2}^2 + x_1\ x_2}, \tag{4.243}$$
$$W_3 = \frac{C + (x_1 - x)\ x_2 - x\ x_1 + x^2}{{x_3}^2 + (-x_2 - x_1)\ x_3 + x_1\ x_2}. \tag{4.244}$$

Using the displacement invariance property (4.217) and Eq. (4.216), we obtain

$$W(x) = \begin{cases} \frac{x^2-2H_1x+H_2(H_1-x)+H_1^2+C}{H_1(H_1+H_2)}, & -\frac{3H_1}{2} < x < -\frac{H_1}{2}, \\ \frac{-x^2+H_2(x+H_1)-H_1x-C}{H_1H_2}, & -\frac{H_1}{2} < x < -\frac{H_2}{2}, \\ \frac{x^2+2H_2x+H_1(H_2+x)+H_2^2+C}{H_2(H_1+H_2)}, & \frac{H_2}{2} < x < \frac{3H_2}{2}, \\ 0, & \text{else}, \end{cases} \tag{4.245}$$

where $H_1 = x_2 - x_1$, $H_2 = x_3 - x_2$, $H_1 + H_2 = x_3 - x_1$ and $C \in [0, \frac{H_1H_2}{4}]$. So we deal with an adaptive interface with grid lengths $H_1$ and $H_2$.

#### 4.4.4.2 Correction of the Shape Function

For the physical constraints, it has to be fulfilled in the shape functions, and therefore we have additional algorithms to correct the derived shape functions:

- Algorithm 1 (Multigrid idea) for corrected shape function,
- Algorithm 2 (Fixpoint idea) for corrected shape function,
- Improved Pusher: Velocity Verlet,
- Momentum conserved constraint, and
- Spline fitting to fulfil the momentum conservation.

**Algorithm 1 (Multigrid Idea) for Corrected Shape Function**

In the following, we present the algorithm of the corrected shape function. This is an initialization process, which we have to do first one and afterwards we have at the interface such a corrected shape function.

**Algorithm 4.7** (1) Compute the corrected potential at the interface with the fine grid:

$$\phi_{fine}(x') = W_{2,fine}(x)G(x' - x), \tag{4.246}$$

where $G$ is the Greens function (which is given locally in 2D or 3D) and $\rho(x) = 1$, i.e. we assume $W_{2,fine}(x) = 1$.

(2) Compute the uncorrelated potential at the interface with the coarse-grid local (quadratic spline with an assumed $C = C_{uncorrelated}$, e.g. $C_{uncorrelated} = (\Delta x/2)^2$.)

$$\phi_{coarse,uncorrelated}(x') = W_{2,coarse}(x)G(x' - x), \tag{4.247}$$

where $G$ is the Greens function (which is given locally in 2D or 3D) and $\rho(x) = 1$, but we have a different shape function based on the adaptation $W_{2,coarse}(x) \neq W_{2,fine}(x) = 1$.

(3) Compute the corrected adaptive shape function (compute the parameter $C$):

$$\begin{aligned}\phi_{coarse,correlated} &= \frac{C}{2} W_{2,coarse}(x) \frac{\partial}{\partial x} \frac{\partial}{\partial \tilde{x}} G + W_{2,coarse}(x) G \\ &= \frac{C}{2} W_{2,coarse}(x) \frac{\partial}{\partial x} \frac{\partial}{\partial \tilde{x}} G + \phi_{coarse,uncorrelated}(x') \\ &= \phi_{fine}(x'), \end{aligned} \tag{4.248}$$

and we have

$$C = 2 \frac{\phi_{fine}(x') - \phi_{coarse,uncorrelated}(x')}{W_{2,coarse}(x) \frac{\partial}{\partial x} \frac{\partial}{\partial \tilde{x}}}. \tag{4.249}$$

For the initialization of the interface, we have first computed this corrected shape function, and if we do not change the interface afterwards, we could use the fitted spline for all the particles.

In the following, we describe the spline fitting algorithm in Fig. 4.15.

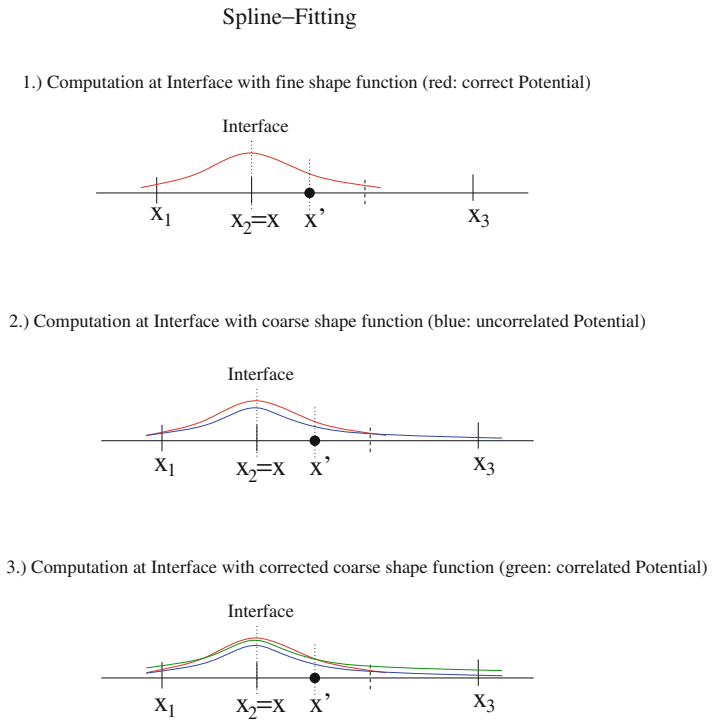



**Fig. 4.15** Spline fitting: fine-coarse interface at interface point $x$

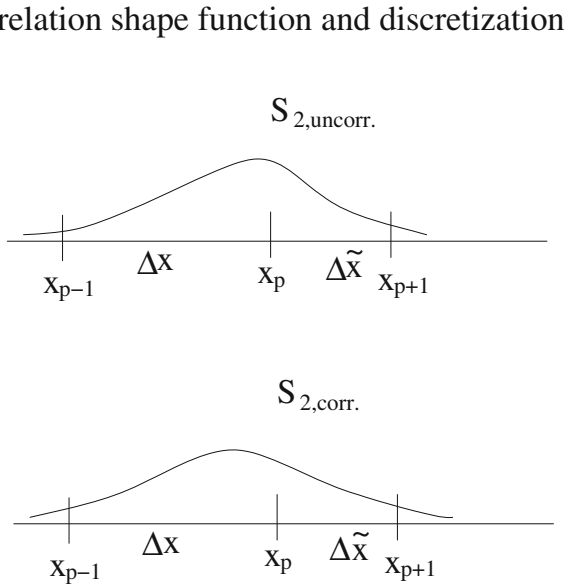



**Fig. 4.16** Correlated shape functions to the adaptive discretization scheme (adapted TSC function)

In Fig. 4.16, we present the correlated shape function with respect to the adaptive discretization scheme.

**Algorithm 2 (Fixpoint Idea) for Corrected Shape function**

In the following, we present an alternative algorithm of the corrected shape function based on forward and backward computations at the interface, which can be formulated to a fixpoint scheme.

The algorithm is given as follows.

**Algorithm 4.8** We start with known $x_i^n, x_p, v_i^{n-1/2}$ and $C_0 = 0$

(1) Forward PIC algorithm starting with $x_i^n$ and $+q$ (positive charge)

$$x_i^n \to \rho_p^n \to \phi_p^n \to E_p^n \to F_i^n \to v_i^{n+1/2} \to x_i^{n+1} \tag{4.250}$$

(2) Backward PIC algorithm starting with $x_i^{n+1}$ and $-q$ (negative charge)

$$x_i^{n+1} \to -\rho_p^{n+1} \to -\phi_p^{n+1} \to -E_p^{n+1} \to -F_i^{n+1} \to -\tilde{v}_i^{n+1/2} \to \tilde{x}_i^n \tag{4.251}$$

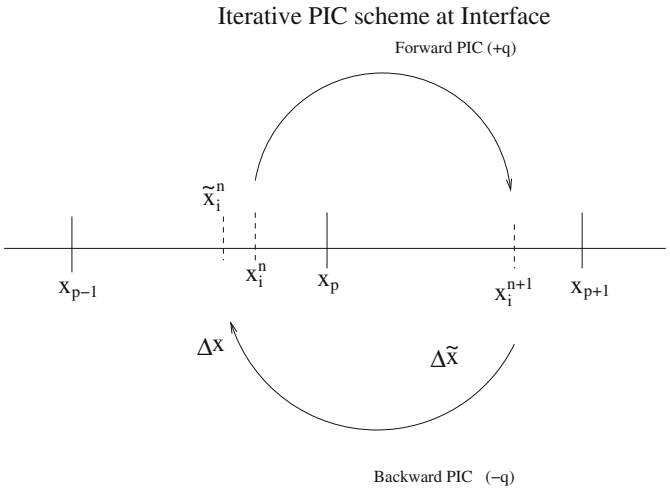



**Fig. 4.17** Iterative PIC (forward and backward computations with PIC at the interface)

(3) Difference Forward PIC and Backward PIC algorithm

$$\Delta x_i = x_i^n - \tilde{x}_i^n, \tag{4.252}$$

(4) Adaptation of parameter $C_j$
We compute the error of the schemes (forward, backward)

$$|W(x_i^n - x_p, C_{j-1}) - W(\tilde{x}_i^n - x_p, C_{j-1})| = \delta W, \tag{4.253}$$

if $\Delta W \leq error$, we are done and $C_{j-1}$ is our novel parameter for the shape function else we compute $C_j$ with

$$W(x_i^n - x_p, C_j) - W(\tilde{x}_i^n - x_p, C_{j-1}) = 0, \tag{4.254}$$

and go to step (1)

In Fig. 4.17, we see the idea of the iterative forward and backward PIC scheme.

**Improved Pusher: Velocity Verlet**

For the backward PIC, we have a problem in computing the backward velocity $\tilde{v}_i^{n+1/2}$, with the simple leap-frog algorithm, see [69, 70], and we have to apply $v_i^{n+3/2}$ which is not given.

Here, an improved second-order scheme to compute also backward a PIC algorithm.

We have to apply the velocity Verlet, which is given as a forward scheme $x_n \rightarrow x_{n+1}$ ($x_n$, $v_n$ is known)

$$v_{n+1/2} = v_n + \frac{1}{2}\Delta t \; F(x_n), \tag{4.255}$$

$$x_{n+1} = x_n + \Delta t \; v_{n+1/2}, \tag{4.256}$$

$$v_{n+1} = v_{n+1/2} + \frac{1}{2}\Delta t \; F(x_{n+1}), \tag{4.257}$$

or a backward scheme $\tilde{x}_{n+1} \rightarrow \tilde{x}_n$ ($\tilde{x}_{n+1}$, $\tilde{v}_{n+1}$ is known):

$$\tilde{v}_{n+1/2} = \tilde{v}_{n+1} - \frac{1}{2}\Delta t \; F(\tilde{x}_{n+1}), \tag{4.258}$$

$$x_n = \tilde{x}_{n+1} - \Delta t \; \tilde{v}_{n+1/2}, \tag{4.259}$$

$$\tilde{v}_n = \tilde{v}_{n+1/2} - \frac{1}{2}\Delta t \; F(\tilde{x}_n). \tag{4.260}$$

*Remark 4.17* Higher order schemes with respect to magnetic and electric field can be obtained by extrapolation schemes [71] or cyclotronic integrators [61].

**Momentum Conserved Constraint**

Idea of spline fitting, see [54], reduces the spatially localized errors based on the adaption at the interface.

We are motivated to embed higher order shape and discretization functions to reduce the local error at the interface of the adaptation.

In book of Hockney [54], the higher order shape functions are introduced to fit at the long-range constraints, and we apply them as a freedom degree to the adaptive grids, see Fig. 4.18.

The full PIC cycle is given as follows (discrete model):

(1) Charge assignment (Method: Spline functions):

$$\rho_p^n = \frac{q}{H} \sum_{i=1}^{N_p} W(x_i - x_p) \tag{4.261}$$

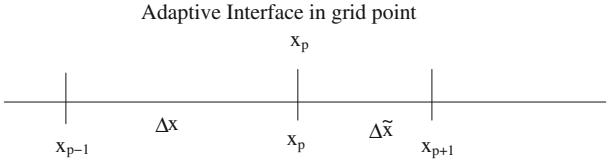



**Fig. 4.18** Adaptive interface

(2) Field equation (Method: Solver)
We have to solve

$$\nabla^2 \phi_p = -\frac{\rho_p}{\varepsilon_0} \tag{4.262}$$

We obtain the notation with the Greens function:

$$\phi_{p,h} = \sum_j \rho_{p',h} G_{h,H}(p, p') \tag{4.263}$$

while we have a discrete Greens function, see the idea of the composite grids, [54].

The discrete analogue of the Greens function to the adaptive finite difference scheme is given as

$$G_{h,H}(\cdot, e_{\Gamma_h^*}) = A_{h,H}^{-1} e_{\Gamma_h^*} \tag{4.264}$$

where $\Gamma_h^* = \Gamma - h$.

If we have not a translation invariant matrix, we have also a non-translation invariant inverse matrix, and therefore also the discrete Greens function is not translation invariant.

The we discretize the electric field with

$$E_{p,h} = \sum_s a_s (\phi_{p+s,h} - \phi_{p-s,h}) \tag{4.265}$$

where $a_s$ is the coefficient for the finite difference discretization.

(3) Force interpolation (Method: Spline functions)

$$F(x_i) = \sum_p W(x_i - x_p) F_p \tag{4.266}$$

(4) Equation of motion (Method: Pusher)

$$v_{i+1/2} = v_{i-1/2} + \frac{\delta t}{2} F(x_i), \tag{4.267}$$

$$x_{i+1} = x_i + \delta t \; v_{i+1/2}, \tag{4.268}$$

$$v_{i+1} = v_{i+1/2} + \frac{\delta t}{2} F(x_{i+1}). \tag{4.269}$$

Based on the PIC cycle, we fulfil the following constraints and conserve the momentum.

Based on the ideas of [54, 57], the conditions

- Identical charge assignment, and
- Correctly space-centred finite difference approximations, while we have the condition

$$d(x_p; x_{p'}) = -d(x_{p'}, x_p) \tag{4.270}$$

are sufficient to fulfil the self-force and inter-particle force, and therefore the momentum constraint.

While we deal with adaptive grids, the constraint 2 (4.270) is only fulfilled with uniform grids.

We propose the following constraint, which is a combination of constraints 1 and 2, while we balance between the freedom degree of the shape functions:

$$d(x_p - x_{p'})W(x_p - x_{p'}, C_{opt}) = -d(x_{p'} - x_p)W(x_{p'} - x_p, C_{opt}) \tag{4.271}$$

$C_{opt} \in [0, H^2/4]$.

Then the momentum conservation is given with respect to the self-force and inter-particle force.

**Spline Fitting to Fulfil the Momentum Conservation**

We obtain the following approach to the Greens function:

$$\begin{aligned} G^p_{ex,q} &= G^p_q + C_{1,exact}\Delta G^p_q\phi(x') \\ &= G^{p-e^d h_l}_{q-e^d h_l} + C_{2,exact}\Delta G^{p-e^d h_l}_{q-e^d h_l} = G^{p-e^d h_l}_{ex,q-e^d h_l}, \end{aligned} \tag{4.272}$$

where $C_{1,exact} = |p-q|/2$, $C_{2,exact} = |(p-e^d h_l) - (q-e^d h_l)|/2$.

To obtain the translation invariant $G^p_{ex,q} == G^{p-e^d h_l}_{ex,q-e^d h_l}$, we have to fit

$$G^p_q + C_1\Delta G^p_q\phi(x') = G^{p-e^d h_l}_{q-e^d h_l} + C_2\Delta G^{p-e^d h_l}_{q-e^d h_l}, \tag{4.273}$$

$$G^{p-e^d h_l}_{q-e^d h_l} = G^p_q + C_1\Delta G^p_q\phi(x') - C_2\Delta G^{p-e^d h_l}_{q-e^d h_l} \tag{4.274}$$

and we can fit $C_1$ and $C_2$ to have a translation invariant function $G^p_q$. Furthermore, $C_1$ and $C_2$ have fulfilled the adaptive higher order discretization scheme.

*Remark 4.18* Here, we apply the similar ideas as in [72] for AMR (adaptive mesh refinement). While we are only approaching to one interface and we deal with higher order shape functions, we are more flexible to derive the constants $C_1$ and $C_2$.

### *4.4.5 2D Adaptive PIC*

In the following, we discuss the extension to the two-dimensional particle in cell method based on adaptive schemes. Here, we have the influence of the higher dimensions to the discretization and shape functions.

In the following, we describe the different tools for the 2d adaptive PIC:

- 2D discretization scheme based on finite difference methods for the different equations (e.g. Maxwell and Newton equation),
- Shape functions:
  - 2D Shape functions (general introduction),
  - 2D adaptive Shape function (linear functions), and
  - 2D adaptive Shape function (quadratic functions).

*Remark 4.19* The discretization and solver schemes are similar to the 1D problem. Based on the FD method, we have only to increase the standard method to a two-dimensional scheme, see [64], and apply the linear equation systems to the solver methods, see [54]. More important are the modifications related to the shape functions, which connect the different models (microscopic and macroscopic model).

In the following, we concentrate on the 2D shape functions.

**2D Shape Functions (General Introduction)**

In the following, we describe higher order shape function for 2D problems.

We can extend the idea of the derivation of the shape function to higher dimensions, in the following, we discuss the 2D shape functions.

*Constraints for the two-dimensional shape functions* n$th$ *order*

For n$th$ order, we have $n+1$ constraint equations:

$$\sum_{\mathbf{P}} W_{\mathbf{P}} = 1, \text{ charge conservation,} \tag{4.275}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} \Delta_i = 0, \text{ first order,} \tag{4.276}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} \Delta_i \Delta_j = C_1 \delta_{ij}, \text{ second order,} \tag{4.277}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} \Delta_i \Delta_j \Delta_k = 0, \text{ third order,} \tag{4.278}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} \Delta_{i_1} \Delta_{i_2} \Delta_{i_3} \Delta_{i_4} = C_2 \delta_{i_1,i_2,i_3,i_4}, \text{ fourth order,} \tag{4.279}$$

$$\vdots, \tag{4.280}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} \Delta_{i_1} \Delta_{i_2} \ldots \Delta_{i_n} = 0, \text{ nth order (n odd),} \tag{4.281}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} \Delta_{i_1} \Delta_{i_2} \ldots \Delta_{i_n} = C_{n/2} \delta_{i_1,i_2,\ldots,i_n}, \text{ nth order (n even),} \tag{4.282}$$

where $\mathbf{p} = (p_1, p_2)$ is a pair labelling the mesh point $\mathbf{p}$ at position $\mathbf{x}_{\mathbf{p}}$. The expansion

of the additional constant $C$ is given as

$$\phi(\mathbf{x}') = \sum_{\mathbf{P}} W_{\mathbf{P}}(\mathbf{x}) \sum_{r,s=0}^{\infty} \frac{\Delta_1^r \Delta_2^s}{r!\, s!} \frac{\partial^{r+s} G}{\partial x^r \partial y^s}, \tag{4.283}$$

where $G$ is the Greens function and $\phi(x')$ is the correct potential at $x'$.

**2D Adaptive Shape Function (Linear Function), Linear Spline $n = 1$, CIC Adaptive for 2D**

In the following, we discuss the two ideas to create 2D shape functions for the two-dimensional case:

- Local one-dimensional (splitting in the locally dimensions).
- Full two-dimensional (non-splitting of the locally dimensions).

We discuss the different approximations.

- One-dimensional Local
  In the following, we discuss the adaptive shape functions.

**Assumption 4.9** We assume that the dimensions can be separated and the shape functions can be constructed as locally one-dimensional problems:

$$W(x, y) = P(x)P(y) \tag{4.284}$$

We assume a four-point stencil for the adaptive finite difference scheme.

We assume the domain $\Omega = [0, L_1] \times [0, L_2]$. In the adaptive grid, we assume that $\Delta x$ is operating in the domain $[0, L_{1,1}] \times [0, L_2]$, while $\Delta\tilde{x}$ is operating in the domain $[L_{1,1}, L_1] \times [0, L_2]$. Furthermore, we assume that $\Delta y$ is operating in the domain $[0, L_1] \times [0, L_{2,1}]$, while $\Delta\tilde{y}$ is operating in the domain $[0, L_1] \times [L_{2,1}, L_2]$. We have the following shape function:

$$S(\mathbf{x} - \mathbf{X}) = \begin{cases} \left(1 - \frac{|x-X|}{\Delta x}\right)\left(1 - \frac{|y-Y|}{\Delta y}\right), & \text{when } |x - X| < \Delta x, |y - Y| < \Delta y \\ & \text{and } (x, y) \in [0, L_{1,1}] \times [0, L_{2,1}], \\ \left(1 - \frac{|x-X|}{\Delta\tilde{x}}\right)\left(1 - \frac{|y-Y|}{\Delta y}\right), & \text{when } |x - X| < \Delta\tilde{x}, |y - Y| < \Delta y \\ & \text{and } (x, y) \in [L_{1,1}, L_1] \times [0, L_{2,1}], \\ \left(1 - \frac{|x-X|}{\Delta x}\right)\left(1 - \frac{|y-Y|}{\Delta\tilde{y}}\right), & \text{when } |x - X| < \Delta x, |y - Y| < \Delta\tilde{y} \\ & \text{and } (x, y) \in [0, L_{1,1}] \times [L_{2,1}, L_2], \\ \left(1 - \frac{|x-X|}{\Delta\tilde{x}}\right)\left(1 - \frac{|y-Y|}{\Delta\tilde{y}}\right), & \text{when } |x - X| < \Delta\tilde{x}, |y - Y| < \Delta\tilde{y} \\ & \text{and } (x, y) \in [L_{1,1}, L_1] \times [L_{2,1}, L_2], \\ 0, & \text{else,} \end{cases} \tag{4.285}$$

where we have $\mathbf{x} = (x, y)^t$.

For the nonuniform mesh function, we have to fulfil the consistency (mass conservation) (4.204).

**Theorem 4.10** *For the nonuniform shape function (4.285), we fulfil the consistency (4.204).*

*Proof* It is sufficient to prove that the shape functions based on each different domain fulfil the condition.

While we can separate to local one-dimensional problem and each dimension is fulfil, see Sect. 4.4.4.1 and we are done.

- Two-dimensional

For $n = 1$, we have three constraint equations:

$$\sum_{\mathbf{P}} W_{\mathbf{P}} = 1, \tag{4.286}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} \Delta_i = 0, \tag{4.287}$$

where $\mathbf{p} = (p_1, p_2)$ is a pair labelling the mesh point $\mathbf{p}$ at position $\mathbf{x}_{\mathbf{p}}$.
We have the following equations:

$$W_1 + W_2 + W_3 = 1, \tag{4.288}$$

$$W_1 x_1 + W_2 x_2 + W_3 x_3 = x, \tag{4.289}$$

$$W_1 y_1 + W_2 y_2 + W_3 y_3 = y. \tag{4.290}$$

By solving Eqs. (4.317)–(4.320), we obtain (using program-code Maxima [68])

$$W_1 = \frac{x\ (y_3 - y_2) + x_2\ (y - y_3) + x_3\ (y_2 - y)}{x_1\ (y_3 - y_2) + x_2\ (y_1 - y_3) + x_3\ (y_2 - y_1)}, \tag{4.291}$$

$$W_2 = -\frac{x\ (y_3 - y_1) + x_1\ (y - y_3) + x_3\ (y_1 - y)}{x_1\ (y_3 - y_2) + x_2\ (y_1 - y_3) + x_3\ (y_2 - y_1)}, \tag{4.292}$$

$$W_3 = \frac{x\ (y_2 - y_1) + x_1\ (y - y_2) + x_2\ (y_1 - y)}{x_1\ (y_3 - y_2) + x_2\ (y_1 - y_3) + x_3\ (y_2 - y_1)}, \tag{4.293}$$

$$\text{for } -\frac{\mathbf{H_2}}{2} \le \mathbf{x} \le \mathbf{H_1}. \tag{4.294}$$

where $\mathbf{x} = (x, y)^t$, $\mathbf{H_1} = (H_{11}, H_{12})^t$ and $\mathbf{H_2} = (H_{21}, H_{22})^t$.

Using the displacement invariance property (4.217) and Eq. (4.216), we obtain

$$
W(x) = \begin{cases}
1 - \frac{x}{H_{11}} - \frac{y}{H_{21}}, & 0 < x < \frac{H_{11}}{2},\ 0 < y < \frac{H_{21}}{2}, \\
1 - \frac{x}{H_{11}} + \frac{y}{H_{22}}, & 0 < x < \frac{H_{11}}{2},\ -\frac{H_{22}}{2} < y < 0, \\
1 + \frac{x}{H_{12}} - \frac{y}{H_{21}}, & -\frac{H_{12}}{2} < x < 0,\ 0 < y < \frac{H_{21}}{2}, \\
1 + \frac{x}{H_{12}} + \frac{y}{H_{22}}, & -\frac{H_{12}}{2} < x < 0,\ -\frac{H_{22}}{2} < y < 0, \\
\\
1 - \frac{x}{H_{11}}, & \frac{H_{11}}{2} < x < \frac{3H_{11}}{2},\ 0 < y < \frac{H_{21}}{2}, \\
1 - \frac{x}{H_{11}}, & \frac{H_{11}}{2} < x < \frac{3H_{11}}{2},\ -\frac{H_{22}}{2} < y < 0, \\
1 + \frac{x}{H_{12}}, & -\frac{3H_{12}}{2} < x < -\frac{H_{12}}{2},\ 0 < y < \frac{H_{21}}{2}, \\
1 + \frac{x}{H_{12}}, & -\frac{3H_{12}}{2} < x < -\frac{H_{12}}{2},\ -\frac{H_{22}}{2} < y < 0, \\
\\
1 - \frac{y}{H_{21}}, & 0 < x < \frac{H_{11}}{2},\ \frac{H_{21}}{2} < y < \frac{3H_{21}}{2}, \\
1 - \frac{y}{H_{21}}, & -\frac{H_{12}}{2} < x < 0,\ \frac{H_{21}}{2} < y < \frac{3H_{21}}{2}, \\
1 + \frac{y}{H_{22}}, & 0 < x < -\frac{H_{11}}{2},\ -\frac{3H_{22}}{2} < y < -\frac{H_{22}}{2}, \\
1 + \frac{y}{H_{22}}, & -\frac{H_{12}}{2} < x < 0,\ -\frac{3H_{22}}{2} < y < -\frac{H_{22}}{2}, \\
\\
0, & \text{else},
\end{cases}
\tag{4.295}
$$

where $\mathbf{H_1} = (H_{11}, H_{21})^t$ and $\mathbf{H_2} = (H_{21}, H_{22})^t$. We deal with an adaptive interface with grid lengths $\mathbf{H_1}$ and $\mathbf{H_2}$, given in Fig. 4.19.

### 2D Adaptive Shape Function (Quadratic Function), Quadratic Splines $n = 2$, CIC Adaptive for 2D

In the following, we discuss the adaptive shape functions.

**Assumption 4.11** We assume that the dimensions can be separated and the shape functions can be constructed as locally one-dimensional problems:

$$
W(x, y) = P(x)P(y). \tag{4.296}
$$

We assume the domain $\Omega = [0, L_1] \times [0, L_2]$. In the adaptive grid, we assume that $\Delta x$ is operating in the domain $[0, L_{1,1}] \times [0, L_2]$, while $\Delta\tilde{x}$ is operating in the

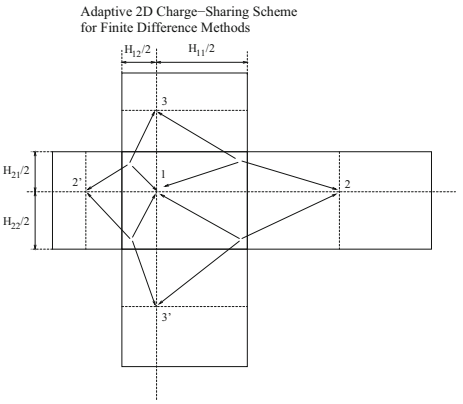



**Fig. 4.19** Two-dimensional adaptive five-point charge-sharing scheme assigns charge to the nearest grid point (labelled 1) and the next-nearest grid points in the east–west direction (labelled 2 and 2′) and in the north–south direction (labelled 3 and 3′)

domain $[L_{1,1}, L_1] \times [0, L_2]$. Furthermore, we assume that $\Delta y$ is operating in the domain $[0, L_1] \times [0, L_{2,1}]$, while $\Delta \tilde{y}$ is operating in the domain $[0, L_1] \times [L_{2,1}, L_2]$.

We have the following pair of equation for the shape functions.

We have the following equations:

$$P_{1,x} + P_{2,x} + P_{3,x} = 1, \tag{4.297}$$

$$P_{1,x}x_1 + P_{2,x}x_2 + P_{3,x}x_3 = x, \tag{4.298}$$

$$P_{1,x}x_1^2 + P_{2,x}x_2^2 + P_{3,x}x_3^2 = x^2 + C_x, \tag{4.299}$$

and

$$P_{1,y} + P_{2,y} + P_{3,y} = 1, \tag{4.300}$$

$$P_{1,y}y_1 + P_{2,y}y_2 + P_{3,y}y_3 = y, \tag{4.301}$$

$$P_{1,y}y_1^2 + P_{2,y}y_2^2 + P_{3,y}y_3^2 = y^2 + C_y. \tag{4.302}$$

The locally one-dimensional shape functions are given as

$$P_x(x) = \begin{cases} P_{x,1}(x) = \frac{x^2 - 2H_{12}x + H_{11}(H_{12} - x) + H_{12}^2 + C_x}{H_{12}(H_{12} + H_{11})}, & -\frac{3H_{12}}{2} < x < -\frac{H_{12}}{2}, \\ P_{x,2}(x) = \frac{-x^2 + H_{11}(x + H_{12}) - H_{12}x - C_x}{H_{12}H_{11}}, & -\frac{H_{12}}{2} < x < -\frac{H_{11}}{2}, \\ P_{x,3}(x) = \frac{x^2 + 2H_{11}x + H_{12}(H_{11} + x) + H_{11}^2 + C_x}{H_{11}(H_{12} + H_{11})}, & \frac{H_{11}}{2} < x < \frac{3H_{11}}{2}, \\ P_{x,4}(x) = 0, & \text{else} \end{cases} \tag{4.303}$$

where $H_{12} = x_2 - x_1$, $H_{11} = x_3 - x_2$, $H_{11} + H_{12} = x_3 - x_1$ and $C_x \in [0, \frac{H_{12}H_{11}}{4}]$. So we deal with an adaptive interface in $x$ direction with grid length $H_{11}$ and $H_{12}$, see also Fig. 4.20. Furthermore, we have

$$P_y(y) = \begin{cases} P_{y,1}(y) = \frac{y^2 - 2H_{22}y + H_{21}(H_{22} - y) + H_{22}^2 + C_y}{H_{22}(H_{22} + H_{21})}, & -\frac{3H_{22}}{2} < y < -\frac{H_{22}}{2}, \\ P_{y,2}(y) = \frac{-y^2 + H_{21}(y + H_{22}) - H_{22}y - C_y}{H_{22}H_{21}}, & -\frac{H_{22}}{2} < y < -\frac{H_{21}}{2}, \\ P_{y,3}(y) = \frac{y^2 + 2H_{21}y + H_{22}(H_{21} + y) + H_{21}^2 + C_y}{H_{21}(H_{22} + H_{21})}, & \frac{H_{21}}{2} < y < \frac{3H_{21}}{2}, \\ P_{y,4}(y) = 0, & \text{else} \end{cases} \tag{4.304}$$

where $H_{22} = y_2 - y_5$, $H_{21} = y_4 - y_2$, $H_{21} + H_{22} = y_4 - y_5$ and $C_y \in [0, \frac{H_{21}H_{22}}{4}]$. So we deal with an adaptive interface in $y$ direction with grid length $H_{21}$ and $H_{22}$, see also Fig. 4.20.

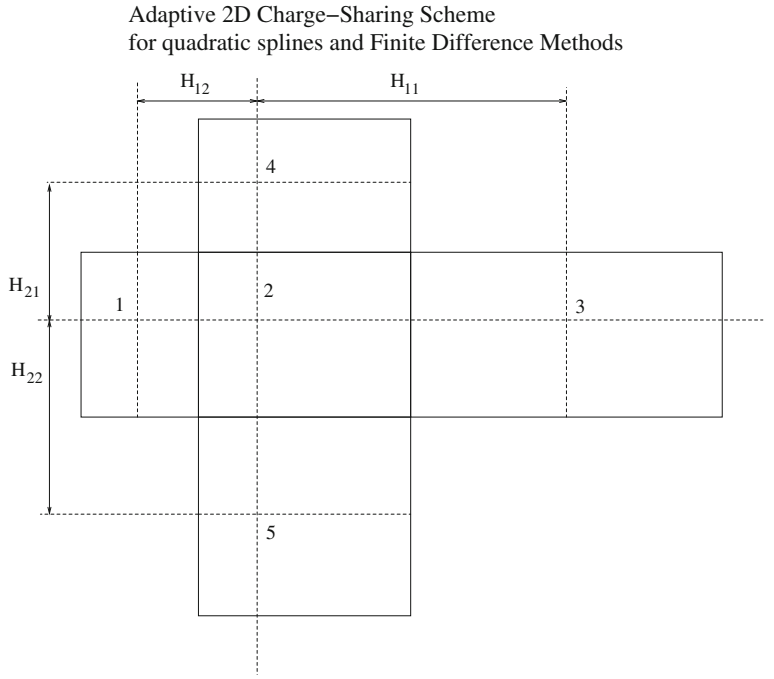



**Fig. 4.20** Two-dimensional adaptive five-point charge-sharing scheme assigns charge to the nearest and the next-nearest grid points

Finally, we obtain the 2D shape function with locally one-dimensional shape functions:

$$W(x,y)=\left\{\begin{array}{l} \\ P_{x,1}(x)P_{y,1}(y),\\ P_{x,2}(x)P_{y,1}(y),\\ P_{x,3}(x)P_{y,1}(y),\\ P_{x,4}(x)P_{y,1}(y),\\ P_{x,1}(x)P_{y,2}(y),\\ P_{x,2}(x)P_{y,2}(y),\\ P_{x,3}(x)P_{y,2}(y),\\ P_{x,4}(x)P_{y,2}(y),\\ P_{x,1}(x)P_{y,3}(y),\\ P_{x,2}(x)P_{y,3}(y),\\ P_{x,3}(x)P_{y,3}(y),\\ P_{x,4}(x)P_{y,3}(y),\\ P_{x,1}(x)P_{y,4}(y),\\ P_{x,2}(x)P_{y,4}(y),\\ P_{x,3}(x)P_{y,4}(y),\\ P_{x,4}(x)P_{y,4}(y). \end{array}\right\} \tag{4.305}$$

For the nonuniform mesh function, we have to fulfil the consistency (mass conservation) (4.204).

**Theorem 4.12** *For the nonuniform shape function (4.285), we fulfil the consistency (4.204).*

*Proof* It is sufficient to prove that the shape functions based on each different domain fulfil the condition.

While we can separate to local one-dimensional problem and each dimension is fulfil, see Sect. 4.4.4.1 and we are done.

### 4.4.6 Application: Multidimensional Finite Difference Method

In the following, we discuss the multidimensional discretization of the Poisson and electric field equation.

The Poisson equations is given as

$$\Delta\phi(X_{i,j,k})=-\tfrac{1}{\varepsilon_0}\rho(X_{i,j,k}),\quad X_{i,j,k}\in[0,L]^3=\Omega, \tag{4.306}$$

$$\phi(X_{i,j,k})=0,\ X_{i,j,k}\in\partial\Omega, \tag{4.307}$$

where $X_{i,j,k}=(x_i,y_j,z_k)^t$ is the three-dimensional coordinate of the particle $(i,j,k)$.

The electric field is given as

$$E_{i,j,k} = -\nabla\phi(X_{i,j,k}), \tag{4.308}$$

where $E_{i,j,k}$ is the electric field in grid point $X_{i,j,k}$.

The multidimensional finite difference equations are given as

$$\frac{\phi_{i+1,j,k} - 2\phi_{i,j,k} + \phi_{i-1,j,k}}{\Delta x^2} + \frac{\phi_{i,j+1,k} - 2\phi_{i,j,k} + \phi_{i,j-1,k}}{\Delta y^2} + \frac{\phi_{i,j,k+1} - 2\phi_{i,j,k} + \phi_{i,j,k-1}}{\Delta z^2} = -\frac{1}{\varepsilon_0}\rho_{i,j,k} \in [0, L], \tag{4.309}$$

$$\phi(0,0,0) = 0,\ \phi(L,0,0) = 0,\ \phi(0,L,0),\ \ldots,\ \phi(L,L,L) = 0, \tag{4.310}$$

where $\phi(x_i, y_j, z_k) = \phi_{i,j,k}$

The electric fields are given as

$$E_{i+1/2,j,k} = -\frac{\phi_{i,j,k} - \phi_{i+1,j,k}}{\Delta x}, \tag{4.311}$$

$$E_{i,j+1/2,k} = -\frac{\phi_{i,j,k} - \phi_{i,j+1,k}}{\Delta y}, \tag{4.312}$$

$$E_{i,j,k+1/2} = -\frac{\phi_{i,j,k} - \phi_{i,j,k+1}}{\Delta z}, \tag{4.313}$$

where we called such a discretization "staggered grids", see [64].

### 4.4.7 Application: Shape Functions for the Multidimensional Finite Difference Method

In the following subsection, we modify the shape functions to the previous introduced multidimensional finite difference method.

For $n = 1$, we have three constraint equations (additional we need one constraint for the second momentum):

$$\sum_{\mathbf{P}} W_{\mathbf{P}} = 1, \tag{4.314}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} \Delta_i = 0, \tag{4.315}$$

$$\sum_{\mathbf{P}} W_{\mathbf{P}} x_p y_p = xy, \tag{4.316}$$

where $\mathbf{p} = (p_1, p_2)$ is a pair labelling the mesh point $\mathbf{p}$ at position $\mathbf{x}_{\mathbf{p}}$.

We have the following equations:

$$W_1 + W_2 + W_3 + W_4 = 1, \tag{4.317}$$
$$W_1 x_1 + W_2 x_2 + W_3 x_3 + W_4 x_4 = x, \tag{4.318}$$
$$W_1 y_1 + W_2 y_2 + W_3 y_3 + W_4 y_4 = y, \tag{4.319}$$
$$W_1 x_1 y_1 + W_2 x_2 y_2 + W_3 x_3 y_3 + W_4 x_4 y_4 = xy. \tag{4.320}$$

By solving Eqs. (4.317)–(4.320), we obtain (using program-code Maxima [68])

$$W_1 = \frac{W_{11}}{W_{12}}, \tag{4.321}$$
$$\begin{aligned} W_{11} = {} & x_3\ (x_2\ ((y_3 - y_2)\ y_4 - y\ y_3 + y\ y_2) + x\ ((y - y_3)\ y_4 + y_2\ y_3 - y\ y_2)) \\ & + x_4\ (x\ ((y_3 - y_2)\ y_4 - y\ y_3 + y\ y_2) + x_2\ ((y - y_3)\ y_4 + y_2\ y_3 - y\ y_2) \\ & + x_3\ ((y_2 - y)\ y_4 + (y - y_2)\ y_3)) + x\ x_2\ ((y_2 - y)\ y_4 + (y - y_2)\ y_3), \end{aligned} \tag{4.322}$$
$$\begin{aligned} W_{12} = {} & x_3\ (x_2\ ((y_3 - y_2)\ y_4 - y_1\ y_3 + y_1\ y_2) + x_1\ ((y_1 - y_3)\ y_4 + y_2\ y_3 - y_1\ y_2)) \\ & + x_4\ (x_1\ ((y_3 - y_2)\ y_4 - y_1\ y_3 + y_1\ y_2) + x_2\ ((y_1 - y_3)\ y_4 + y_2\ y_3 - y_1\ y_2) \\ & + x_3\ ((y_2 - y_1)\ y_4 + (y_1 - y_2)\ y_3)) + x_1\ x_2\ ((y_2 - y_1)\ y_4 + (y_1 - y_2)\ y_3), \end{aligned} \tag{4.323}$$
$$W_2 = -\frac{W_{21}}{W_{22}}, \tag{4.324}$$
$$\begin{aligned} W_{21} = {} & x_3\ (x_1\ ((y_3 - y_1)\ y_4 - y\ y_3 + y\ y_1) + x\ ((y - y_3)\ y_4 + y_1\ y_3 - y\ y_1)) \\ & + x_4\ (x\ ((y_3 - y_1)\ y_4 - y\ y_3 + y\ y_1) + x_1\ ((y - y_3)\ y_4 + y_1\ y_3 - y\ y_1) \\ & + x_3\ ((y_1 - y)\ y_4 + (y - y_1)\ y_3)) + x\ x_1\ ((y_1 - y)\ y_4 + (y - y_1)\ y_3) \end{aligned} \tag{4.325}$$
$$\begin{aligned} W_{22} = {} & x_3\ (x_2\ ((y_3 - y_2)\ y_4 - y_1\ y_3 + y_1\ y_2) + x_1\ ((y_1 - y_3)\ y_4 + y_2\ y_3 - y_1\ y_2)) \\ & + x_4\ (x_1\ ((y_3 - y_2)\ y_4 - y_1\ y_3 + y_1\ y_2) + x_2\ ((y_1 - y_3)\ y_4 + y_2\ y_3 - y_1\ y_2) \\ & + x_3\ ((y_2 - y_1)\ y_4 + (y_1 - y_2)\ y_3)) + x_1\ x_2\ ((y_2 - y_1)\ y_4 + (y_1 - y_2)\ y_3), \end{aligned} \tag{4.326}$$
$$W_3 = \frac{W_{31}}{W_{32}} \tag{4.327}$$
$$\begin{aligned} W_{31} = {} & x_2\ (x_1\ ((y_2 - y_1)\ y_4 - y\ y_2 + y\ y_1) + x\ ((y - y_2)\ y_4 + y_1\ y_2 - y\ y_1)) \\ & + x_4\ (x\ ((y_2 - y_1)\ y_4 - y\ y_2 + y\ y_1) + x_1\ ((y - y_2)\ y_4 + y_1\ y_2 - y\ y_1) \\ & + x_2\ ((y_1 - y)\ y_4 + (y - y_1)\ y_2)) + x\ x_1\ ((y_1 - y)\ y_4 + (y - y_1)\ y_2) \end{aligned} \tag{4.328}$$
$$\begin{aligned} W_{32} = {} & x_3\ (x_2\ ((y_3 - y_2)\ y_4 - y_1\ y_3 + y_1\ y_2) + x_1\ ((y_1 - y_3)\ y_4 + y_2\ y_3 - y_1\ y_2)) \\ & + x_4\ (x_1\ ((y_3 - y_2)\ y_4 - y_1\ y_3 + y_1\ y_2) + x_2\ ((y_1 - y_3)\ y_4 + y_2\ y_3 - y_1\ y_2) \\ & + x_3\ ((y_2 - y_1)\ y_4 + (y_1 - y_2)\ y_3)) + x_1\ x_2\ ((y_2 - y_1)\ y_4 + (y_1 - y_2)\ y_3), \end{aligned} \tag{4.329}$$
$$W_4 = -\frac{W_{41}}{W_{42}}, \tag{4.330}$$
$$\begin{aligned} W_{41} = {} & x_2\ (x_1\ ((y_2 - y_1)\ y_3 - y\ y_2 + y\ y_1) + x\ ((y - y_2)\ y_3 + y_1\ y_2 - y\ y_1)) \\ & + x_3\ (x\ ((y_2 - y_1)\ y_3 - y\ y_2 + y\ y_1) + x_1\ ((y - y_2)\ y_3 + y_1\ y_2 - y\ y_1) \\ & + x_2\ ((y_1 - y)\ y_3 + (y - y_1)\ y_2)) + x\ x_1\ ((y_1 - y)\ y_3 + (y - y_1)\ y_2) \end{aligned} \tag{4.331}$$

$$W_{42} = x_3\ (x_2\ ((y_3 - y_2)\ y_4 - y_1\ y_3 + y_1\ y_2) + x_1\ ((y_1 - y_3)\ y_4 + y_2\ y_3 - y_1\ y_2)) + x_4\ (x_1\ ((y_3 - y_2)\ y_4 - y_1\ y_3 + y_1\ y_2) + x_2\ ((y_1 - y_3)\ y_4 + y_2\ y_3 - y_1\ y_2) + x_3\ ((y_2 - y_1)\ y_4 + (y_1 - y_2)\ y_3)) + x_1\ x_2\ ((y_2 - y_1)\ y_4 + (y_1 - y_2)\ y_3), \tag{4.332}$$

$$\text{for } -\frac{\mathbf{H}}{2} \le \mathbf{x} \le \frac{\mathbf{H}}{2}. \tag{4.333}$$

where $\mathbf{x} = (x, y)^t$, $\mathbf{H} = (H_x, H_y)^t$.

Using the displacement invariance property (4.217) and Eq. (4.216), we obtain

$$W(x) = \begin{cases} \frac{(3\,H_x+3\,|x|)\,H_y-3\,|y|\,H_x-8\,|x|\,|y|}{3\,H_x\,H_y}, & -\frac{\mathbf{H}}{2} < \mathbf{x} < \frac{\mathbf{H}}{2}, \\ \frac{-3\,H_x\,|y|+2\,|x|\,|y|+(3\,H_x-2\,|x|)\,H_y}{3\,H_x\,H_y}, & -\frac{H_x}{2} < x < \frac{H_x}{2},\ \frac{H_y}{2} < |y| < \frac{3\,H_y}{2}, \\ -\frac{6\,H_x\,|y|+x\,(2\,H_y-4\,|y|)-3\,H_x\,H_y}{3\,H_x\,H_y}, & \frac{H_x}{2} < x < \frac{3H_x}{2},\ -\frac{H_y}{2} < y < \frac{H_y}{2}, \\ -\frac{6\,H_x\,y+x\,(2\,H_y-4\,y)-3\,H_x\,H_y}{3\,H_x\,H_y}, & \frac{H_x}{2} < x < \frac{3H_x}{2},\ \frac{H_y}{2} < y < \frac{3H_y}{2}, \\ \frac{H_y\,(x+H_x)-2\,|y|\,x-2\,|y|\,H_x}{H_x\,H_y}, & -\frac{3H_x}{2} < x < -\frac{H_x}{2},\ -\frac{H_y}{2} < y < \frac{H_y}{2}, \\ 0, & \text{else}, \end{cases} \tag{4.334}$$

where $y_2 - y_1 = H_y, x_3 - x_1 = \frac{3}{2}H_x, x_4 - x_1 = -H_x, y_3 - y_1 = \frac{1}{2}H_y, y_2 - y_1 = H_y$. We deal with an adaptive interface with grid length $\mathbf{H} = (H_x, H_y)^t$ and $\mathbf{H}_{coarse} = 2\mathbf{H} = (2H_x, 2H_y)^t$, given in Fig. 4.21.

### 4.4.8 Simple Test Example: Plume Computation of Ion Thruster with 1D PIC Code

In the following, we present a real-life experiment of an ion thruster with plume computations in 1D, see also the work in [55, 56].

In the following, we present a many particle experiment, which is closer to real numerical applications. The experiment is a simplified thruster model in one space dimension and three velocity dimensions, including the channel and the plume region. Referred to [73], we took the following physics parameters:

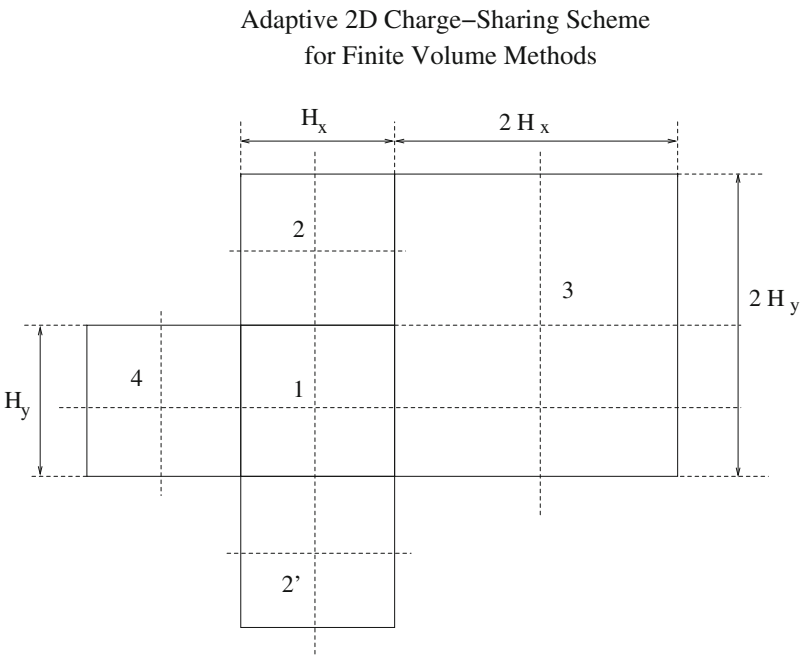



**Fig. 4.21** Two-dimensional adaptive five-point charge-sharing scheme assigns charge to the nearest grid point (labelled 1) and the next-nearest grid points in the east–west direction (labelled 2 and 2′) and the further north–south direction (labelled 3 and 4)

- Potential at the thruster anode was $\Phi A = 400\,\mathrm{V}$ , while the potential at the simulated plume end was taken as zero.
- A static neutral background (here Argon), exponentially decaying in space, was taken for the channel region, with a total density of $n_n = 5.0 \times 10^{18}\mathrm{m}^3$.
- An electron gun was placed in front of the channel exit ($x \in [300\lambda_{De}; 320\lambda_{De}]$) with an injection flux of $f_e = 2.82 \times 10^{11}\,\mathrm{s}^1$. The injected particles had an Gaussian-distributed velocity, due to the thermal velocity $v_{th,e} = 1.03 \times 10^{+6}\,\mathrm{m/s}$. The initial electron temperature was taken as $T_e = 6\,\mathrm{eV}$.
- The implemented reactions are as follows: ionization of $Ar$ with $Ar + e \rightarrow Ar^+ + 2e$ and elastic collisions of electrons and neutrals.

In the 1D model as well as in the real-life thruster, the emitted electrons are getting accelerated by the potential of the anode. These electrons are ionizing the Argon neutrals in the channel, and a plasma is building up, as can be seen in Fig. 4.22. In the real thruster, a configuration of the magnetic field over the whole domain, as well as the resulting in particle–wall interaction, is keeping the plasma in the channel and producing a flat potential, which has a steep decrease at the thruster exit, which accelerates the ions and gives the thrust. While our model is only one dimension, we adapted the magnetic field to the simplified model and took a weak

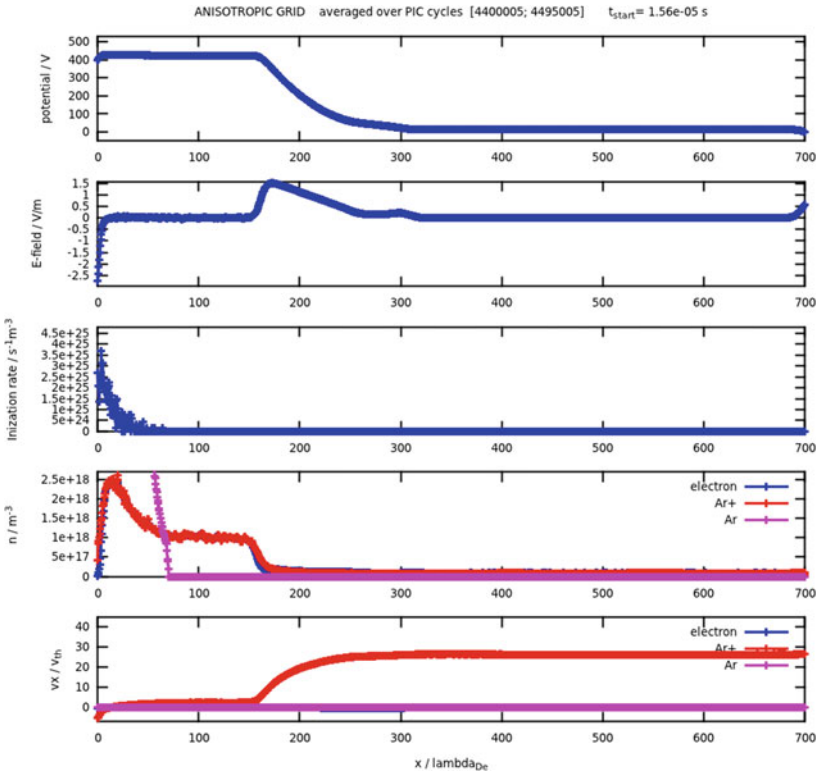



**Fig. 4.22** Stable situation in the plume region with potential, electrical field, particle density and particle velocity plotted over the spatial grid length $L$

magnetic field in the thruster exit region, perpendicular to our space axis $x$. In this region ($x \in [150\lambda_{De}; 20\lambda_{De}]$), the electron velocity in x direction gets weakened, so that electrons can only pass via collisions. With this configuration, we were able to simulate a simple 1D thruster model, which gets steady state after about $1.5 \times 10^6$ $PICsteps = 5.3 \times 10^6$ s, as can be seen in Fig. 4.23.

More computation parameters and the steady-state particle parameters are given in Table 4.2.

*Remark 4.20* The test results are produced with uniform and nonuniform grids. In both results, we could achieve the same one-dimensional behaviours. At least, the numerical results validate the behaviour of the steep gradient on the potential, see Fig. 4.22, that decouples the inner and outer part of the ion thruster.

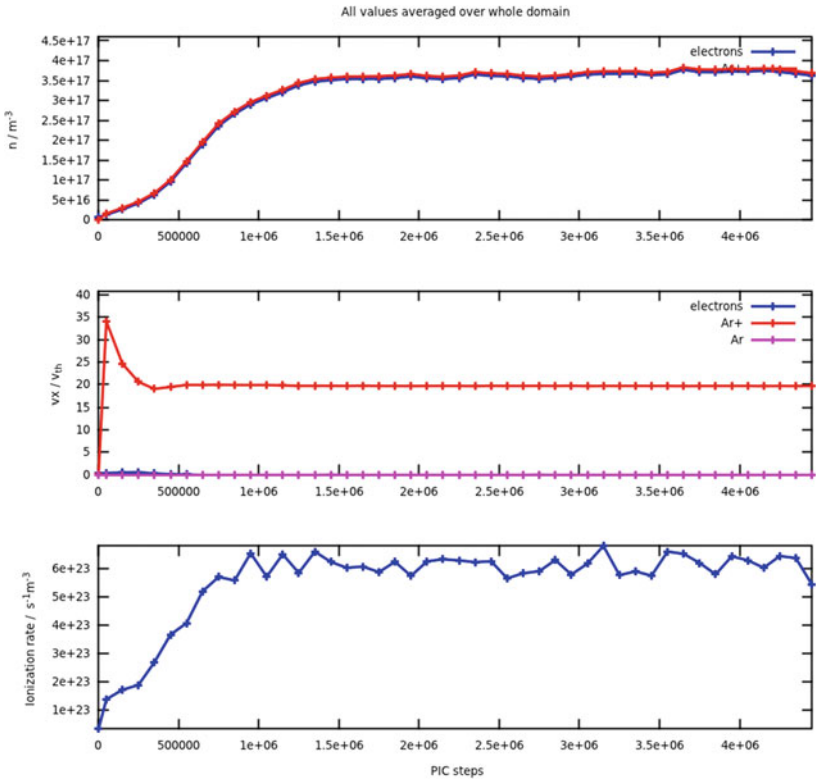



**Fig. 4.23** Averaged species in the domain over the computed time

**Table 4.2** Parameters for the plume computation

| | |
|---|---|
| Electrons | $T_e = 6.00\,\text{eV}(69627\,\text{K})$ |
| Superparticles (electrons, PIC) | $N_{db} \times N_{sp} = 100 \times 6.04 \times 10^1 = 6.038 \times 10^3$<br>$ne = 1.0 \times 10^{12}\,\text{cm}^{-3}(1.0 \times 10^{18}\,\text{m}^{-3})$<br>$v_{th,e} = 1.027274\,\text{e} + 06\,\text{m/s}$ |
| Scaling factors | $\omega_{pe} = 5.64146 \times 10^{10}\,\text{Hz}$<br>$\lambda_{De} = 1.820937 \times 10^{-5}\,\text{m}$ |
| Ions (Ar+) | $v_{th,Ar} = 8.474025 \times 10^2\,\text{m/s}$ |
| Neutrals (Ar) | $n_n = 5.000000e + 18\,\text{m}^{-3}$ |
| Output of the computations | |
| Time step | $dt = 3.545181 \times 10^{-12}\,\text{s}$ |
| Averaging time | $3.545 \times 10^{-3}\text{ns}{-}0.0\,\text{ns}$ |
| Spatial length | $L_{system} = 1.274656 \times 10^1\,\text{mm}$ |

### 4.4.9 Conclusion

We have derived an extension of uniform particle in cell method to nonuniform grids for 1D and 2D equations. The multiscale method, which is given with the parts pusher (microscopic level), solver (macroscopic level) and interpolation/Restriction (complying microscopic and macroscopic level), can be extended with respect to an adaptive scheme. The extensions have been done for the solver, pusher and interpolation functions, which coupled the microscopic and macroscopic model equations. The problem is to modify all parts of the cycle to achieve an extension of the adaptive or nonuniform grids. At least, we can accelerate a simple real-life problem, which has a gap between the high-density (apparatus) and low-density (plume) area, such that adaptive schemes can overcome the uniform step sizes and modify to each disparate spatial and time scales, see [56, 74].

## References

1. E. Weinan, *Principle of Multiscale Modelling* (Cambridge University Press, Cambridge, 2010)
2. J. Geiser, in *Coupled Systems: Theory, Models and Applications in Engineering*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.H. Lai (CRC Press, Chapman & Hall/CRC, 2014)
3. J. Geiser, Operator splitting method for coupled problems: transport and Maxwell equations. Am. J. Comput. Math. Sci. Res. Publ. USA **1**, 163–175 (2011)
4. H. Kim, *Multiscale and Multiphysics Computational Frameworks for Nano- and Bio-systems*. Springer Theses (Springer, Heidelberg, 2011)
5. J. Heinlin, G. Morfill, M. Landthaler, W. Stolz, G. Isbary, J.L. Zimmermann, T. Shimizu, S. Karrer, Plasma medicine: possible applications in dermatology. JDDG, J. Ger. Soc. Dermatol. (2010). ISSN 1610-0379
6. J. Liebmann, J. Scherer, N. Bibinov, P. Rajasekaran, R. Kovacs, R. Gesche, P. Awakowicz, V. Kolb-Bachofen, Biological effects of nitric oxide generated by an atmospheric pressure gas-plasma on human skin cells. Nitric Oxide **24**(1), 8–16 (2011)
7. K.H. Becker, U. Kogelschatz, K.H. Schoenbach, R.J. Barker, *Non-Equilibrium Air Plasmas at Atmospheric Pressure*. Series in Plasma Physics (Taylor and Francis, London, 2004)
8. J. Meichsner, M. Schmidt, R. Schneider, H.-E. Wagner, *Nonthermal Plasma Chemistry and Physics* (CRC Press, Taylor and Francis, Boca Raton, 2012)
9. T.K. Senega, R.P. Brinkmann, Generalized transport coefficients of multicomponent low-temperature plasmas. IEEE Trans. Plasma Sci. **35**(5), 1196–1203 (2007)
10. T.K. Senega, *Schwerteilchen-Transport in Niedertemperatur-Plasmen: Modellierung Technisch Relevanter Plasmen* (Logos, Berlin, 2007)
11. W. Dobrygin, *Modelling and Simulation of a Plasmajet*. Diplomarbeit, Theoretische Elektrotechnik (Ruhr-Universität Bochum, 2014)
12. W. Dobrygin, J. Trischmann, T. Hemke, R.P. Brinkmann, *Simulation und Modellierung der Strömungsdynamik eines nicht-thermischen Plasmajets in Atmosphärendruck.* Vortrag, Theoretische Elektrotechnik (Ruhr-Universität Bochum, Bochum, 2014)
13. J. Schäfer, F. Sigeneger, R. Foest, D. Loffhagen, K.-D. Weltmann, On plasma parameters of a self-organized plasma jet at atmospheric pressure. Eur. Phys. J. D **60**, 531–538 (2010)
14. T.K. Senega, R.P. Brinkmann, A multi-component transport model for non-equilibrium low-temperature low-pressure plasmas. J. Phys. D: Appl. Phys. **39**, 1606–1618 (2006)
15. K.-H. Spatschek, *Theoretische Plasmaphysik* (Teubner Studienbücher, 1990)

16. D. Bothe, On the Maxwell-Stefan approach to multicomponent diffusion. Parabol. Probl. Progr. Nonlinear Differ. Equ. Appl. **80**, 81–93 (2011)
17. J.A. Wesselingh, R. Krishna, *Mass Transfer in Multicomponent Mixtures*. VSSD, 1st edn. (Delft, The Netherland, 2000–2006)
18. K. Böttcher, Numerical solution of a multi-component species transport problem combining diffusion and fluid flow as engineering benchmark. Int. J. Heat Mass Transf. **53**, 231–240 (2010)
19. M. Herberg, M. Meyries, J. Prüss, M. Wilke, Reaction-diffusion systems of Maxwell-Stefan type with reversible mass-action kinetics. Preprint, eprint arXiv:1310.4723 (2013)
20. J. Geiser, in *Iterative Splitting Methods for Differential Equations*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.H. Lai (Chapman & Hall/CRC, 2011)
21. J. Geiser, in *Modelling and Simulation in Engineering with Multi-physics and Multiscale Methods: Theory and Application*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.H. Lai (Chapman & Hall/CRC, 2014)
22. D. Fang, M. Hieber, R. Zi, Global existence results for Oldroyd-B fluids in exterior domains: the case of non-small coupling parameters. Mathematische Annalen **357**(2), 687–709 (2013)
23. V. Giovangigli, *Multicomponent Flow Modeling* (Birkhäuser, Basel, 1999)
24. R. Krishna, R. Taylor, Multicomponent mass transfer theory and applications, in *Handbook for Heat and Mass Transfer*, vol. 2, Chap. 7, ed. by N. Cheremisinoff (Gulf, Houston, 1986)
25. R. Krishna, J. Wesselingh, The Maxwell-Stefan approach to mass transfer. Chem. Eng. Sci. **52**, 861911 (1997)
26. J. Bear, Y. Bachmat, *Introduction to Modeling of Transport Phenomena in Porous Media* (Kluwer Academic Publishers, Dordrecht, 1991)
27. D.J. Acheson, *Elementary Fluid Dynamics.* Oxford Applied Mathematics & Computing Science Series (2002)
28. R. Balescu, *Transport Processes in Plasma: Classical Transport*, vol. 1 (North Holland Publishing, Amsterdam, 1988)
29. B.V. Alexeev, *Generalized Boltzmann Physical Kinetics*, 1st edn. (Elsevier Science, Amsterdam, 2004)
30. M.A. Lieberman, A.J. Lichtenberg, *Principle of Plasma Discharges and Materials Processing*, 2nd edn. (Wiley, New York, 2005)
31. K.N. Kulkarni, Multicomponent diffusion in ternary and quaternary diffusion couples and in multilayered assemblies. Ph.D. thesis, Purdue University, 2008
32. A. Spille-Kohoff, E. Preus, K. Böttcher, Numerical solution of multi-component species transport in gases at any total number of components. Int. J. Heat Mass Transf. **55**, 5373–5377 (2012)
33. J. Bear, *Dynamics of Fluids in Porous Media* (American Elsevier, New York, 1972)
34. J. Geiser, *Discretization and Simulation of Systems for Convection-Diffusion-Dispersion Reactions with Applications in Groundwater Contamination*. Monograph, Series: Groundwater Modelling, Management and Contamination (Nova Science Publishers, Inc., New York, 2008)
35. R.E. Ewing, Up-scaling of biological processes and multiphase flow in porous media. *IIMA Volumes in Mathematics and its Applications*, vol. 295 (Springer, New York, 2002), pp. 195–215
36. E. Fein, Software package $r^3t$: model for transport and retention in porous media. Final report, GRS-192, Braunschweig (2004)
37. M. Genuchten, Convective-dispersive transport of solutes involved in sequential first-order decay reactions. Comput. Geosci. **11**(2), 129–147 (1985)
38. S. Larsson, V. Thomee, *Partial Differential Equations with Numerical Methods.* Text in Applied Mathematics, vol. 45 (Springer, Heidelberg, 2003)
39. P. Knabner, L. Angerman, *Numerical Methods for Elliptic and Parabolic Partial Differential Equations: An Applications-oriented Introduction*. Texts in Applied Mathematics (Springer, Heidelberg, 2003)
40. R.J. LeVeque, *Finite Volume Methods for Hyperbolic Problems.* Cambridge Texts in Applied Mathematics (Cambridge University Press, Cambridge, 2002)

41. S.V. Patankar, *Numerical Heat Transfer and Fluid Flow* (Taylor and Francis, 1980)
42. J. Geiser, Discretization methods with analytical solutions for convection-diffusion dispersion-reaction equations and applications. J. Eng. Math. **57**(1), 79–98 (2007)
43. J. Geiser, Iterative operator-splitting methods with higher order time-integration methods and applications for parabolic partial differential equations. J. Comput. Appl. Math. Elsevier, Amsterdam, The Netherlands **217**, 227–242 (2008)
44. J. Geiser, Recent advances in splitting methods for multiphysics and multiscale: theory and applications. J. Algorithms Comput. Technol. Multi-Sci. Brentwood, Essex, UK, accepted August 2014 (to be published second issue 2015)
45. K. Nanbu, Theory of cumulative small-angle collisions in plasmas. Phys. Rev. E Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top. **55**(4), 4642–4652 (1997)
46. B.I. Cohen, L. Divol, A.B. Langdon, E.A. Williams, Effects of ion-ion collisions and inhomogeneity in two-dimensional kinetic ion simulations of stimulated Brillouin backscattering. Phys. Plasmas **13**(2), 022705 (2006)
47. B.I. Cohen, A.M. Dimits, A. Friedman, R.E. Caflisch, Time-step considerations in particle simulation algorithms for coulomb collisions in plasmas. IEEE Trans. Plasma Sci. **38**(9), 2394–2406 (2010)
48. P. Vabishchevich, *Additive Operator-difference Schemes: Splitting Schemes* (De Gruyter, Berlin, 2014)
49. A. Taflove, S.C. Hagness, *Computational Electrodynamics: The Finite-Difference Time-Domain Method*, 3rd edn. (Artech House, Norwood, 2005)
50. Y. Yang, R.S. Chen, E.K.N. Yung, The unconditionally stable Crank Nicolson FDTD method for three-dimensional Maxwell's equations. Microw. Opt. Technol. Lett. **48**(8), 1619–1622 (2006)
51. J. Shibayama, M. Muraki, J. Yamauchi, H. Nakano, Efficient implicit FDTD algorithm based on locally one-dimensional scheme. Electron. Lett. **41**(19) (2005)
52. A. Taflove, *Computational Electrodynamics: The Finite Difference Time Domain Method* (Artech House Inc., Boston, 1995)
53. R.I. McLachlan, G.R.W. Quispel, Splitting methods. Acta Numerica 341–434 (2002)
54. R. Hockney, J. Eastwood, *Computer Simulation Using Particles* (CRC Press, 1985)
55. J. Duras, K. Matyash, D. Tskhakaya, O. Kalentev, R. Schneider, Self-force in 1D electrostatic particle-in-cell codes for non-equidistant grids. Contrib. Plasma Phys. **54**(8), 697–711 (2014)
56. K. Lueskow, J. Duras, O. Kalentev, K. Matyash, J. Geiser, R. Schneider, D. Tskhakaya. Non-equidistant particle-in-cell for ion thruster plumes, in *Proceedings of the 33rd IEPC*, October, 2013, Washington, DC, USA, IEPC-2013-067 (2013)
57. D. Tskhakaya, K. Matyash, R. Schneider, F. Taccogna, The particle-in-cell method. Contrib. Plasma Phys. **47**(8–9), 563–594 (2007)
58. O. Kalentev, K. Matyash, J. Duras, K. Lueskow, R. Schneider, N. Koch, M. Schirra, Electrostatic ion thrusters—towards predictive modeling. Contrib. Plasma Phys. **54**(2), 235–248 (2014)
59. F.H. Harlow, The particle-in-cell method for numerical solution of problems in fluid dynamics. Methods Comput. Phys. 319–343 (1964)
60. P. Colella, M.R. Dorr, J.A.F. Hittinger, D.F. Martin, High-order, finite-volume methods in mapped coordinates. J. Comput. Phys. **230**(8), 2952–2976 (2011)
61. L. Patacchini, I.H. Hutchinson, Explicit time-reversible orbit integration in particle in cell codes with static homogeneous magnetic field. J. Comput. Phys. **228**(7), 2604–2615 (2009)
62. G. Lapenta, DEMOCRITUS: an adaptive particle in cell (PIC) code for object-plasma interactions. J. Comput. Phys. **230**(12), 4679–4695 (2011)
63. J. Geiser, M. Arab, Modelling, optimzation and simulation for a chemical vapor deposition. J. Porous Media, Begell House Inc., Redding, USA **12**(9), 847–867 (2009)
64. B. Gustafsson, *High Order Difference Methods for Time dependent PDE*. Springer Series in Computational Mathematics, vol. 38 (Springer, Heidelberg, 2007)
65. G.H. Shortly, R. Weller, Numerical solutions of Laplace's equation. J. Appl. Phys. **9**, 334–348 (1938)

66. Chr. Grossmann, H.G. Roos, M. Stynes, *Numerical Treatment of Partial Differential Equations*. Universitext, 1st edn. (Springer, New York, 2007)
67. F. Taccogna, S. Longo, M. Capitelli et al., Particle-in-cell simulation of stationary plasma thruster. Contrib. Plasma Phys. **47**(8–9), 635–656 (2007)
68. Maxima verison 5.26.0. Maxima: A Computer Algebra System. Online software resource: http://maxima.sourceforge.net/ (2011)
69. I.P. Omelyana, I.M. Mrygloda, R. Folk, Optimized Forest-Ruth- and Suzuki-like algorithms for integration of motion in many-body systems. Comput. Phys. Commun. **146**(2), 188–202 (2002)
70. E. Forest, R.D. Ruth, Fourth-order symplectic integration. Phys. D: Nonlinear Phenom. **43**(1), 105–117 (1990)
71. S.A. Chin, J. Geiser, Multi-product operator splitting as a general method of solving autonomous and non-autonomous equations. IMA J. Numer. Anal. **31**(4), 1552–1577 (2011)
72. P. Colella, P.C. Norgaard, Controlling self-force errors at refinement boundaries for AMR-PIC. J. Comput. Phys. **229**, 947–957 (2010)
73. K. Matyash, O. Kalentev, R. Schneider, Kinetic simulation of the stationary HEMP thruster including the near-field plume region, in *Presented at the 31st International Electric Populsion Conference (IEPC), IEPC-2009-110* (2009)
74. J. Geiser, F. Riedel, Comparison of integrators for electromagnetic particle in cell methods: algorithms and applications, in *Proceedings*, arXiv:1411.0816, November 2014

# Chapter 5
# Engineering Applications

**Abstract** In this chapter, we discuss the different engineering applications related to multicomponent and multiscale models, that occur in different categories (microscopic, mesoscopic and macroscopic). As we have described in the Introduction on p. xxv. That such models can be used as basic model to couple to more complicate models, describing materials, interfaces, etc., see Rosso and de Baas (Review of materials modelling: what makes a material function? Let me compute the ways, 2014, [1]).

We deal with the following engineering applications with the two classified multiscaling approaches, see also the Introduction on p. xxv and in Fig. 5.1:

- Different time- or spatial scales of same model (mono model or basic model).
- Linking of different models (different models or multimodel).

One of the main contributions to link the different scales and models together are the coupling techniques. In our motivation, such coupling of different time- and spatial scales or coupling of different models, need the suggested methods, e.g. multiscale methods, multicomponent methods, that allow a data transfer between the different scales and models. Such methods, we have introduced in the previous sections and now, we will close the gap between the theoretical discussions of methods and their applications to engineering problems. In such a stage, we have to adapt the numerical schemes with respect to the real-life properties and we obtain truly working multiscale approaches, that we solve the engineering complexities, see [1].

Based on the real-life applications, we could study such helpful coupling of different scales or different models. Therefore, we overcome the gap of the recent problem in linking different scales of complicated engineering models in the industrial applications.

## 5.1 Multiscale Methods for Langevin-Like Equations

**Abstract** In this section, we discuss multiscale methods, that solve Langevin-like equations, see [2]. The underlying ideas are to split into a deterministic and stochastic part of the Langevin equation, see [3]. The splitting methods are based on



J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5_5

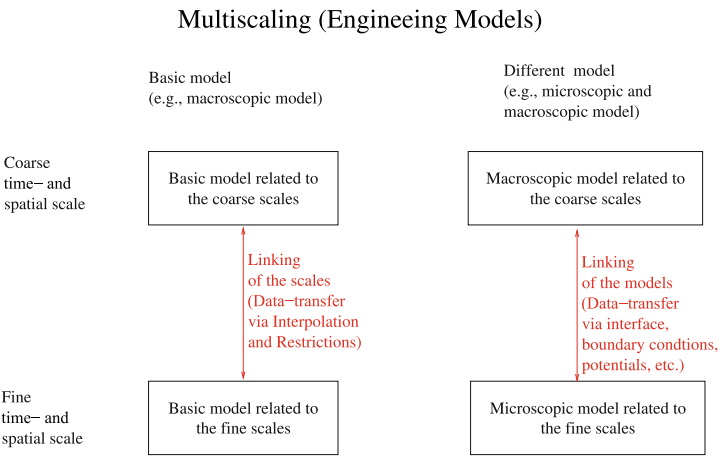



**Fig. 5.1** Engineering models of multiscaling with one model and different models

additive and iterative schemes, which are discussed with respect of their benefits and drawbacks, see [4]. We are motivated to reduce computational time for Coulomb collisions in plasma done in particle simulations. We extend splitting schemes, which are well-known in deterministic applications, to stochastic applications and modify the methods with respect to the stochastic terms. Such an idea allows to solve the multiscale behaviour of the coarse deterministic and fine stochastic timescales in an adequate computational time.

### *5.1.1 Introduction of the Problem*

In the underlying problem, we are motivated to develop fast algorithms to solve the Coulomb collisions in plasma simulations, see [5].

Recently in the literature, we can find two main ideas to deal with the Coulomb collisions in particle simulations. Here we have the following two ideas:

- Binary algorithm: Particles in a finite cell selected into binary pairs. The collision is computed by the scattered velocities by an underlying angle whose statistical variance is modelled by the theory of Coulomb collisions, see [6, 7].
- Test particle algorithm: We deal with a dual idea to present the collisions by defining test and field particles. The velocity in the dual space of the test particle is computed by a Langevin equations with the drag and the diffusion coefficients. This dual space is influenced by the moments of the primary space the field-particle velocity distribution, which are deposited on the primary space mesh [8–12].

The second idea is scratched in the following Fig. 5.2.

We have the following contributions based on the multiscale approximation, which is done by splitting methods:

- Reduction of the numerical error: Each splitting method has a numerical error (splitting error). To reduce the error, we apply adaptivity or higher order splitting schemes for the deterministic and also stochastic part.
- Conservation of the underlying physics: for example, particle transport problems need long-term evolutions, means conservation of the dynamics, e.g. symplecticity of the schemes.

The following characterization is given to the underlying model problem:

- Microscopic model (each particle is treated via an individual equation (transport and collision operators)).
- Plasma simulations are done with particle transport models, where ionized particles are transported via an electromagnetic field and particles can be collide.

Another classification is given to the solver process of the plasma simulation, here we distinguish to two different solver processes and different problems to solve such an equation:

- Forward problem: All parameters of the model equation (e.g. stochastic differential equation) are known, e.g. physical laws, heuristics etc.

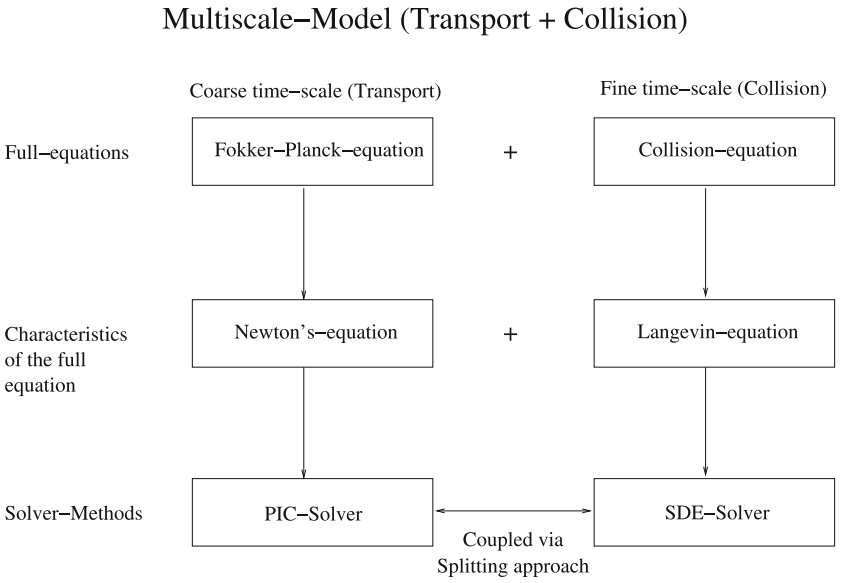



**Fig. 5.2** Fokker–Planck and collision equation solved with their characteristics as Newton's and Langevin equations

- Backward problem: An experimental data set of the particles are given and we reconstruct the parameters, e.g. drag, diffusion, potential, etc., of the underlying model equation (e.g. ambit stochastics, inverse modelling).

For our contribution, we deal with the first-model approach, means we assume to have all underlying parameters of the stochastic differential equation (SDE).

We have the following Assumption 5.1 to our model problem for which we can apply the underlying Langevin equations.

**Assumption 5.1** Coulomb collisions can be approximated via defining test and field particles. The test particle velocity is subjected to drag and diffusion and can be derived as a stochastic differential equations for the three velocity dimensions $(v, \mu, \phi)$ by using Langevin equations, see [8].

A first example of the underlying results are given in Fig. 5.3, we see the particle velocities in a 2D and 3D presentation.

**From Fokker–Planck to Langevin Equations**

In the following, we discuss the modification from Fokker–Planck to Langevin equation. We deal with the Fokker–Planck equation with collision operator given as

$$\frac{\partial f_\alpha}{\partial t} + \mathbf{v} \cdot \frac{\partial f_\alpha}{\partial \mathbf{x}} + \frac{q_\alpha}{m_\alpha}(\mathbf{E} + \mathbf{v} \times \mathbf{B}) \cdot \frac{\partial f}{\partial \mathbf{v}} = \frac{\partial f_\alpha}{\partial t}|_{coll}, \tag{5.1}$$

where $f_\alpha(\mathbf{x}, \mathbf{v})$ is the phase-space distribution function (density) of a charged plasma species $\alpha$ submitted to electromagnetic field $(\mathbf{E}, \mathbf{B})$.

Further the Landau's collision term is given as

$$\frac{\partial f_\alpha}{\partial t}|_{coll} = \frac{\partial}{\partial \mathbf{v}} \cdot \left( \pi \, q_\alpha^2 \, \lambda \sum_\beta q_\beta^2 \int \left( f_\alpha \frac{\partial f'_\beta}{\partial \mathbf{v}'} - f'_\beta \frac{\partial f_\alpha}{\partial \mathbf{v}'} \right) \frac{u^2 I - \mathbf{uu}}{u^3} \right) d^3\mathbf{v}', \tag{5.2}$$

where the sum is over the index $\beta$ of the plasma charged particle species, $q_\beta$ is the charge of species $\beta, f_\beta(\mathbf{x}, \mathbf{v}')$, $\mathbf{u} = \mathbf{v} - \mathbf{v}'$, $u = |\mathbf{u}|$ and $\lambda$ is the Coulomb logarithm.

**From Langevin Equation to the Coulomb Scattering Test Particle Problem**

We apply Eq. (5.2) with respect to a consistent test particle, isotropic Maxwellian background reduction, see [13].

We obtain the following test particle equation:

$$\begin{aligned} \frac{\partial f_t}{\partial t}|_{coll} = &- \frac{\partial}{\partial v}(F_D(v) f_t) + \frac{\partial^2}{\partial v^2}(D_v(v) f_t) \\ &+ \frac{\partial}{\partial \mu}(2 D_a(v) \mu f_t) + \frac{\partial^2}{\partial \mu^2}(D_a(v)(1-\mu^2) f_t) + \frac{\partial^2}{\partial \phi^2}\left( \frac{D_a(v)}{(1-\mu^2)} f_t \right) \end{aligned} \tag{5.3}$$

where $v$ is the speed, $\mu = \cos(\theta)$, with $\theta$ is the angle of the axial direction and $\phi$ is the azimuthal angle.

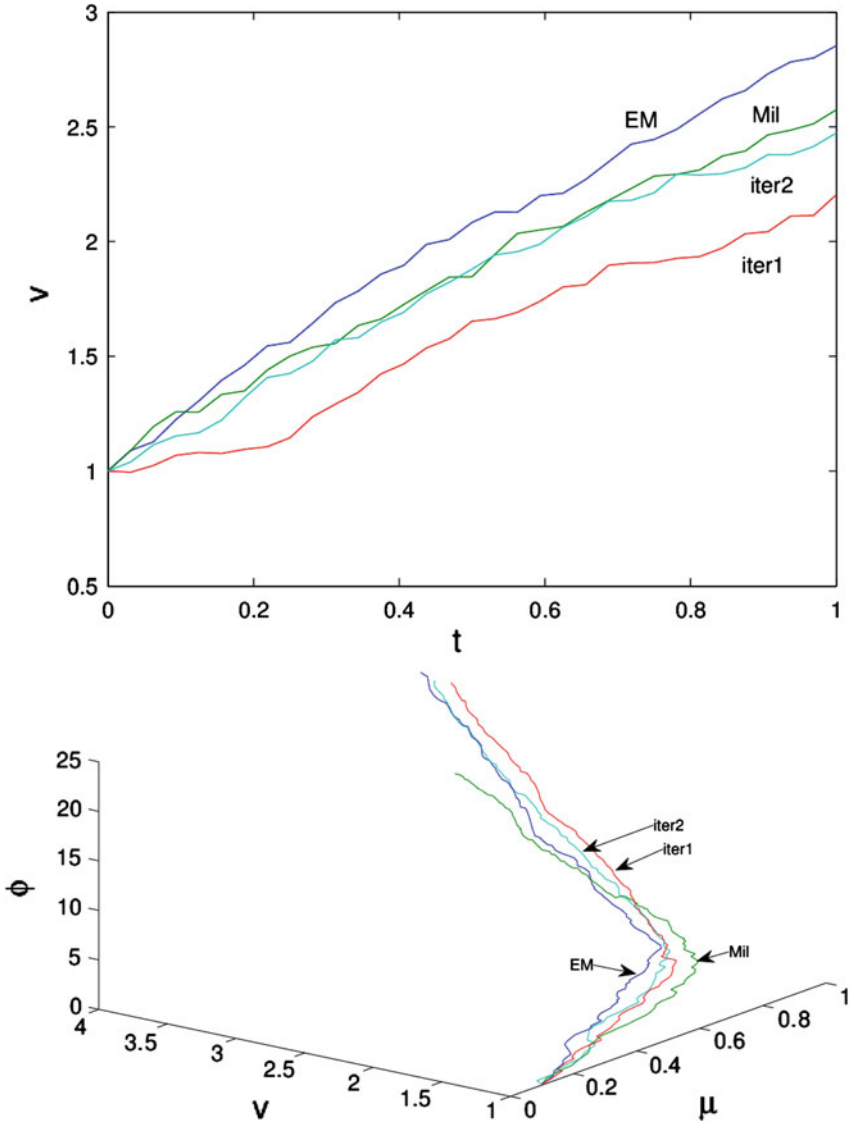



**Fig. 5.3** Velocity $v$ of a particle and 3D presentation of the velocity components for one underlying particle, see [3]

The SDE system of the Coulomb scattering test particle problem is given in the following form:

$$dv(t) = F_D(v)dt + \sqrt{2D_v(v)}dW_v(t), \tag{5.4}$$

$$d\mu(t) = -2D_a(v)\mu\ dt + \sqrt{2D_a(v)(1-\mu^2)}dW_\mu(t), \tag{5.5}$$

$$d\phi(t) = \sqrt{\frac{D_a(v)}{(1-\mu^2)}}dW_\phi(t). \tag{5.6}$$

*Remark 5.1* The SDE system is strongly coupled and also nonlinear, therefore, we have taken into account linearization techniques, e.g. fixpoint or Newton's schemes, see also [14, 15], or derive higher order methods, e.g. Mitstein schemes, as discussed in [13].

For a detailed understanding, we discuss in the following 1D problem.

### 5.1.2 Introduction of the 1D Model Equations

We are motivated to develop fast algorithms to solve Fokker–Planck equation with Coulomb collisions in plasma simulations.

The Fokker–Planck equations are given as

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} - E(x)\frac{\partial f}{\partial v} = \frac{\partial}{\partial v}\left(-\gamma v f + \beta^{-1}\gamma\frac{\partial f}{\partial v}\right), \tag{5.7}$$

where we could decouple such a FP equation into the PIC (particle in cell) part and the SDE part.

- PIC-part

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} - E(x)\frac{\partial f}{\partial v} = 0, \tag{5.8}$$

- SDE part

$$\frac{\partial f}{\partial t} = \frac{\partial}{\partial v}\left(-\gamma v f + \beta^{-1}\gamma\frac{\partial f}{\partial v}\right), \tag{5.9}$$

where we solve the characteristics:

- PIC-part

$$\frac{dx}{dt} = v, \tag{5.10}$$

$$\frac{dv}{dt} = -E(x) = \frac{\partial U}{\partial x}, \tag{5.11}$$

where $U$ is the potential.

- SDE part

$$\frac{dx}{dt} = 0, \tag{5.12}$$

$$dv = -\gamma v dt + \sqrt{2\beta^{-1}\gamma} dW, \tag{5.13}$$

We apply the following nonlinear SDE problem:

$$\frac{dx}{dt} = v, \tag{5.14}$$

$$dv(t) = \frac{\partial}{\partial x} U(x) - \gamma v dt + \sqrt{2\beta^{-1}\gamma} dW, \tag{5.15}$$

where $W$ is a Wiener process, $\gamma$ is the thermostat parameter, $\beta$ the inverse Temperature.

A long solution to the SDE is distributed according to a probability measure with density $\pi$ satisfying:

$$\pi(x, v) = C^{-1} \exp\left(-\beta\left(\frac{v^2}{2} + U(x)\right)\right), \tag{5.16}$$

where $x > 0.0$, $v \in \mathbb{R}$.

### 5.1.3 Analytical Methods for Mixed Deterministic–Stochastic Ordinary Differential Equations

The following, we present an algorithm, which is based on solving the mixture of deterministic and stochastic ordinary differential equations.

The idea is based on the deterministic variation of constants to embed perturbed right-hand sides.

We deal with the following equations:

$$\frac{dX}{dt} = V, \tag{5.17}$$

$$dV = -E(x)dt - AVdt + BdW,$$

$$\text{with } X(0) = X_0,\ V(0) = V_0, \tag{5.18}$$

where $W$ is a Wiener process with the $N(0, \sqrt{\Delta})$ distributed.

We rewrite to a linear operator and a nonlinear and stochastic function.

$$\frac{d\mathbf{X}}{dt} = \tilde{A}\mathbf{X} + \mathbf{E}(\mathbf{X}) + \frac{d\mathbf{W}}{dt},$$

$$\text{with } \mathbf{X}_0 = (X_0, V_0)^t, \tag{5.19}$$

where $\mathbf{X} = (X, V)^t$ is the solution vector, $\mathbf{X}_0 = (X_0, V_0)^t$ is the initial vector, the matrix is $\tilde{A} = \begin{pmatrix} 0 & 1 \\ 0 & -A \end{pmatrix}$, the nonlinear function is $\mathbf{E} = \begin{pmatrix} 0 \\ -E(X) \end{pmatrix}$ and the stochastic function is $\frac{d\mathbf{W}}{dt} = \begin{pmatrix} 0 \\ B\frac{dW}{dt} \end{pmatrix}$.

The analytical solution is given with the exact integration of the $\exp(\tilde{A}s)$(variation of constants):

$$\begin{aligned}\mathbf{X}(t^{n+1}) &= \exp(\tilde{A}\Delta t)\mathbf{X}_0 + \int_{t^n}^{t^{n+1}} \exp(\tilde{A}(t^{n+1} - s))\, \mathbf{E}(\mathbf{X}(s))\, ds \\ &\quad + \int_{t^n}^{t^{n+1}} \exp(\tilde{A}(t^{n+1} - s))\, d\mathbf{W}_s, \\ \mathbf{X}(t^{n+1}) &= \exp(\tilde{A}\Delta t)\mathbf{X}_0 + \tilde{\mathbf{E}}(\mathbf{X}_0) + \tilde{\mathbf{W}}(\mathbf{X}_0), \end{aligned} \tag{5.20}$$

where the electric field integral is computed with a higher order exponential Runge–Kutta method, see:

Integration of the E-field function with fourth-order Runge–Kutta method:

$$\mathbf{k}_1 = \Delta t \mathbf{E}(\mathbf{X}^n), \tag{5.21}$$

$$\mathbf{k}_2 = \Delta t \left( \mathbf{E} \left( \exp(\tilde{A}\Delta t/2)\mathbf{X}^n + \frac{1}{2}\exp(\tilde{A}\Delta t/2)\mathbf{k}_1 \right) \right), \tag{5.22}$$

$$\mathbf{k}_3 = \Delta t \left( \mathbf{E} \left( \exp(\tilde{A}\Delta t/2)\mathbf{X}^n + \frac{1}{2}\mathbf{k}_2 \right) \right), \tag{5.23}$$

$$\mathbf{k}_4 = \Delta t\, (\mathbf{E}(\exp(\tilde{A}\Delta t)\mathbf{X}^n + \exp(\tilde{A}\Delta t/2)\mathbf{k}_2)), \tag{5.24}$$

$$\tilde{\mathbf{E}}(\mathbf{X}^n) = \frac{1}{6}\Big( \exp(\tilde{A}\Delta t)k_1 + 2\exp(\tilde{A}\Delta t/2)(\mathbf{k}_2 + \mathbf{k}_3) + \mathbf{k}_4) \Big), \tag{5.25}$$

and the stochastic integral is computed as

$$\begin{aligned}\tilde{\mathbf{W}}(\mathbf{X}^n) &= \int_{t^n}^{t^{n+1}} \exp(\tilde{A}(t^{n+1} - s)) d\mathbf{W}_s \\ &= \sum_{j=0}^{N-1} \exp\left( \tilde{A} \left( \frac{t^{n,j} + t^{n,j+1}}{2} \right) \right) (\mathbf{W}(t^{n,j+1}) - \mathbf{W}(t^{n,j})), \end{aligned} \tag{5.26}$$

$$\Delta t = (t^{n+1} - t^n)/N,\, t^{n,j} = \Delta t + t^{n,j-1},\, t^{n,0} = t^n, \tag{5.27}$$

where we can decide the accuracy based on the number of intermediate time steps $\Delta t = (t^{n+1} - t^n)/N$ and $N$ is the number of the finer time points in the coarse time step $\Delta t_{coarse} = t^{n+1} - t^n$.

### 5.1.4 A–B Splitting with Analytical Methods for Mixed Deterministic–Stochastic Ordinary Differential Equations

We deal with the following equations:

$$\frac{dX}{dt} = V, \tag{5.28}$$

$$dV = -E(x)dt - AVdt + BdW,$$

$$\text{with } X(0) = X_0,\ V(0) = V_0, \tag{5.29}$$

where $W$ is a Wiener process with the $N(0, \sqrt{\Delta t})$ distributed.

We propose the following A–B splitting scheme of Eq. (5.29):

$$\frac{dX}{dt} = V,\ X(t^n) = X_n, t \in [t^n, t^{n+1}], \tag{5.30}$$

$$dV_1 = -AVdt + BdW,\ V_1(t^n) = V_n, t \in [t^n, t^{n+1}], \tag{5.31}$$

$$dV_2 = -E(X)dt,\ V_2(t^n) = V_1(t^{n+1}),\ X(t^n) = X_n, t \in [t^n, t^{n+1}], \tag{5.32}$$

where $W$ is a Wiener process with $N(0, \sqrt{\Delta t})$ distributed.

We apply the analytical solutions in the different A–B splitting steps and they are given as:

$$X(t^{n+1}) = X(t^n) + \int_{t^n}^{t^{n+1}} V(s)\, ds, \tag{5.33}$$

$$V_1(t^{n+1}) = \mathscr{E}(t)V(t^n) + \int_{t^n}^{t^{n+1}} \mathscr{E}(t^{n+1} - s)B\, dW_s \tag{5.34}$$

$$V_2(t^{n+1}) = V_1(t^{n+1}) + \int_{t^n}^{t^{n+1}} (-E(X(s)))\, ds, \tag{5.35}$$

where the operator $\mathscr{E}(t)$ is given as

$$\mathscr{E}(t) = \exp(-At). \tag{5.36}$$

*Remark 5.2* The simple A–B splitting scheme did not preserve the symplectic behaviour and cannot be applied for large-scale computations. Such a problem can be solved by adding more steps and correct the previous solutions of the A–B splitting method. Therefore we discuss in the next subsection the modification to a predictor–corrector A–B splitting scheme.

### 5.1.5 Improved A–B Splitting Scheme: Predictor–Correction Idea

In the following, we present an implicit AB scheme, which is related to symplectic Störmer–Verlet methods, see [16].

We deal with the following approach:

$$X_1(t^{n+1}) = X(t^n) + \int_{t^n}^{t^{n+1}} V(s)\, ds, \tag{5.37}$$

$$V_1(t^{n+1}) = \mathscr{E}(t)V(t^n) + \int_{t^n}^{t^{n+1}} \mathscr{E}(t^{n+1} - s)B\, dW_s \tag{5.38}$$

$$V_2(t^{n+1}) = V_1(t^{n+1}) + \int_{t^n}^{t^{n+1}} (-E(X_1(s)))\, ds, \tag{5.39}$$

$$X_2(t^{n+1}) = X(t^n) + \int_{t^n}^{t^{n+1}} V_2(s)\, ds, \tag{5.40}$$

where the fourth step of the algorithm is of the idea to add to semi-implicit Euler schemes together and obtain a symplectic A–B splitting scheme or also known in the context of the Strörmer–Verlet method.

*Remark 5.3* We deal with a weak first-order scheme which has a symplectic behaviour.

Originally the idea to develop such schemes was based on the midpoint scheme, which is an implicit scheme. Then, we have the freedom degree to transform the numerical scheme to a symplectic scheme.

The midpoint rule is given as

$$y(t^{n+1}) = y(t^n) + hJ^{-1}\nabla H\left(\frac{y^n + y^{n+1}}{2}\right), \tag{5.41}$$

which can be approximated as a simple A–B splitting:

$$\tilde{y}(t^{n+1}) = y(t^n) + hJ^{-1}\nabla H\left(\frac{y^n}{2}\right), \tag{5.42}$$

$$y(t^{n+1}) = y(t^{n+1}) + hJ^{-1}\nabla H\left(\frac{\tilde{y}^{n+1}}{2}\right), \tag{5.43}$$

for sufficient small $\Delta t$ and large $n$, we achieve $||y(t^{n+1}) - \tilde{y}(t^{n+1})|| \to 0$.

*Proof* As a first-order approximation the improved A–B splitting can be written as an improved Euler–Mayurama scheme. Therefore we have a sympectic scheme of first order.

### 5.1.6 Improved Explicit Scheme Based on the Predictor–Correction Idea

The ideas are based on the Predictor–Corrector methods, first predict a solution of $X$ and later correct the solution of $X$, like the staggered grid idea in the Verlet algorithm.

As for the A–B splitting scheme, we can also modify the Euler–Maruyama and the Milstein scheme as shown in the following.

**Predictor–Corrector Euler–Maruyama and Milstein Schemes**

With an additional time step, we could improve the explicit schemes to symplectic preserving schemes.

- The Euler–Maruyama scheme is given as

$$X(t^{n+1}) = X(t^n) + \Delta t\ V(t^n), \tag{5.44}$$
$$V(t^{n+1}) = V(t^n) - \Delta t\ E(X(t^n)) - \Delta t\ AV(t^n) + B\Delta W, \tag{5.45}$$
$$X(t^{n+1}) = X(t^n) + \Delta t\ V(t^{n+1}), \tag{5.46}$$

  where $\Delta W = W(t^{n+1} - W(t^n)) = rand\sqrt{\Delta t}$ and *rand* is the Gaussian normal distribution $N(0, 1)$.
- The Milstein scheme is given as

$$X(t^{n+1}) = X(t^n) + \Delta t\ V(t^n), \tag{5.47}$$
$$V(t^{n+1}) = V(t^n) - \Delta t\ E(X(t^n)) - \Delta t\ AV(t^n) + B\Delta W + \frac{1}{2}BB^t((\Delta W)^2 - \Delta t), \tag{5.48}$$
$$X(t^{n+1}) = X(t^n) + \Delta t\ V(t^{n+1}), \tag{5.49}$$

  where $\Delta W = W(t^{n+1} - W(t^n)) = rand\sqrt{\Delta t}$ and *rand* is the Gaussian normal distribution $N(0, 1)$.

**Theorem 5.2** *The predictor–corrector Euler–Maruyama scheme is symplectic, which means*

$$dx_{n+1} \wedge dy_{n+1} = dx_n \wedge dy_n. \tag{5.50}$$

*Proof* The predictor–corrector Euler–Maruyama scheme is given as

$$x(t^{n+1}) = x(t^n) + \Delta t\ y(t^n), \tag{5.51}$$
$$y(t^{n+1}) = y(t^n) - \Delta t\ x(t^n) + \sigma\Delta W, \tag{5.52}$$
$$x(t^{n+1}) = x(t^n) + \Delta t\ y(t^{n+1}), \tag{5.53}$$

and we have

$$x(t^{n+1}) = (1 - (\Delta t)^2)x(t^n) + \Delta t\ y(t^n) + \Delta t\sigma\,\Delta W, \tag{5.54}$$
$$y(t^{n+1}) = y(t^n) - \Delta t\ x(t^n) + \sigma\,\Delta W, \tag{5.55}$$

and the algorithm is given as

$$\begin{pmatrix} x(t^{n+1}) \\ y(t^{n+1}) \end{pmatrix} = \begin{pmatrix} (1 - (\Delta t)^2) & \Delta t \\ -\Delta t & 1 \end{pmatrix} \begin{pmatrix} x(t^n) \\ y(t^n) \end{pmatrix} + \begin{pmatrix} r_n \\ s_n \end{pmatrix} \Delta W, \tag{5.56}$$

where $a_n = (1 - \Delta t^2)$, $b_n = \Delta t$, $c_n = -\Delta t$, $d_n = 1$ and $r_n = \Delta t\sigma$, $s_n = \sigma$.

Based on the symplecticity, we have

$$dx_{n+1} \wedge dy_{n+1} = (a_n d_n - b_n c_n) dx_n \wedge dy_n, \tag{5.57}$$
$$dx_{n+1} \wedge dy_{n+1} = ((1 - (\Delta t)^2) - (\Delta t)^2) x_n \wedge dy_n, \tag{5.58}$$
$$dx_{n+1} \wedge dy_{n+1} = x_n \wedge dy_n, \tag{5.59}$$

and it is independent based on the time step $\Delta t$.

*Remark 5.4* The same proof idea can be applied to the Milstein scheme.

### 5.1.7 CFL Condition for the Explicit Schemes

We apply explicit schemes and also our semi-analytical scheme is embedded to explicit schemes, therefore we have CFL conditions, which restrict our time steps.

We deal with the following equations:

$$\frac{dX}{dt} = V, \tag{5.60}$$
$$dV = -E(x)dt - AV dt + B dW,$$
$$\text{with } X(0) = X_0,\ V(0) = V_0, \tag{5.61}$$

or

$$\frac{d^2X}{dt^2} = -E(x) - A\frac{dX}{dt} + B dW,$$
$$\text{with } X(0) = X_0,\ \frac{dX(0)}{dt} = V_0. \tag{5.62}$$

In Eq. (5.62), we have the CFL condition for the term $A\frac{dX}{dt}$ as

$$\Delta t \leq \frac{1}{||A||}, \tag{5.63}$$

where we assume $||\cdot||$ is an appropriate norm for the matrices, e.g. maximum norm. Further for the second part $-E(x)$ of the Eq. (5.62), we have the CFL condition:

$$\Delta t^2 \leq \frac{|X^n|}{|E(X^n)|}, \tag{5.64}$$

where $X^n = X(t^n)$ is the solution of $X$ to the old time point $t^n$ and $|\cdot|$ is the equivalent norm for the vectors, e.g. maximum vector norm.

*Example 5.1* If we apply the delicate singular electric field $E(X) = \frac{2}{X^3} - 2X$, we have the following CFL condition:

$$\Delta t \leq \sqrt{\frac{1}{\frac{1}{(|X^n|)^4} + 1}}, \tag{5.65}$$

where for small $|X^n| < 1$, we have

$$\Delta t \leq \sqrt{|X^n|}. \tag{5.66}$$

*Remark 5.5* For the scheme, we have prepared the additional conditions, here the CFL condition and the symplecticity. Now, we fulfil the criterion of a stable and long-term preserving method, which can be applied to our engineering problems.

### *5.1.8 Numerical Examples*

We deal with the Coulomb test particle problem with the following 1D Langevin equations, which is given in the following nonlinear SDE problem:

$$\frac{dx}{dt} = v, \tag{5.67}$$

$$dv(t) = \frac{\partial}{\partial x} U(x) - \gamma v dt + \sqrt{2\beta^{-1}\gamma} dW, \tag{5.68}$$

where $W$ is a Wiener process, $\gamma$ is the thermostat parameter, $\beta$ the inverse temperature.

A long solution to the SDE is distributed according to a probability measure with density $\pi$ satisfying:

$$\pi(x, v) = C^{-1} \exp\left(-\beta\left(\frac{v^2}{2} + U(x)\right)\right), \tag{5.69}$$

where $x > 0.0$, $v \in \mathbb{R}$.

We test the following methods:

- Verlet Integrator,
- Analytical Solution as discussed in Sect. 5.1.4,
- A–B Splitting method,
- Predictor–Corrector A–B Splitting method as discussed in Sect. 5.1.5,
- Improved Explicit schemes (Euler–Maruyama and Milstein scheme) as discussed in Sect. 5.1.5.

We test the following oscillators:

1. The harmonic oscillator $U(x) = \frac{1}{2}x^2, E(x) = -x$.
2. The unharmonic oscillator $U(x) = \frac{1}{3}x^3, E(x) = -x^2$.
3. The trigonometric oscillator $U(x) = -\cos(x), E(x) = -\sin(x)$.
4. The impact oscillator $U(x) = \frac{1}{x^2} + x^2, E(x) = 2\frac{1}{x^3} - 2x$.

We discuss in the following paragraphs the different oscillators solved with our proposed methods:

1. The harmonic oscillator $U(x) = \frac{1}{2}x^2, E(x) = -x$ is presented in Fig. 5.4.

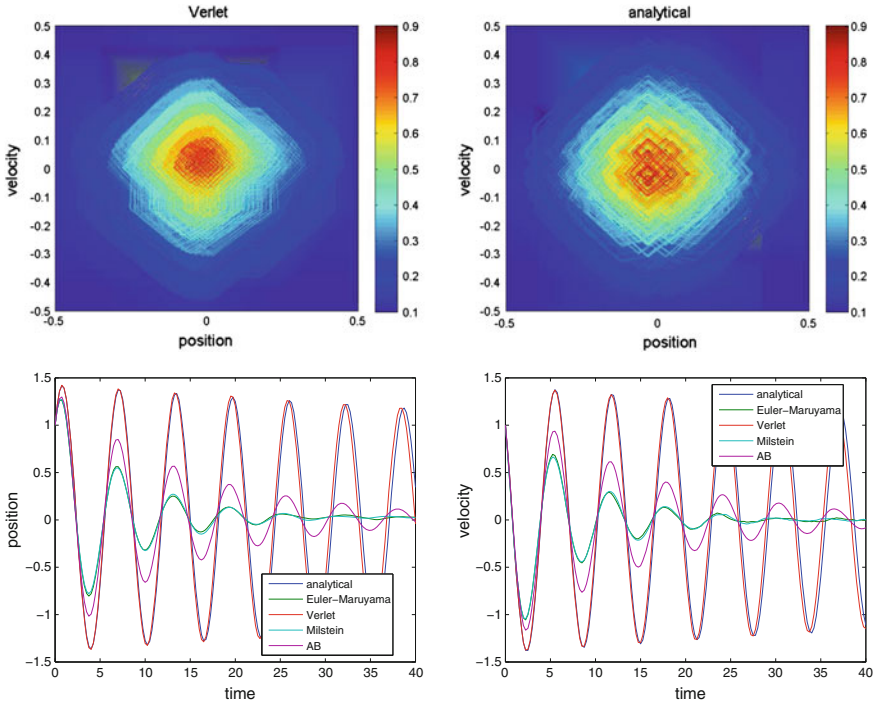



**Fig. 5.4** We apply $U(x) = \frac{1}{2}x^2, E(x) = -x$. The *upper figures* present the contours of the Hamiltonian with the Verlet algorithm (*left figure*) and the analytical algorithm (*right figure*), the *lower figures* presents the $x$ and $v$ solutions of the Verlet algorithm

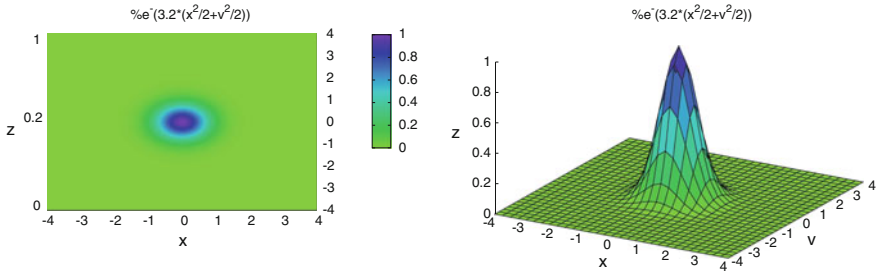



**Fig. 5.5** The distribution of the harmonic oscillator $U(x) = \frac{1}{2}x^2$ with $\beta = 3.2$, where $A = 0.1$, $B = 0.25$

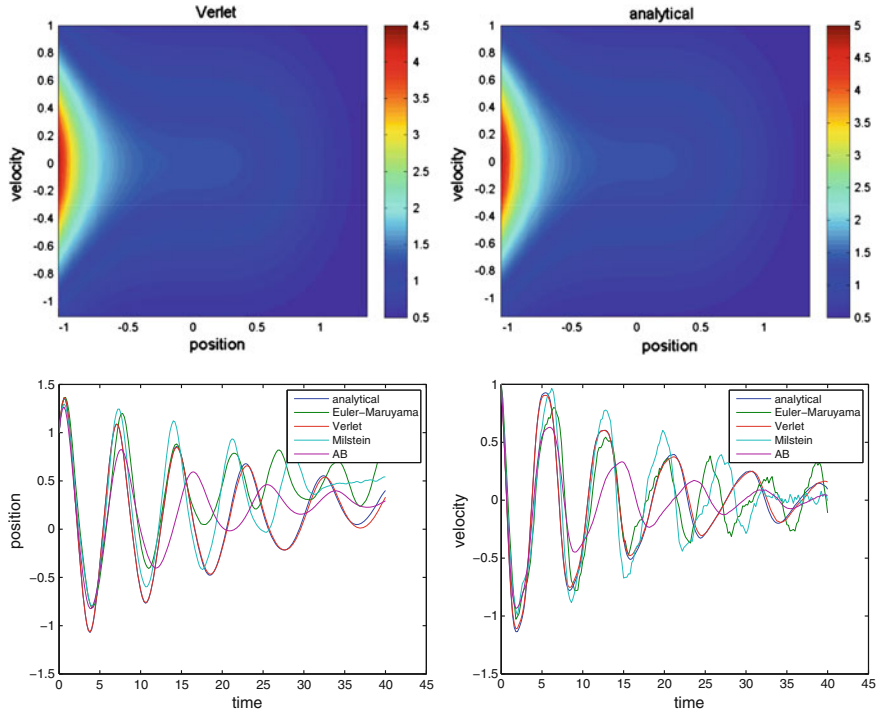



**Fig. 5.6** We apply $U(x) = \frac{1}{3}x^3, E(x) = -x^2$. The *upper figures* present the contours of the Hamiltonian with the Verlet algorithm (*left figure*) and the analytical algorithm (*right figure*), the *lower figures* presents the $x$ (*left*) and $v$ (*right*) solutions of the Verlet algorithm

The distribution of the harmonic oscillator is given in Fig. 5.5.

2. The unharmonic oscillator $U(x) = \frac{1}{3}x^3, E(x) = -x^2$ is presented in Fig. 5.6. The distribution of the unharmonic oscillator is given in Fig. 5.7.
3. The trigonometric oscillator $U(x) = -\cos(x), E(x) = -\sin(x)$ is presented in Fig. 5.8.

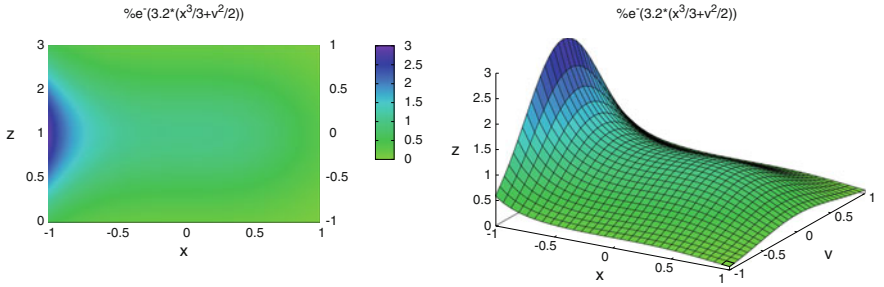



**Fig. 5.7** The distribution of the unharmonic oscillator $U(x) = \frac{1}{3}x^3$ with $\beta = 3.2$, where $A = 0.1, B = 0.25$

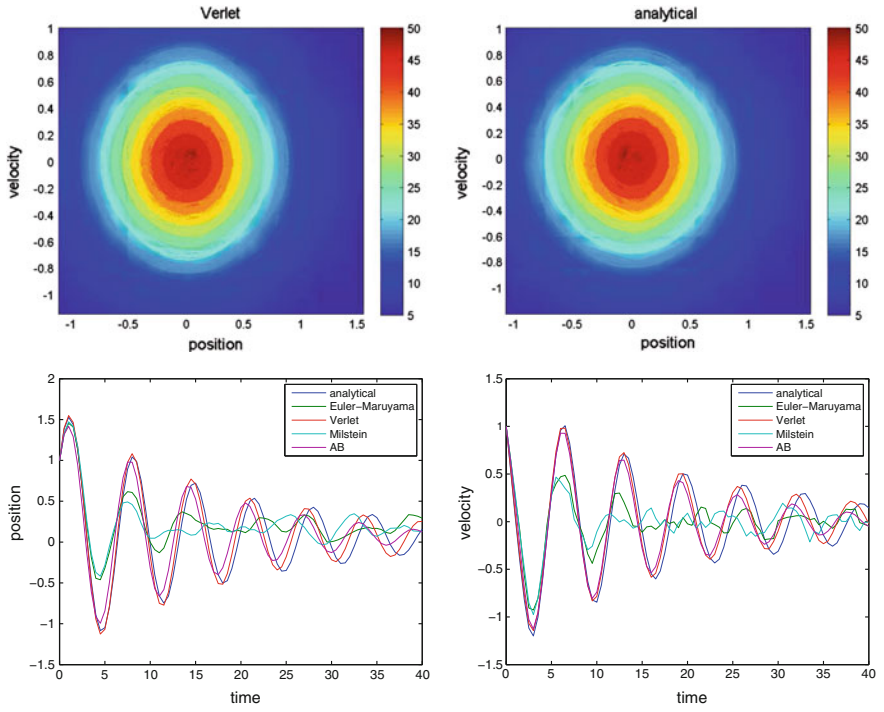



**Fig. 5.8** We apply $U(x) = -\cos(x), E(x) = -\sin(x)$. The *upper figures* present the contours of the Hamiltonian with the Verlet algorithm (*left figure*) and the analytical algorithm (*right figure*), the *lower figures* presents the $x$ (*left*) and $v$ (*right*) solutions of the Verlet algorithm, where $A = 0.1, B = 0.25$

The distribution of the trigonometric oscillator is given in Fig. 5.9.

4. The impact oscillator $U(x) = \frac{1}{x^2} + x^2, E(x) = 2\frac{1}{x^3} - 2x$ is presented in Fig. 5.10.

   We discuss the equilibrium distribution of the impact oscillator, which is given with

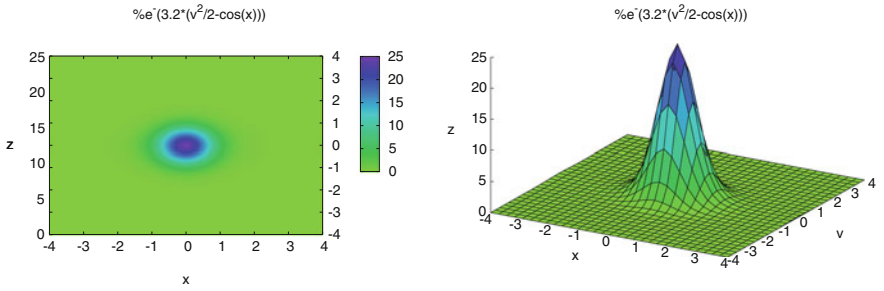



**Fig. 5.9** The distribution of the trigonometric oscillator $U(x) = -\cos(x)$ with $\beta = 3.2$, where $A = 0.1, B = 0.25$

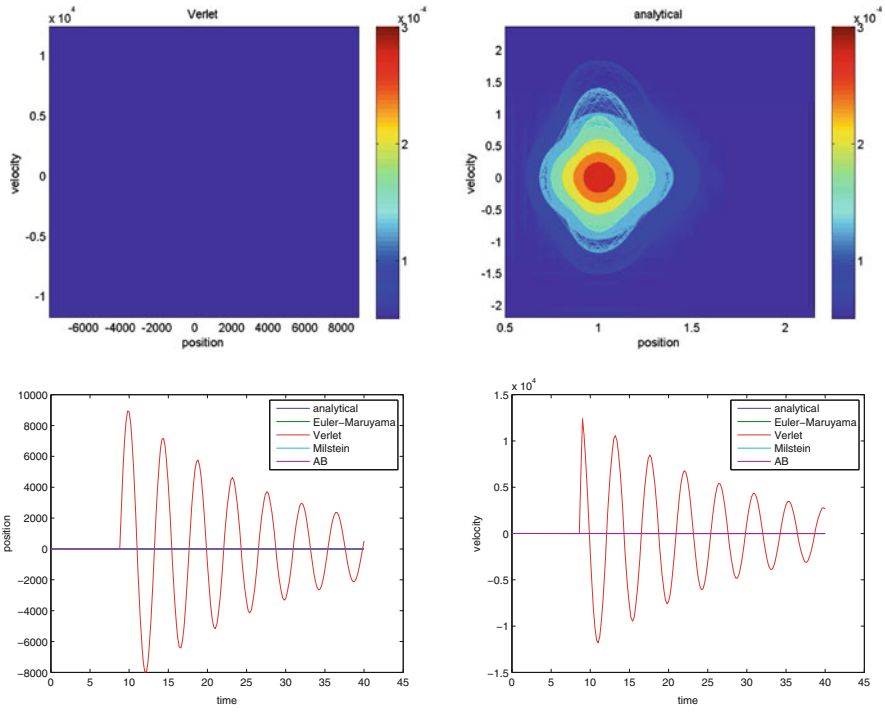



**Fig. 5.10** We apply $U(x) = \frac{1}{x^2} + x^2$, $E(x) = 2\frac{1}{x^3} - 2x$ and the starting points $(x, v)^t = (1.0, 1.0)^t$. The *upper figures* present the contours of the Hamiltonian with the Verlet algorithm (*left figure*) and the analytical algorithm (*right figure*), the *lower figures* present the $x$ (*left*) and $v$ (*right*) solutions of the Verlet algorithm, where $A = 0.1, B = 0.25$

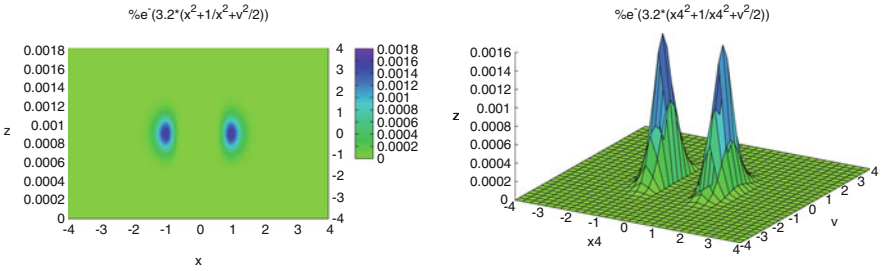



**Fig. 5.11** The distribution of the impact oscillator $U(x) = \frac{1}{x^2} + x^2$ with $\beta = 3.2$, where $A = 0.1, B = 0.25$

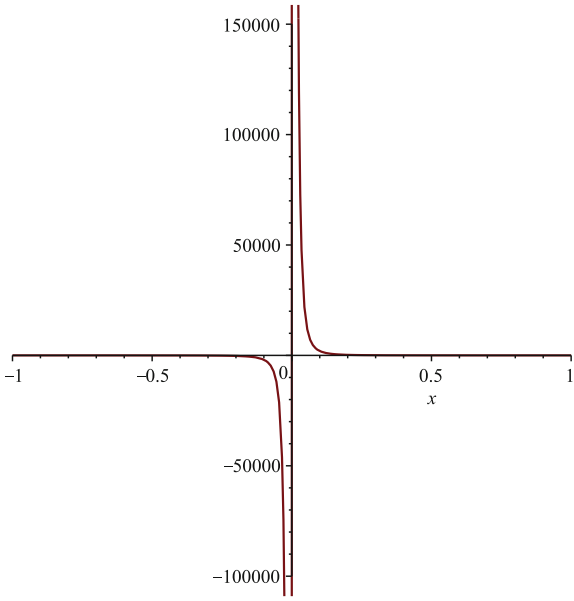



**Fig. 5.12** The graph of the fine resolution of the impact oscillator $U(x) = \frac{1}{x^2} + x^2$

$$\pi(\beta, x, v) = \exp\left(-\beta\left(\frac{v^2}{2}\right) + U(x)\right), \tag{5.70}$$

where $\beta = 3.2$ and $U(x) = \frac{1}{x^2} + x^2$.
The distribution of the impact oscillator is given in Fig. 5.11.
In the following, we see the blow up around $0.1 < x \leq 0.1$ in Fig. 5.12.

### Critical Points of the Impact Oscillator

Based on the blow up in $(x, v)^t = (0.5, 0.5)^t$ we compare the different integrators:
The impact oscillator $U(x) = \frac{1}{x^2} + x^2$, $E(x) = 2\frac{1}{x^3} - 2x$ is presented in Fig. 5.13.
We apply the CFL conditions:

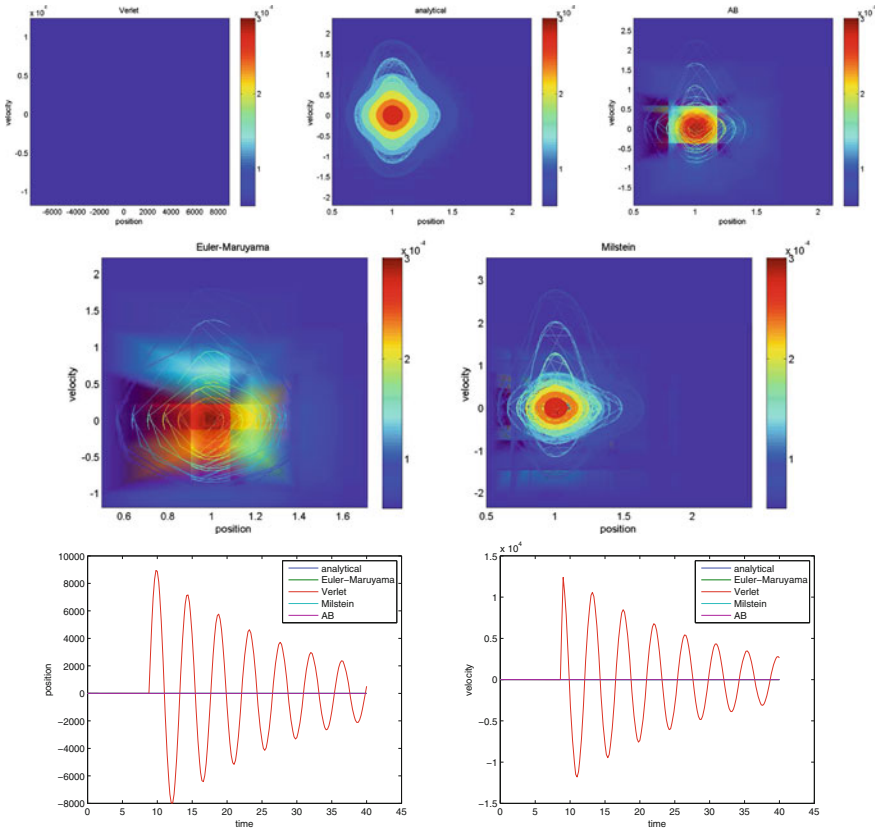



**Fig. 5.13** We apply $U(x) = \frac{1}{x^2} + x^2$, $E(x) = 2\frac{1}{x^3} - 2x$ and the starting points $(x, v)^t = (0.5, 0.5)^t$. The *upper figures* present the contours of the Hamiltonian with the Verlet algorithm (*left figure*), the analytical algorithm (*middle figure*) and the improved A–B splitting (*left figure*), the *middle figures* presents the improved EM scheme (*right figure*) and the improved Milstein scheme (*left figure*), the *lower figures* present the $x$ (*left*) and $v$ (*right*) solutions of the Verlet algorithm, where $A = 0.1, B = 0.25$

$$\Delta t \leq \sqrt{\frac{1}{\frac{1}{(|X^n|)^4} + 1}}, \tag{5.71}$$

where for small $|X^n| < 1$, we have

$$\Delta t \leq \sqrt{|X^n|}. \tag{5.72}$$

*Remark 5.6* The Verlet algorithm is only stable for dominant symplectic equations, means in our case for dominant deterministic parts. The analytical algorithm is stable also for the stochastic dominant parts.

*Remark 5.7* We see that the semi-analytical scheme resolves optimal the contours, while the Verlet algorithm could not resolve the problem. Here, we have to apply higher order schemes to take into account the singularity.

### Symplectic or Non-symplectic A–B Splitting

We present the influence of the symplecticity with respect to the harmonic oscillator at $(x, v)^t = (1.0, 1.0)^t$.

We have $U(x) = \frac{x^2}{2}$, $E(x) = -x$ is presented in Fig. 5.14.

*Remark 5.8* For a symplectic problem, it is important to improve the integrators with respect to their conservation of the symplecticity. We see the improvement of the symplectic A–B splitting scheme.

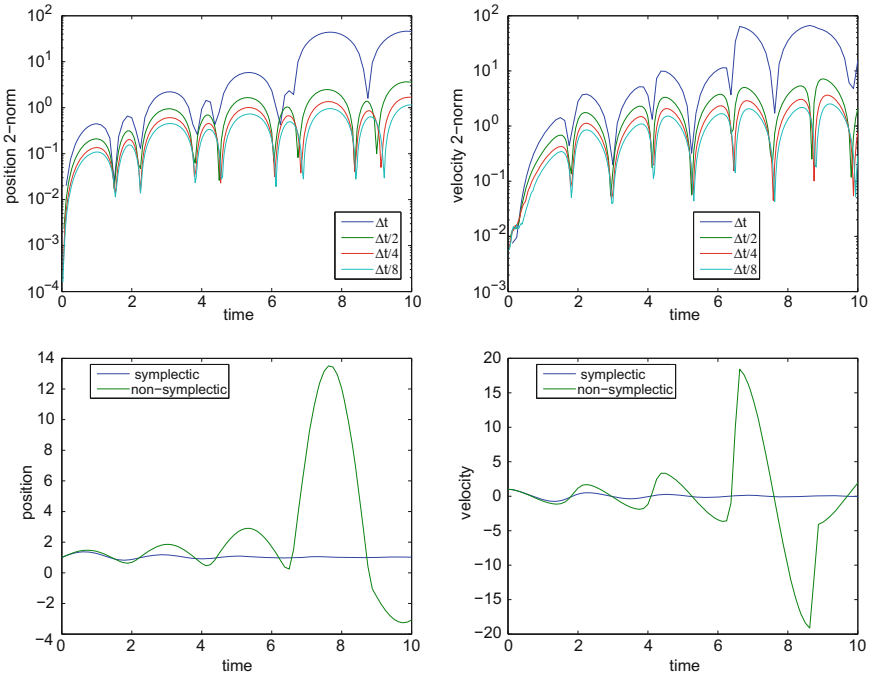



**Fig. 5.14** We apply $U(x) = \frac{x^2}{2}$, $E(x) = -x$ and the starting points $(x, v)^t = (1.0, 1.0)^t$ with the parameters $A = 0.1, B = 0.25$. The *upper figures* present the logarithmic $L_2$-error contours of the symplectic and non-symplectic A–B splitting scheme where $x$ (*left figure*) and $v$ (*right figure*) The *lower figures* present the solutions of the symplectic and non-symplectic A–B splitting scheme where $x$ (*left figure*) and $v$ (*right figure*)

### Accuracy of the Symplectic Splitting Schemes

We present the influence of the symplecticity with respect to the harmonic oscillator at $(x, v)^t = (1.0, 1.0)^t$.

We have $U(x) = \frac{x^3}{3}$, $E(x) = -x^2$ is presented in Figs. 5.15 and 5.16.

*Remark 5.9* For the accuracy of the schemes, here we tested the unharmonic oscillator (symplecticity), we achieve the best results with the higher order schemes. That means the Verlet algorithm which is of a second-order scheme is optimal for such problems.

*Remark 5.10* For all the schemes, we see the problems based on the different timescales in the impact oscillator, while the other oscillators have nearly uniform time steps. Here, we see the benefits of restricting to an adaptive version, which taken into account the CFL condition to each scheme.

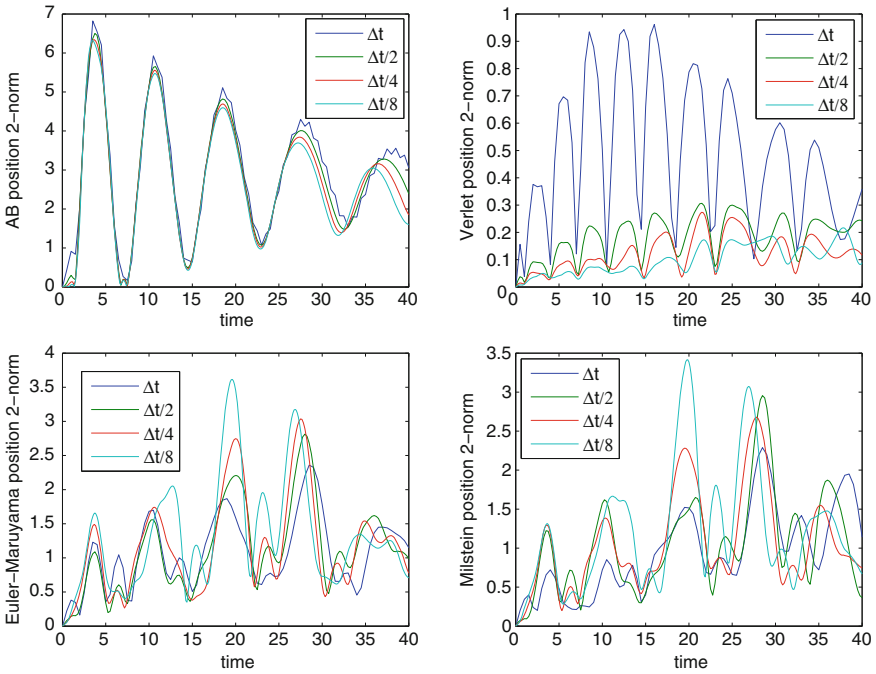



**Fig. 5.15** We apply $U(x) = \frac{x^3}{3}$, $E(x) = -x^2$ and the starting points $(x, v)^t = (1.0, 1.0)^t$ with the parameters $A = 0.1, B = 0.25$. The *upper figures* present the $L_2$-error of the position (reference is the analytical solution) where the A–B splitting scheme (*left figure*) and Verlet algorithm (*right figure*). The *lower figures* present the $L_2$-error of the position (reference is the analytical solution) where the EM scheme (*left figure*) and Milstein scheme (*right figure*)

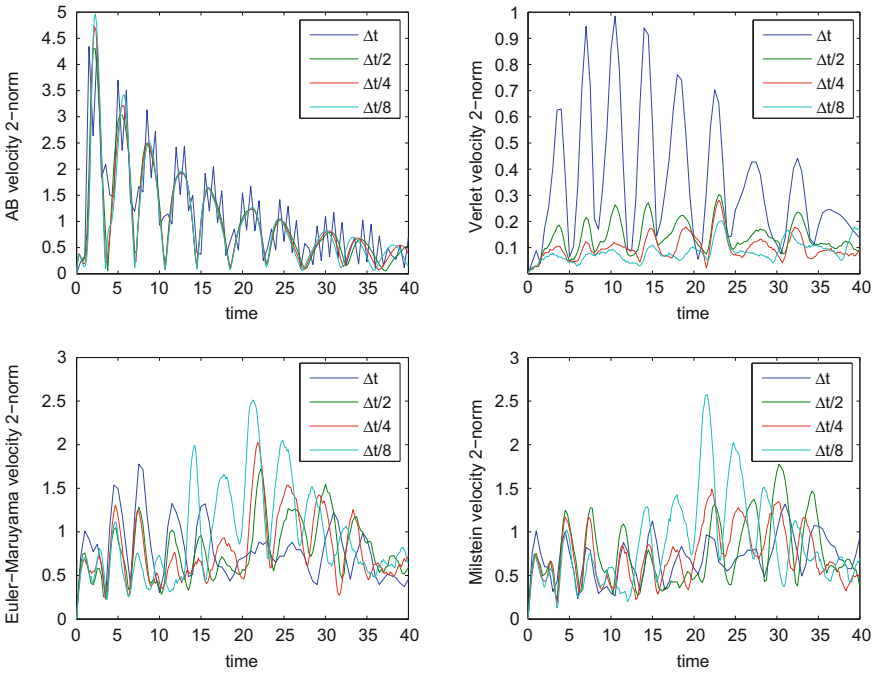



**Fig. 5.16** We apply $U(x) = \frac{x^3}{3}$, $E(x) = -x^2$ and the starting points $(x, v)^t = (1.0, 1.0)^t$ with the parameters $A = 0.1, B = 0.25$. The *upper figures* present the $L_2$-error of the velocity (reference is the analytical solution) where the A–B splitting scheme (*left figure*) and Verlet algorithm (*right figure*). The *lower figures* present the $L_2$-error of the velocity (reference is the analytical solution) where the EM scheme (*left figure*) and Milstein scheme (*right figure*)

### *5.1.9 Conclusion*

We discuss the multiscale problems of the Langevin equation, which can be solved by splitting of the deterministic and stochastic part. Here, the numerical stability of explicit integrators for stochastic differential equations (SDEs) are important to obtain long-time behaviours. Such problems arose of the stochastic part of the Langevin-like equations.

We can modify the explicit integrators by adding additional step of corrections and fulfil a long-time behaviour. We also taken into account the special potential energy functions, which has an impact and is more delicate to solve.

Real-life problems in the plasma flusion applications have such behaviours and can be studied with such modifications of the solvers.

## 5.2 Multiscale Problem in Code Coupling: Coupling Methods for the Aura Fluid Package

**Abstract** In this section, we discuss a real-life problem arose from coupling to different codes, while each code is responsible of solving a partial differential equation and we have different spatial and timescale to couple. Such a problem can also be seen as multiscale or multicomponent problem, while different scales (time and space) or different components are coupled to one *large* equation system and we have to be careful to resolve all the different scale or embed finer scale to the coarser scales, see [17, 18]. We deal with the following problem:

- Decoupled partial differential equations are solved with different codes, e.g. Aura-Solver: radiation, Fluid-solver: heat transfer.
- Each software package has their independent spatial and timescales, while the different physical problems have various scales (multiscale problem).
- A rewriting of a full coupled code is too expensive, a novel idea based on coupling the different codes, e.g. via splitting methods is important.
- We have taken into account the coupling of the boundary conditions and coupling via the initial problems, see [19, 20].

### *5.2.1 Introduction*

The Aura Fluid software package is known as a heat transfer and radiation software for large-scale computations of heat and radiation problems, see [21]. The model problem is to simulate the influence of solar heat in car bodies, while radiation and heat flow transfer is important.

So we deal with two software packages and two physical problems:

- Aura Software: Heat transfer and Radiation Model (modelling the heat transfer: fine scales),
- Fluid Software: Flow field (modelling the flow field of the problem: coarse scales).

Based on that huge amount of computational time, if we directly couple such codes, we have taken into account a fine and coarse scale computation.

Here, we deal with a splitting idea in the following manner, see Fig. 5.17:

Such a coupling can be done by operator splitting methods and also with parallel interfaces. We discuss the idea of applying splitting methods with Parareal, see [22] and [21], which is a parallel method for time-parallelization problems. We can apply such ideas for parabolic problems and as a speedup for such problems. We combine the splitting schemes as a time-splitting method and accelerate the large time intervals with Parareal as a time-parallelization method.

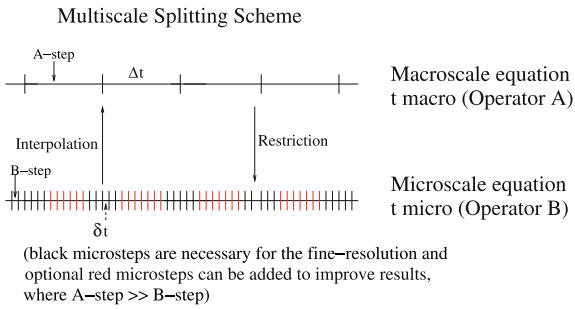



**Fig. 5.17** Multiscale A–B Splitting scheme

## *5.2.2 Mathematical Model*

In the following, we have delicate models of coupled heat transfer and radiation models. While the heat transfer and radiation is solved with the Aura software package, the flow field of the temperature is done with a flow-field solver, e.g. Openfoam or Vectis, see [23, 24]. The idea is to obtain a speedup with fast coupling schemes and parallel time schemes.

We deal with the following equations, which are simulated with the different software packages:

1. Heat equation:

$$\begin{aligned} \partial_t\, T(x,t) &= \mathscr{H}(T(x,t)) \\ &= \mathscr{A}T(x,t) + \mathscr{B}(T(x,t)),\ (x,t) \in \Omega \times [0,T], \qquad (5.73) \\ T(x,0) &= T_0(x),\ x \in \Omega, \qquad (5.74) \end{aligned}$$

   where the unknown temperature is $T$, $\mathscr{A}$ is the heat transfer operator and $\mathscr{B}$ the nonlinear boundary operator and we assume that the full operator $\mathscr{H}$ can be decoupled into the two operators. $T_0(x)$ is the initial temperature.
2. Flow-field equation:

$$\begin{aligned} \partial_t\, \alpha_{fl} &= \mathscr{C}(\alpha_{fl}),\ (x,t) \in \Omega \times [0,T], \qquad (5.75) \\ \alpha_{fl}(x,0) &= \alpha_{fl,0}(x), x \in \Omega, \qquad (5.76) \end{aligned}$$

   where $\mathscr{C}$ is the nonlinear flow operator and $\alpha_{fl}$ the unknown flow field. $\alpha_{fl,0}(x)$ is the initial flux.
3. Coupling of the fluid- and heat transfer equation is done by the boundary condition to the heat radiation equation:

$$\mathscr{B}(T(x,t)) = -\lambda \frac{\partial}{\partial n} T(x,t) = \alpha_{fl}(x,t)\,(T(x,t) - T_{fl}(x,t)) + q_{\text{rad}}(T,x), \qquad (5.77)$$

for all $x \in \partial\Omega$, $q_{\text{rad}}(T, x)$ is the heat transfer of the radiation.
Further $T_{fl}(x, t)$ is a given temperature of the heat transfer, e.g. temperature of the initialization.

*Remark 5.11* The modelling problem was motivated by a realistic engineering problem in simulating warming in the vehicle interior, when we have a worst-case scenario, that the cooling system is malfunctioned, see [25] and [2]. Here, we deal with different scales, based on the flow and radiation model, that have to be coupled via a boundary condition, between the radiation and flow domains, see [21].

### 5.2.3 Splitting Methods

The different codes are coupled by splitting methods that are discussed in the following:

We assume that the boundary conditions are embedded into the spatial discretized matrices and that we deal with ordinary differential equations, see [26].

1. Lie–Trotter or A–B Splitting method
   The standard implemented scheme is the well-known A–B splitting method.
   A–B splitting:

$$\begin{aligned} &\frac{\partial T(t)}{\partial t} = H(T(t), t), \\ &\text{with } t^n \le t \le t^{n+1},\ T(t_n) = c^n, \end{aligned} \tag{5.78}$$

$$\begin{aligned} &\frac{\partial \alpha_{fl}(t)}{\partial t} = C(\alpha_{fl}(t)), \\ &\text{with } t^n \le t \le t^{n+1},\ \alpha_{fl}(t_n) = T(t^{n+1}), \end{aligned} \tag{5.79}$$

   where $c^n$ is the known initial value of the previous solution and $c(t^{n+1}) = \alpha_{fl}(t^{n+1})$ is the approximated solution of the full equation.
   We have a global splitting error of $O(\Delta t)$, where $\Delta t$ is the time step.
2. Strang Splitting method
   The standard Strang Splitting is given in the following 3-step algorithm:

$$\begin{aligned} &\frac{\partial T(t)}{\partial t} = H(T(t), t), \\ &\text{with } t^n \le t \le t^{n+1/2},\ T(t_n) = c^n, \end{aligned} \tag{5.80}$$

$$\begin{aligned} &\frac{\partial \alpha_{fl}(t)}{\partial t} = C(\alpha_{fl}(t)), \\ &\text{with } t^n \le t \le t^{n+1},\ \alpha_{fl}(t_n) = T(t^{n+1/2}), \end{aligned} \tag{5.81}$$

$$\begin{aligned} &\frac{\partial T(t)}{\partial t} = H(T(t), t), \\ &\text{with } t^{n+1/2} \le t \le t^{n+1},\ T(t_{n+1/2}) = \alpha_{fl}(t^{n+1}), \end{aligned} \tag{5.82}$$

where $c^n$ is the known initial value of the previous solution and $c(t^{n+1}) = \alpha_{fl}(t^{n+1})$ is the approximated solution of the full equation.
Here we achieve a coupling method, which is one order higher than the previous one. We obtain $O(\Delta t^2)$.

*Remark 5.12* With such improved methods, we obtain higher accuracy and faster computations.

3. Non-iterative splitting method: Richardson Extrapolation
We deal with the following semi-discretized method. Our operators are derived by space-discretization methods.
The considered systems of ordinary differential equations are given as:

$$\begin{aligned} &u_t + (A_1 + A_2)u = 0, \\ &u(0) = u_0, \ \text{(initial condition)}. \end{aligned} \tag{5.83}$$

The fourth-order splitting method based on the Richardson extrapolation, as discussed in [27, 28], is given as

$$D_4(\Delta t) = 4/3\ S_2(\Delta t/2)\ S_2(\Delta t/2) - 1/3\ S_2(\Delta t), \tag{5.84}$$

where $S_2(\Delta t) = \exp(A_2\Delta t)\exp(A_1 2\Delta t)\exp(A_2\Delta t)$ is the Strang splitting operator [29].
The higher order is reached after applying three times the Strang splitting method in a proper way.
The fifth-order splitting method based on the Richardson extrapolation, as discussed in [27, 28], is given as:

$$D_5(\Delta t) = 16/15\ S_4(\Delta t/2)\ S_4(\Delta t/2) - 1/15\ S_4(\Delta t), \tag{5.85}$$

4. Iterative splitting method
The following algorithm is based on the iteration with fixed splitting discretization step size $\tau$. On the time interval $[t^n, t^{n+1}]$ we solve the following sub-problems consecutively for $i = 0, 2, \ldots 2m$.
The iterative method is given as, see also [30],

$$\begin{aligned} &\frac{\partial c_i(t)}{\partial t} = Ac_i(t) \ + \ Bc_{i-1}(t), \\ &\text{with } c_i(t^n) = c^n, \ c_0(t^n) = c^n, \ c_{-1} = 0.0, \\ &\text{and } c_i(t) = c_{i-1}(t) = c_1, \ \text{on } (0, T), \end{aligned} \tag{5.86}$$

$$\begin{aligned} &\frac{\partial c_{i+1}(t)}{\partial t} = \ Ac_i(t) \ + \ Bc_{i+1}(t), \\ &\text{with } c_{i+1}(t^n) = c^n \\ &\text{and } c_i(t) = c_{i-1}(t) = c_1, \ \text{on } (0, T), \end{aligned} \tag{5.87}$$

where $c^n$ is the known split approximation at time level $t = t^n$ [1].
The higher order is obtained by applying recursively the fixed-point iteration to reconstruct the analytical solution of the coupled operators, see [31].

*Remark 5.13* If we do not iterative over the several operators, means we have only $i = 1$, we define a 0-iterative scheme. Such a scheme is highly parallel, but is not stable, while the iteration process is not finished. We obtain only a 0 order scheme.

The improved iterative schemes deal with $i = 2, 3, 4$ steps and we stop after we reach a final criterion:

$$||c_i(t) - c_{i-1}|| \leq err, \tag{5.88}$$

$err \in \mathbb{R}^+$ is a given error tolerance e.g. $10^{-4}$.

In the real-life application, the computation of the operator $B$ is delicate and the exchange is only forward. Means, we could only apply initial and boundary conditions to the software code and the results of the computations. An intermediate exchange like in the Strang Splitting $ABA$ is impossible only ideas like $ABB$ or $AAB$ are possible. Therefore, we have to develop some new modified splitting schemes.

Such one-step ideas are discussed in the next subsection.

### 5.2.4 Modified A–B Splitting Method: Only One Exchange to Operator B

We assume to be restricted by the operator $B$, means we can only change one time in a cycle to operator $B$, but we cannot change back to operator $A$.

That means, we have to deal with splitting ideas of the form: $AB$, $AAB$, $ABB$, etc., means we are restricted to one-sided splitting schemes and assume that also the commutator deal with such a behaviour, see idea of a fourth-order scheme, which assume some special behaviour of the commutators [32].

The idea is to apply $t/2$ time steps with respect to operator $A$ and $B$, to start with operator $A$, but restricted to change only one time to operator $B$.

The idea of the algorithm with the restriction of the evaluation of operator $B$ is given in Fig. 5.18:

The following algorithm is based on the modified A–B Splitting:

$$\tilde{c}_1(t) = \exp(At/2)\exp(Bt/2)\exp(Bt/2)c(t^n), \tag{5.89}$$
$$\tilde{c}_2(t) = \exp(At/2)\exp(At/2)\exp(Bt/2)c(t^n), \tag{5.90}$$
$$\tilde{c}(t) = 2/3\tilde{c}_1(t) + 2/3\tilde{c}_2(t) - 1/3c(t^n), \tag{5.91}$$
$$t \in [t^n, t^{n+1}], \tag{5.92}$$

where $c(t^n) = c^n$ is the known split approximation at time level $t = t^n$ and $\tilde{c}(t)$ is the next approximated solution to time $t$.

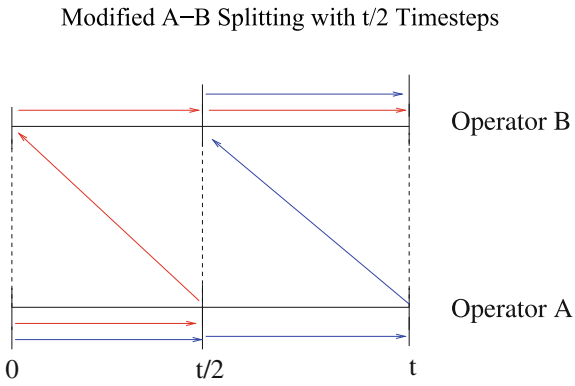



**Fig. 5.18** Modified A–B splitting scheme, means *AAB* and *ABB* evaluations

**Theorem 5.3** *The modified A–B splitting scheme, given in Fig. 5.18, has the order* $O(\tau)$.

*Proof* We have

$$\tilde{c}(t) = a\tilde{c_1}(t) + b\tilde{c_2}(t) + cc(t^n), \tag{5.93}$$

where $\tilde{c}_1(t)$ is given in (5.89) and $\tilde{c}_2(t)$ is given in (5.90), $a, b, c$ are real numbers and $t = t^{n+1} - t^n$ is the time step, with the time-discretization $t^n,\ n = 0, \ldots, N$ and $t^{N+1} = T$.

The exact solution of the ODE is given as

$$c(t) = \exp((A + B)t)c(t^n). \tag{5.94}$$

We deal with the following error:

$$\begin{aligned} ||c(t) - \tilde{c}(t)|| \leq\ & 1 + (A + B)t + t^2/2(A + B)^2 \\ & - a(1 + A/2\ t + A^2/4\ t^2/2)(1 + B/2\ t + B^2/4\ t^2/2)(1 + B/2\ t + B^2/4\ t^2/2) \\ & - b(1 + A/2\ t + A^2/4\ t^2/2)(1 + A/2\ t + A^2/4\ t^2/2)(1 + B/2\ t + B^2/4\ t^2/2) \\ & - c + O((t/2)^3). \end{aligned} \tag{5.95}$$

Further by comparison of the coefficients we obtain for the first order, while the reconstruction of higher terms is not possible:

$a = 2/3, b = 2/3, c = -1/3$.

We have a second-order term given as $\frac{1}{6}||A^2 + 2(AB + BA) + B^2||$, while the A–B splitting has an term belonging to the commutator $||[A, B]|| = ||AB - BA||$. So, we benefit of the idea $||(A + B)^2|| \leq 5[A, B]$, means if the commutator is dominant in the scheme, we might have some reduction of the local error.

For higher orders it is impossible to skip the terms and without stepping again to $A$ as, for example with Strang- or Iterative-splitting it is impossible to obtain a higher order.

*Remark 5.14* With the modified A–B splitting, we gain locally benefits and could reduce locally the error for the scheme. At least, we have an A–B splitting of a global order 1. The benefit is given with the reduction, if we deal with a very large commutator error, then it might sense and the modified version is of lower local error.

### 5.2.5 Coupling of Initial Dates and Multiscale Approach

The initialization is very important for the splitting schemes and also the connection via the boundary conditions.

While we deal with separate software code, which compute the different model equations, an update in the starting procedure is important to synchronize the processes. On the one hand, we have a macroscopic model (heat transfer model), where we can apply large time and spatial steps, while on the other hand, we have a microscopic model (radiation model), where we have taken into account the small time and spatial steps. A efficient coupling, while each code can be applied independently, is done via the Strang splitting, where we couple the models via their initial conditions, means the solution, that is necessary for the next time step. Here we apply the macroscopic step, while the synchronization step (embedding of the microsteps) is done with the small time steps and we couple via an iterative scheme (e.g. successive approximation or fixpoint scheme) the two models.

In the following, we explain the coupling of the Aura and Foam code. At the beginning, we have to exchange their initial values at $t = 0$. Such that the starting step is important and synchronize via iterative steps the starting conditions of the codes, see Fig. 5.19. The next steps is based on the macroscopic time step and we apply the next time frame of the simulation, e.g. from $t = 0$ to $t = 1.0$ (where the time step $\Delta t = 1.0$ [min]). Then we apply the synchronization or microscopic time step to embed the microscopic model (where the time step $\Delta_{micro} = 1.0$ [s]) and we apply, for example $N_{micro} = 10$–$15$ microscopic time steps and iterate the models via a fixpoint scheme, see Fig. 5.19.

*Remark 5.15* The benefit of such an initial value coupling is independent codes, while the microscopic solver is based on an iterative solver, that couples the models only between the intermediate solutions. Such a weakly coupling allows to speed up the codes and only interfaces are needed to couple the codes.

In the next subsection, we deal with the error estimates.

### 5.2.6 Error Estimates

For the coupling of such micro- and macroscopic models by a splitting scheme, we have to be sure about the underlying splitting error and control them in the simulations.

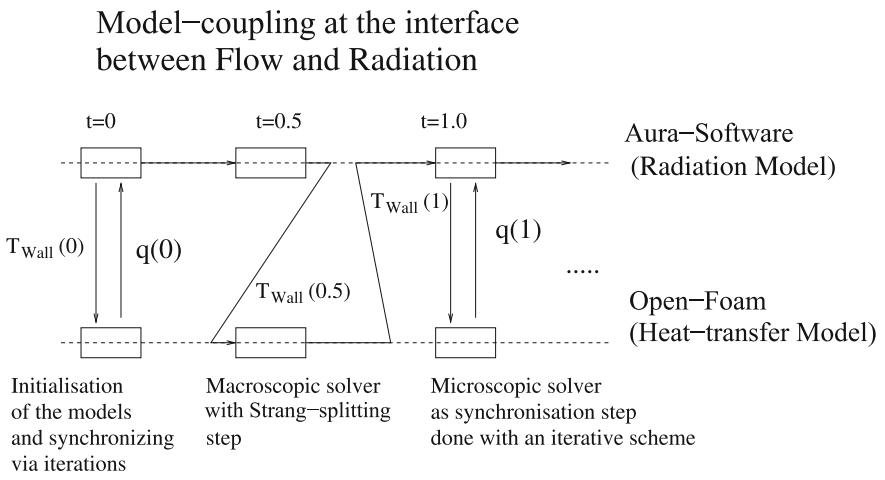



**Fig. 5.19** Initialization, solver-steps, coupling and synchronization of the codes

Here, we deal with the two error estimates of the weakly coupled models:

- A priori error estimate (An assumed error, which can be estimated before the computations),
- A posteriori error estimate (a corrected error, which can be computed after the computations or compared with a grid resolution).

While the a priori error estimates give a time step before the computation, the a posteriori error estimates give an optimal time step after the computation.

### *5.2.7 A Priori Error Estimates for the Splitting Scheme*

We deal with the two software codes that are coupled via the ODE.

We have the following operators (related to the program codes)

- A: Aura Program,
- B: Openfoam.

The A–B Splitting error estimates is given as

$$\tau_n \leq= \frac{1}{\frac{1}{2}||[A,B]||}, \tag{5.96}$$

We compute the commutator $[A, B]$. In practice, we compute one time step with Aura-Openfoam and one time step with Openfoam-Aura, based on this we con optimize the time step.

We assume we have the result previously.

The Strang Splitting error estimates is given as

$$\tau_n \leq= \frac{1}{\frac{1}{24}||([B,[B,A]]-2[A,[A,B]])||}, \tag{5.97}$$

we compute the commutator $[A,[A,B]]$ and $[B,[B,A]]$. In practice, we compute one time step with 1/2 Aura—Openfoam—1/2 Aura and with 1/2 Openfoam-Aura—1/2 Openfoam. The inverse of the differences are the optimal time step.

### 5.2.8 A Posteriori Error Estimates for the Splitting Scheme

We define a next error estimator to compare and reduce the given error in one cycle of the computation.

To have an a posteriori error estimates, means an error after the computation, we deal with splitting methods of different orders and compare their results.

Based on the relative error between the methods, we could define a next error bound to decide, if we should reduce the time step for our computations.

While the A–B splitting method, has a time step of $\Delta t$, the Strang splitting halfen the time step to $\Delta t/2$, so we have one order of accuracy more with the Strang Splitting.

We define the following a posteriori error:
The maximal error at time $t$ is given as

$$err_{\max,\Delta t} = |c_{Strang} - c_{A-B}|_{\max} = \max_{j=1}^{p} |c_{Strang}(x_j, y_j, t) - c_{A-B}(x_j, y_j, t)|,$$

the numerical convergence rate is given as

$$\rho_{\max} = \log(err_{\max,\Delta t/2}/err_{\max,\Delta t})/\log(0.5).$$

The $L_1$ error at time $t$ is given as

$$err_{L_1,\Delta t} = |c_{Strang} - c_{A-B}|_{L_1} = \sum_{j=1}^{p} \Delta t \ |c_{Strang}(x_j, y_j, t) - c_{A-B}(x_j, y_j, t)|,$$

the numerical convergence rate is given as

$$\rho_{L_1} = \log(err_{L_1,\Delta t/2}/err_{L_1,\Delta t})/\log(0.5),$$

where $\Delta t$ and $\Delta/2$ are time steps.

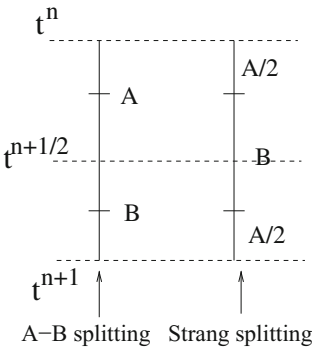



**Fig. 5.20** Error estimates for the splitting scheme

The $L_2$ error at time $t$ is given as

$$err_{\mathrm{L}_2,\Delta \mathrm{t}} = |c_{Strang} - c_{A-B}|_{L_2} = \sqrt{\sum_{j=1}^{p} \Delta t\,(u_{Strang}(x_j, y_j, t) - u_{A-B}(x_j, y_j, t))^2},$$

the numerical convergence rate is given as

$$\rho_{L_2} = \log(err_{\mathrm{L}_2,\Delta \mathrm{t}/2}/err_{\mathrm{L}_2,\Delta \mathrm{t}})/\log(0.5),$$

where $\Delta t$ and $\Delta/2$ are time steps.

Figure 5.20 presents error estimates of the different splitting schemes in one cycle.

*Remark 5.16* Here, we can control the error of the different methods and if the error bound is not fulfilled, we redo the computations with a smaller time step. So, we also obtain with different time steps $\Delta t$, $\Delta t/2$, $\Delta t/4$, ... the numerical errors between the scheme and the resulting convergence rate. Here, we could switch on and off the higher order scheme, if the error bound and convergence rates are given. At least, we should have one order more, when comparing the two methods.

### 5.2.9 *Optimization for the Heat- and Radiation Equation: Newton's Method for Solving the Fixpoint Problem*

We could optimize the solver process of the multiscale problem by including more accurate solvers with respect to the underling nonlinear problem of the equations.

A nonlinear solver reduce the approximation error of the previous applied simpler linear methods, see [33].

The Heat- and radiation equation is given as

$$\rho c \partial_t\, T = \nabla \lambda \cdot \nabla T, \tag{5.98}$$

$$\frac{\partial T}{\partial n}(s) = (T - T_f)s + q_{\text{rad}}, \tag{5.99}$$

where $q_{\text{rad}} = (I - \rho K)^{-1} T^4(s)$ is the radiation term.

The spatial discretized equations are given as

$$\partial_t \, T = A_\Omega T + B_{\partial\Omega}(T^4), \tag{5.100}$$

then we apply implicit Euler time-discretization and we obtain a nonlinear ODE system

$$(M - \Delta t \, A)T_{n+1} = f_n + B_{\partial\Omega}(D(T)T_{n+1}) \tag{5.101}$$

where for the $D(T) = T^3$, while we can choose and explicit or implicit version:

1. $D(T_n) = T_n^3$ and we obtain a quasi Newton's method (explicit version),
2. $D(T_{n+1}) = T_{n+1}^3$ and we obtain a Newton's method (implicit version).

In the following we discuss the iterative splitting scheme with embedded Newton's method, that couples the micro- and macroscopic model in our multiscale approximation.

### 5.2.10 The Modified Jacobian Newton Methods and Fixpoint Iteration Methods

In this section we describe the modified Jacobian Newton methods and fixpoint iteration methods. We propose for weak nonlinearities, e.g. quadratic nonlinearity, the fixpoint iteration method. We apply the iterative operator splitting method as an example for a fixpoint method, see [34]. For stronger nonlinearities, e.g. cubic or higher order polynomial nonlinearities, the iterative splitting methods with embedded modified Jacobian Newton method can be adopted. The benefit from embedding Newton's method into the splitting methods is to decouple the equation system into simpler equations, which can be solved with the scalar Newton methods.

**The Sequential Splitting Method with Embedded Altered Jacobian Newton Iterative Method**

We restrict our attention to time-dependent operator differential equations of the form

$$\frac{dc}{dt} = A(c(t))c(t) + B(c(t))c(t), \quad \text{with } c(t^n) = c^n, \tag{5.102}$$

where $A(c), B(c)$ are matrices, whose entries are dependent on the solution $c = (c_1, \ldots, c_m)^t$, and $m$ is the number of spatial discretization points. We assume the

operators to be linear and densely defined in the real Banach space $\mathbf{X}$ and that they are obtained only from spatial derivatives of $c$, see [35]. We assume also that we have weak nonlinear operators which can be bounded with respect to some norms, e.g. $||A(c)c|| \leq \lambda_1||c||$ and $||B(c)c|| \leq \lambda_2||c||$, where $\lambda_1$ and $\lambda_2$ are constant factors.

In the following we discuss the embedding of Newton's method into the sequential splitting method. In general, in order to solve an equation of the form $F(c) = \frac{dc}{dt} - A(c(t))c(t) - B(c(t))c(t) = 0$ we can apply Newton's method and compute $c^{(k+1)} = c^{(k)} - D(F(c^{(k)}))^{-1}F(c^{(k)})$, where $D(F(c))$ is the Jacobian matrix and $k = 0, 1, \ldots$. We stop the iterations when we obtain $|c^{(k+1)} - c^{(k)}| \leq err$, with $err$ being a sufficiently small error bound, e.g. $err = 10^{-4}$.

We assume the spatial discretization, with $m$ spatial grid points, and obtain the differential equation system

$$F(c) = \begin{pmatrix} F(c_1) \\ F(c_2) \\ \vdots \\ F(c_m) \end{pmatrix}. \tag{5.103}$$

The Jacobian matrix for this system is given as

$$DF(c) = \begin{pmatrix} \frac{\partial F(c_1)}{\partial c_1} & \frac{\partial F(c_1)}{\partial c_2} & \cdots & \frac{\partial F(c_1)}{\partial c_m} \\ \frac{\partial F(c_2)}{\partial c_1} & \frac{\partial F(c_2)}{\partial c_2} & \cdots & \frac{\partial F(c_2)}{\partial c_m} \\ \vdots & & & \\ \frac{\partial F(c_m)}{\partial c_1} & \frac{\partial F(c_m)}{\partial c_2} & \cdots & \frac{\partial F(c_m)}{\partial c_m} \end{pmatrix}.$$

The modified Jacobian is

$$DF(c) = \begin{pmatrix} \frac{\partial F(c_1)}{\partial c_1} + F(c_1) & \frac{\partial F(c_1)}{\partial c_2} & \cdots & \frac{\partial F(c_1)}{\partial c_m} \\ \frac{\partial F(c_2)}{\partial c_1} & \frac{\partial F(c_2)}{\partial c_2} + F(c_2) & \cdots & \frac{\partial F(c_2)}{\partial c_m} \\ \vdots & & & \\ \frac{\partial F(c_m)}{\partial c_1} & \frac{\partial F(c_m)}{\partial c_2} & \cdots & \frac{\partial F(c_m)}{\partial c_m} + F(c_m) \end{pmatrix}.$$

*Remark 5.17* For an ordinary differential equation, we have at least one scalar entry in the Jacobian matrix, while for the assumed spatial discretized PDEs, we deal with the above defined Jacobian matrix.

By considering the sequential splitting method we obtain the following algorithm. We decouple Eq. (5.102) into two equation systems

$$F_1(u_1) = \partial_t u_1 - A(u_1)u_1 = 0 \quad \text{with} \quad u_1(t^n) = c^n, \tag{5.104}$$

$$F_2(u_2) = \partial_t u_2 - B(u_2)u_2 = 0 \quad \text{with} \quad u_2(t^n) = u_1(t^{n+1}), \tag{5.105}$$

where the results of the methods are given by $u_2(t^{n+1})$ and $u_1 = (u_{11}, \ldots, u_{1m})$, $u_2 = (u_{21}, \ldots, u_{2m})$.

Thus we have to apply Newton's method twice, each in one equation system. Here, the contribution is the reduction of the Jacobian matrix with outer-diagonal entries, into an approximated Jacobian matrix with less or without outer-diagonal entries, e.g. with a weighted Newton method, see [33]. The splitting method with embedded Newton's method is given for the continuous method as

$$\begin{aligned}
&u_1^{(k+1)} = u_1^{(k)} - D\left(F_1\left(u_1^{(k)}\right)\right)^{-1}\left(\partial_t u_1^{(k)} - A\left(u_1^{(k)}\right)u_1^{(k)}\right),\\
&\text{with } D\left(F_1\left(u_1^{(k)}\right)\right) = \frac{\partial}{\partial u_1^{(k)}}\left(\partial_t u_1^{(k)} - A\left(u_1^{(k)}\right) - \frac{\partial A\left(u_1^{(k)}\right)}{\partial u_1^{(k)}}u_1^{(k)}\right),\\
&u_1^{(k)}(t^n) = c^n \text{ and } k = 0, 1, 2, \ldots, K,
\end{aligned} \tag{5.106}$$

$$\begin{aligned}
&u_2^{(l+1)} = u_2^{(l)} - D\left(F_2\left(u_2^{(l)}\right)\right)^{-1}\left(\partial_t u_2^{(l)} - B\left(u_2^{(l)}\right)u_2^{(l)}\right),\\
&\text{with } D\left(F_2\left(u_2^{(l)}\right)\right) = \frac{\partial}{\partial u_1^{(k)}}\left(\partial_t u_2^{(k)} - B\left(u_2^{(l)}\right) - \frac{\partial B\left(u_2^{(l)}\right)}{\partial u_2^{(l)}}u_2^{(l)}\right),\\
&u_2^{(l)}(t^n) = u_1^K(t^{n+1}) \text{ and } l = 0, 1, 2, \ldots, L.
\end{aligned} \tag{5.107}$$

For an improvement, we can apply the weighted Newton's method. We try to skip the delicate outer diagonals in the Jacobian matrix and apply

$$u_1^{(k+1)} = u_1^{(k)} - \left(D\left(F_1\left(u_1^{(k)}\right)\right) + \delta_1\left(u_1^{(k)}\right)\right)^{-1}\left(F_1\left(u_1^{(k)}\right) + \varepsilon\, u_1^{(k)}\right), \tag{5.108}$$

where the function $\delta$ can be applied as a scalar, e.g. $\delta = 10^{-6}$, also the same with $\varepsilon$. It is important to be sure that $\delta$ is small enough to preserve the convergence.

Discretizing Eqs. (5.104) and (5.105) with Backward Euler method leads to

$$\begin{aligned}
&F_1(u_1(t^{n+1})) = u_1(t^{n+1}) - u_1(t^n) - \Delta t A(u_1(t^{n+1}))u_1(t^{n+1}) = 0\\
&\text{with} \quad u_1(t^n) = c^n,
\end{aligned} \tag{5.109}$$

$$\begin{aligned}
&F_2(u_2(t^{n+1}) = u_2(t^{n+1}) - u_2(t^n) - \Delta t B(u_2(t^{n+1}))u_2(t^{n+1}) = 0\\
&\text{with} \quad u_2(t^n) = u_1(t^{n+1}),
\end{aligned} \tag{5.110}$$

then we obtain the derivations $D(F_1(u_1(t^{n+1})))$ and $D(F_2(u_2(t^{n+1})))$ which are given as

$$D(F_1(u_1(t^{n+1}))) = I - \Delta t(A(u_1(t^{n+1})) + \frac{\partial A(u_1(t^{n+1}))}{\partial u_1(t^{n+1})} u_1(t^{n+1})), \tag{5.111}$$

$$D(F_2(u_2(t^{n+1})) = I - \Delta t(B(u_2(t^{n+1})) + \frac{\partial B(u_2(t^{n+1}))}{\partial u_2(t^{n+1})} u_2(t^{n+1})), \tag{5.112}$$

where we have the vectorial solutions $u_1(t^{n+1}) = (u_{11}(t^{n+1}) \ldots, u_{1m}(t^{n+1}))^t$ and $u_2(t^{n+1}) = (u_{21}(t^{n+1}) \ldots, u_{2m}(t^{n+1}))^t$.

The matrix $A(u_1(t^{n+1}))$ is defined by

$$A(u_1(t^{n+1})) = \begin{pmatrix} A_{11}(u_1(t^{n+1})) & A_{12}(u_1(t^{n+1})) & \ldots & A_{1m}(u_1(t^{n+1})) \\ A_{21}(u_1(t^{n+1})) & A_{22}(u_1(t^{n+1})) & \ldots & A_{2m}(u_1(t^{n+1})) \\ \vdots & \vdots & \ddots & \vdots \\ A_{m1}(u_1(t^{n+1})) & A_{m2}(u_1(t^{n+1})) & \ldots & A_{mm}(u_1(t^{n+1})) \end{pmatrix},$$

where we have the functions $A_{11}, \ldots, A_{mm} : \mathbb{R}^n \to \mathbb{R}$.

We obtain the derivations of the matrix as

$$\frac{\partial A(u_1(t^{n+1}))}{\partial u_1(t^{n+1})} u_1(t^{n+1}) = \begin{pmatrix} \frac{\partial A_{11}(u_1)}{\partial u_{11}} u_{11} & \frac{\partial A_{12}(u_1)}{\partial u_{12}} u_{12} & \ldots & \frac{\partial A_{1m}(u_1)}{\partial u_{1m}} u_{1m} \\ \frac{\partial A_{21}(u_1)}{\partial u_{11}} u_{11} & \frac{\partial A_{22}(u_1)}{\partial u_{12}} u_{12} & \ldots & \frac{\partial A_{2m}(u_1)}{\partial u_{1m}} u_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial A_{m1}(u_1)}{\partial u_{11}} u_{11} & \frac{\partial A_{m1}(u_1)}{\partial u_{12}} u_{12} & \ldots & \frac{\partial A_{mm}(u_1)}{\partial u_{1m}} u_{1m} \end{pmatrix}.$$

where $u_1(t^{n+1}) = (u_{11}(t^{n+1}), \ldots, u_{1m}(t^{n+1}))^t$ and have derived the derivatives over this vector.

The same structure of matrices can be also obtained for $B(u_2(t^{n+1}))$. For the scalar case $u_1(t^{n+1}) = u_{11}(t^{n+1})$ and we obtain only a scalar Jacobian, the same also for $u_2(t^{n+1})$. Equation (5.108) is applied analogously for $u_2^{(l+1)}$.

**Iterative Operator Splitting Method as Fixpoint Scheme**

The iterative operator splitting method is used as a fixpoint scheme to linearize the nonlinear operators, see [31, 34].

We restrict our attention again to time-dependent partial differential equations of the form (5.102). $A(u)$, $B(u)$ are matrices with nonlinear entries and densely defined, where we assume that the entries involve the spatial derivatives of $c$, see [35]. In the

following we discuss the standard iterative operator splitting method as a fixpoint iteration method to linearize the operators.

We split our nonlinear differential equation (5.102) by applying

$$\frac{du_i(t)}{dt} = A(u_{i-1}(t))u_i(t) + B(u_{i-1}(t))u_{i-1}(t), \text{ with } u_i(t^n) = c^n,$$
$$\frac{du_{i+1}(t)}{dt} = A(u_{i-1}(t))u_i(t) + B(u_{i-1}(t))u_{i+1}(t), \text{ with } u_{i+1}(t^n) = c^n,$$

where the time step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m + 1$. $u_0(t) = c_n$ is the starting solution, where we assume that the solution $c^{n+1}$ is near $c^n$, or $u_0(t) = 0$. So we have to solve the local fixpoint problem. $c^n$ is the known split approximation at the time level $t = t^n$.

The split approximation at time level $t = t^{n+1}$ is defined as $c^{n+1} = u_{2m+2}(t^{n+1})$. We assume that the operators $A(u_{i-1}(t^{n+1}))$, $B(u_{i-1}(t^{n+1}))$ are constant defined for $i = 1, 3, \ldots, 2m+1$. Here the linearization is done with respect to the iterations, such that $A(u_{i-1})$, $B(u_{i-1})$ are at least non-dependent operators in the iterative equations, and we can apply the linear theory. For the linearization we assume at least in the first equation $A(u_{i-1}(t)) \approx A(u_i(t))$, and in the second equation $B(u_{i-1}(t)) \approx B(u_{i+1}(t))$ for small $t$.

We assume to estimate the error of the nonlinear operator as

$$||A(u_{i-1}(t^{n+1}))u_i(t^{n+1}) - A(u^{n+1})u(t^{n+1})|| \leq \varepsilon, \tag{5.113}$$

between the discrete approach $A(u_{i-1}(t^{n+1}))u_i(t^{n+1})$ and the analytical solution $A(u^{n+1})u(t^{n+1})$, such that there exists an iteration index $\tilde{i} \geq i$ with $i \in \{1, 3, \ldots, 2m+1\}$, that fulfils the Eq. (5.113).

*Remark 5.18* The linearization with the fixpoint scheme can be used for smooth or weak nonlinear operators, otherwise we lose the convergence behaviour, while we did not converge to the local fixpoint, see [34].

**Operator Splitting Method with Embedded Jacobian Newton Iterative Method**

The Newton's method is used to solve the nonlinear parts of the iterative operator splitting method, see the linearization techniques in [34, 36]. We apply the iterative operator splitting method and obtain:

$$\begin{aligned}
&F_1(u_i) = \partial_t u_i - A(u_i)u_i - B(u_{i-1})u_{i-1} = 0,\\
&\text{with} \quad u_i(t^n) = c^n,\\
&F_2(u_{i+1}) = \partial_t u_{i+1} - A(u_i)u_i - B(u_{i+1})u_{i+1} = 0,\\
&\text{with} \quad u_{i+1}(t^n) = c^n,
\end{aligned}$$

where the time step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m + 1$. $c_0(t) = 0$ is the starting solution and $c^n$ is the known split approximation at the time

level $t = t^n$. The results of the methods are $c(t^{n+1}) = u_{2m+2}(t^{n+1})$. The splitting method with embedded Newton's method is given as

$$u_i^{(k+1)} = u_i^{(k)} - D\left(F_1\left(u_i^{(k)}\right)\right)^{-1}\left(\partial_t u_i^{(k)} - A\left(u_i^{(k)}\right)u_i^{(k)} - B\left(u_{i-1}^{(k)}\right)u_{i-1}^{(k)}\right),$$

$$\text{with } D\left(F_1\left(u_i^{(k)}\right)\right) = -\left(A\left(u_i^{(k)}\right) + \frac{\partial A\left(u_i^{(k)}\right)}{\partial u_i^{(k)}}u_i^{(k)}\right),$$

and $k = 0, 1, 2, \ldots, K$,

with $u_i(t^n) = c^n$,

$$u_{i+1}^{(l+1)} = u_{i+1}^{(l)} - D\left(F_2\left(u_{i+1}^{(l)}\right)\right)^{-1}\left(\partial_t u_{i+1}^{(l)} - A\left(u_i^{(k)}\right)u_i^{(k)} - B\left(u_{i+1}^{(k)}\right)u_{i+1}^{(k)}\right),$$

$$\text{with } D\left(F_2\left(u_{i+1}^{(l)}\right)\right) = -\left(B\left(u_{i+1}^{(l)}\right) + \frac{\partial B\left(u_{i+1}^{(l)}\right)}{\partial u_{i+1}^{(l)}}u_{i+1}^{(l)}\right),$$

and $l = 0, 1, 2, \ldots, L$,

with $u_{i+1}(t^n) = c^n$.

*Remark 5.19* For the iterative operator splitting method with Newton's method we have two iteration procedures. The first iteration is Newton's method for computing the solution of the nonlinear equations, the second iteration is the iterative splitting method, which computes the resulting solution of the coupled equation systems. The embedded method is used for strong nonlinearities.

### *5.2.11 Parallelization: Parareal*

The Parareal algorithm can be given as a multiple shooting method.

We assume to have a partitioning in time $\Omega_T = [0, T]$ divided into $N$ subdomains:

$$\Omega_n = [T_{n-1}, T_n],\ n = 1, 2, \ldots, N. \tag{5.114}$$

We define the following solvers:

(1) Coarse propagator $G(T_n, T_{n1}, x)$, (coarse solver)
(2) Fine propagator $F(T_n, T_{n-1}, x)$, (fine solver)

where we obtain an approximation $U_n$ of the equation

$$\frac{dU}{dt} = f(t, U(t)),\ \text{with } U(T_{n-1}) = x. \tag{5.115}$$

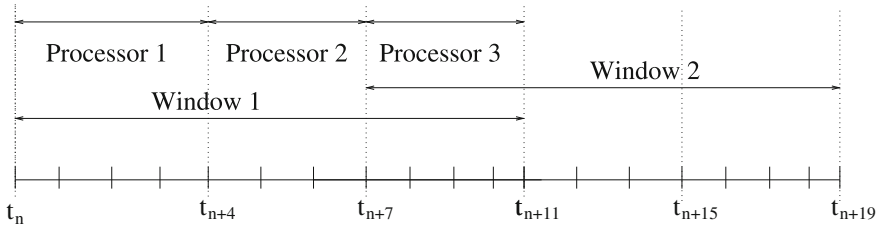



**Fig. 5.21** Parallelization with Parareal, windowing of the parallel process

Here we apply fine and coarse propagators, while fine propagators are expensive and coarse propagator cheap to compute.

The corrections are done with respect to the improved computation of the finer propagator.

$$U_n^{k+1} = F(T_n, T_{n-1}, U_{n-1}^k) + G(T_n, T_{n-1}, U_{n-1}^{k+1}) - G(T_n, T_{n-1}, U_{n-1}^k), \tag{5.116}$$

where the initial guesses are $U_{n-1}^k$ and the coarse propagator is $G$, while the fine propagator is $F$. $k$ is the iteration index.

*Example 5.2* We assume to have $F$ as the iterative splitting propagator and $G$ as the A–B splitting propagator.

Further the iterative splitting scheme include additionally a fixpoint scheme for nonlinear problems.

So we step by each window to the next time interval, see Fig. 5.21.

### *5.2.12 Test Example: Simple Car Body*

We start with a simple test example for the coupling between a heat equation with convection term and a fluid flow given by a convection equation, see also the results in [21]

$$\partial_t\, T = \nabla \cdot (K \nabla T) - \nabla \cdot \mathbf{v} T, \tag{5.117}$$

$$\partial_t \mathbf{v} = -(\mathbf{v} \cdot \nabla)\mathbf{v} - \nabla p, \tag{5.118}$$

$$T(\mathbf{x}, t_0) = T_0(\mathbf{x}), \tag{5.119}$$

$$\mathbf{v}(\mathbf{x}, t_0) = \mathbf{v}_0(\mathbf{x}), \tag{5.120}$$

where the unknown temperature is $T$, $\mathbf{v}$ is the flow field of the temperature and $p$ is a given pressure. We assume to have Neumann conditions at the boundaries. The spacial domain of this problem is the interval $[0, 1]$ with thermal conductivity $K = 0.01$, which we discretize by finite differences at $n + 1$ equidistant nodes $x_j = j/n$ $(j = 0, 1, \ldots, n)$. This results in the following system

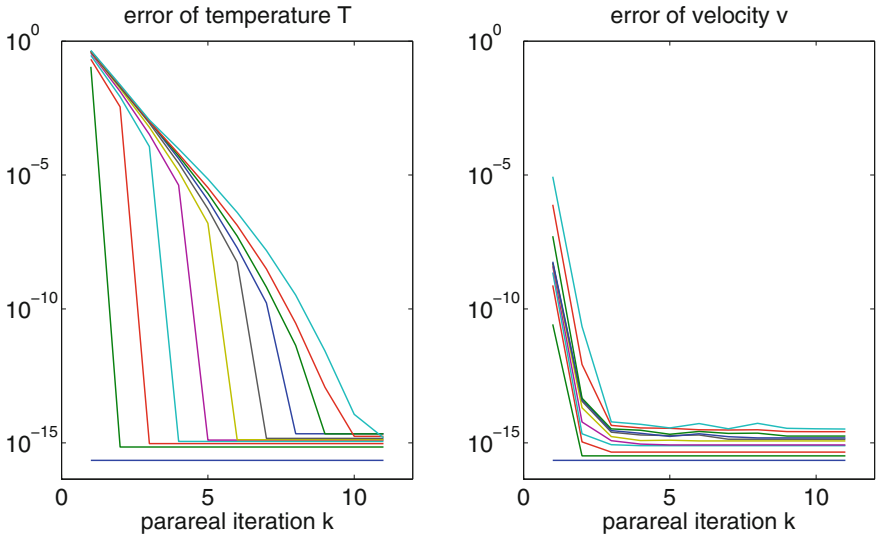



**Fig. 5.22** Convergence of parareal for the test problem

$$\partial_t \mathbf{T} = D_2\mathbf{T} - D_1(\mathbf{T} \circ \mathbf{v}) + \mathbf{b} \tag{5.121}$$

$$\partial_t \mathbf{v} = -\mathbf{v} \circ D_1\mathbf{v} - D_1\mathbf{p}, \tag{5.122}$$

where the matrices $D_1, D_2$ discretize first and second order differential operators and $\mathbf{b}$ is a vector representing the boundary data. In a real-world problem the pressure vector $\mathbf{p}$ will depend on the temperature vector $\mathbf{T}$. The whole problem is integrated over the time domain $\Omega_T = [0, 1]$, which we have divided into 10 subdomains of equal length.

The convergence of parareal is depicted in Fig. 5.22. Each convergence curve corresponds to the error of the computed temperature $T$ (left plot) and velocity $v$ (right plot) measure at each of the 11 coarse time points $T_0, T_1, \ldots, T_N$, where the abscissae indicate the parareal iteration index $k$. Note that after $k = 11$ iterations the algorithm reaches the accuracy of the fine propagator, the result of which was also used as the reference solution for computing the error. We also note that the convergence for the velocity field is much more rapid than for the temperature, see Fig. 5.22.

In Fig. 5.23 we see the convergence behaviour between AB and Strang splitting schemes.

*Remark 5.20* An efficient combination of different splitting schemes and higher order time-integrators allows to optimize the application of a parallel time-propagator. While we achieve fast solvers, that are decoupled for each different systems $A$ or $B$, we achieve realistic time-accelerations.

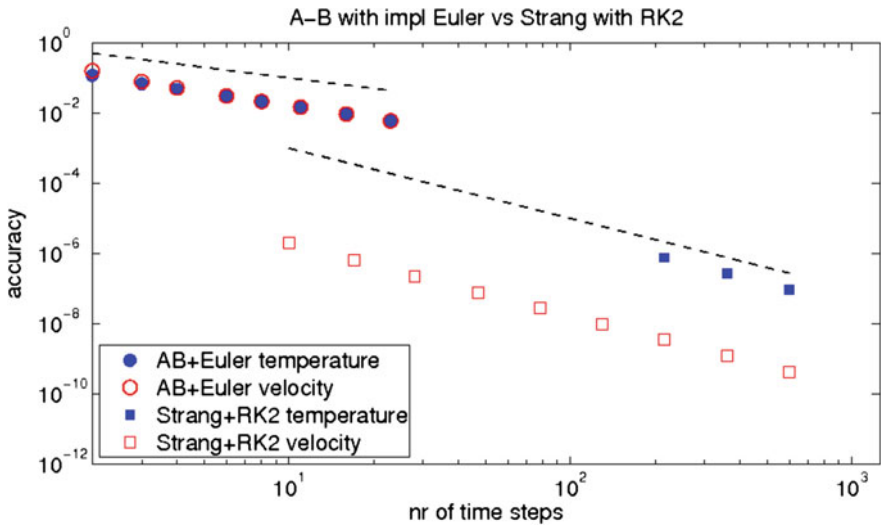



**Fig. 5.23** Convergence of parareal wit AB and Strang splitting scheme

### *5.2.13 Conclusion*

We have discussed a multiscale problem based on two different physical problems, where we have an equation dealing with the heat transport and a next equation based on the flow field. For the technical realization of the problem, e.g. in car body heating, we have to couple two codes with different spatial- and time steps. We could solve such a problem by an improvement of existing coupling methods with higher order splitting schemes and parallelization methods. A numerical test example presents the effectivity of the higher order schemes in a parallel realization. While producing more accurate solutions with larger time steps, it helps to achieve realistic computational times.

## 5.3 Multiscale Methods for Levitron Problem: Iterative Implicit Euler Methods as Multiscale Solvers

**Abstract** We describe a multiscale problem related to a control problem of a gyroscope that circulates in a static magnetic field about the horizontal and vertical axis, also called Levitron, see [37]. While the perturbations of the Levitron in $x$- and $y$-directions are very small (fine scale), the perturbations in $z$-direction is much more larger (coarse scale), such that we have to deal with a multiscale problem, see [38]. Based on the model, we have two modelling options of the asymmetric levitron model:

1. constraint Hamiltonian (full coordinates of the rotationmatrix, i.e. 6 angles), or
2. unconstraint Hamiltonian (minimal coordinates of the rotationmatrix, i.e. 3 angles).

For the minimal coordinates, we substitute the constraint and deal with a non-constraint Hamiltonian. While with full coordinates, we have the constraint given as $Q^tQ - I = 0$, which means the motion of the orientation of the body lies in $Q \in SO(3)$ is important. Non-constraint methods solve the non-constraint Hamiltonian and add the constraint via the stability conditions, see [39, 40]. We discuss semi-implicit methods, that embed the fine scale resolutions into the coarser scales, e.g. semi-implicit Euler methods.

### *5.3.1 Introduction*

Nonlinear dynamical systems with non-separable Hamiltonians are delicate to solve and standard integrators fail, see [39, 41–43]. Based on their multiscale behaviour, we have to integrate via fine and coarse timescales. Therefore, one of the main challenges is to design integrators, which can cover such different scales. Such integrators, which have to be cheap in computational costs and should also fulfil the physical constraints of energy conservation, see [16]. Here, we discuss the two main ideas:

- Explicit Methods: All scales are resolved, but the finest scales are responsible to the time step, means that we have to reduce the model and upscale such a problem or we have taken into account very fast explicit solvers.
- Implicit Methods: Only the coaser scales are resolved, while the finer scales are averaged based on the implicit smoothening. Means we have covered the finer scales, via the implicit or backward scheme in the model problem, but resolve only the coarser scale. Such an idea allows to apply larger time steps and reduce computational time, see [18].

We discuss and compare such explicit and implicit methods for nonlinear dynamical systems, see [38, 44], and use the Levitron as a test case, while we have a multiscale problem, see [41]. The Levitron is a gyroscope that circulates in a static magnetic field about the horizontal and vertical axis, see [37]. Only dynamical stability exists according to the theorem of Earnshaw [45] based on the spinning with angular velocity about its symmetry axis, see [37, 46, 47]. Such stability studies require long-term stability of the integrators.

To overcome the restrictions of standard explicit integrators, we discuss a novel class of integrators using an implicit Euler scheme embedded to a Waveform Relaxation, called iterative Euler scheme, see [48].

We present their effective computational costs and energy conservation properties. Because we deal with a nonlinear and non-separable Hamiltonian, the standard symplectic schemes fails and we have to design integrators which resolve the nonlinearity and approach energy conservation as good as possible, see [16]. A combination of

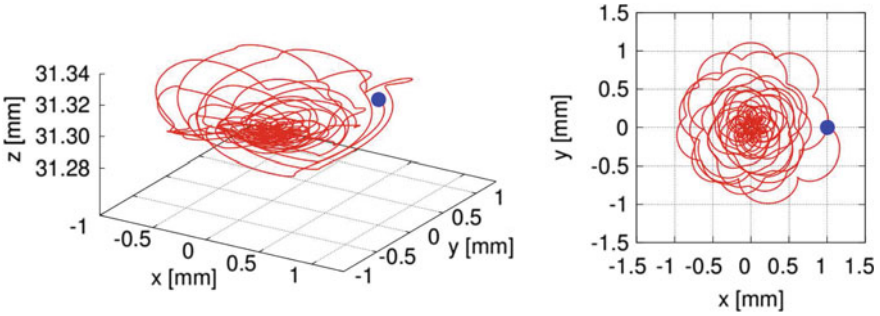



**Fig. 5.24** Trajectory for a Levitron with initial point ($x = 1\,\text{mm}, y = 0\,\text{mm}, z = 31.3\,\text{mm}$) coloured in *blue*

an implicit Euler method embedded to a Waveform-relaxation scheme allows to gain such properties of a computationally cheap and asymptotic symplectic scheme. The test case is a trajectories approach of a stable attractor, which can be computed, see [38]. In Fig. 5.24, we show the trajectory of such a Levitron for a time of about 20 h. The starting point is marked as a blue dot. Obviously, long-time stable integrators are needed for stability studies, otherwise one would fail to reach a stable attractor [38, 46].

### 5.3.2 Unconstraint Hamiltonian of the Levitron Problem

We deal with an asymmetric levitron problem, that is derived in the literature, see [46].

We begin with the kinetic energy equation

$$T = \frac{1}{2}\left[m\left(\dot{x}^2 + \dot{y}^2 + \dot{z}^2\right) + A\left(\dot{\theta}^2 + \dot{\phi}^2 \sin^2\theta\right) + C\left(\dot{\psi} + \dot{\phi}\cos\theta\right)^2\right], \tag{5.123}$$

and the potential energy equation:

$$U = mgz - \mu\left[\sin\psi\sin\theta\frac{\partial V}{\partial x} + \cos\psi\sin\theta\frac{\partial V}{\partial y} + \cos(\theta)\frac{\partial V}{\partial z}\right], \tag{5.124}$$

with $\mu$ as the magnetic moment of the top and $A$ and $C$ as the principal moments of inertia.

Therefore the Hamiltonian is given as

$$\begin{aligned}\mathscr{H} = {} & \frac{1}{2m}\left(p_x^2 + p_y^2 + p_z^2\right) + \frac{p_\theta^2}{2A} + \frac{p_\psi^2}{2C} + \frac{(p_\phi - p_\psi\sin\theta)^2}{2A\cos^2\theta} \\ & + mgz - \mu\left(\frac{1}{2}\Phi_2(z)(x\sin\theta + y\cos\theta\sin\phi)\right. \\ & \left. + (-\Phi_1(z) + \frac{1}{4}(x^2 + y^2)\Phi_3(z))\cos\theta\cos\phi\right).\end{aligned} \tag{5.125}$$

The equations of motions are extended with the Lagrange multiplier: we obtain a non-separable Hamiltonian of (5.125) given as

$$\begin{aligned}\dot{\mathbf{q}} &= \frac{\partial H}{\partial \mathbf{p}}(\mathbf{p},\mathbf{q}) \\ &= \left(\frac{p_x}{m}, \frac{p_y}{m}, \frac{p_y}{m}, \frac{p_\theta}{A}, \frac{p_\psi}{C} - \frac{p_\phi \sin\theta - p_\psi \sin^2\theta}{A\cos^2\theta}, \frac{p_\phi - p_\psi \sin\theta}{A\cos^2\theta}\right), \end{aligned} \tag{5.126}$$

and

$$\begin{aligned}\dot{\mathbf{p}} &= -\frac{\partial H}{\partial \mathbf{q}}(\mathbf{p},\mathbf{q}) \\ &= \left(\mu\rho\left[\frac{1}{2}\Phi_2(z)\sin\theta + \frac{1}{2}x\Phi_3(z)\cos\theta\cos\phi\right],\right. \\ &\quad \mu\rho\left[\frac{1}{2}\Phi_2(z)\cos\theta\sin\phi + \frac{1}{2}y\Phi_3(z)\cos\theta\cos\phi\right], \\ &\quad \mu\rho\left[\frac{1}{2}\Phi_3(z)\,(x\sin\theta + y\cos\theta\sin\phi) + \right. \\ &\qquad \left.\left(-\Phi_2(z) + \frac{1}{4}(x^2+y^2)\Phi_4(z)\right)\cos\theta\cos\phi\right] - mg, \\ &\quad -\frac{2p_\psi(p_\phi - p_\psi\sin\theta)}{\cos\theta} - \frac{2\sin\theta(p_\phi - p_\psi\sin\theta)^2}{\cos^3\theta} \\ &\quad + \mu\rho\left[\frac{1}{2}\Phi_2(z)(x\cos\theta - y\sin\theta\sin\phi)\right. \\ &\qquad \left.- \left(-\Phi_1(z) + \frac{1}{4}(x^2+y^2)\Phi_3(z)\right)\sin\theta\cos\phi\right], 0, \\ &\quad \mu\rho\left[\frac{1}{2}\Phi_2(z)y\cos\theta\cos\phi\right. \\ &\qquad \left.\left.- \left(-\Phi_1(z) + \frac{1}{4}(x^2+y^2)\Phi_3(z)\right)\cos\theta\sin\phi\right]\right). \end{aligned} \tag{5.127, 5.128}$$

### 5.3.3 *Integrator for Unconstraint Hamiltonian*

To circumvent the expensive computations of implicit methods, we use the property of the Hamiltonian, that the nonlinear function depends only on the recent variable in the function producing a decoupling with respect to Picard's-fixpoint schemes.

The initial value of (5.126) and (5.127) is rewritten in the following form:

$$\mathbf{u}' = \mathbf{f}(\mathbf{u}, \mathbf{u}, t), \mathbf{u}(0) = \mathbf{u}_0, \tag{5.129}$$

where we have the special structure of the Hamiltonian problem

$$\mathbf{u} = \begin{pmatrix} \mathbf{p} \\ \mathbf{q} \end{pmatrix}, \ \mathbf{f}(\mathbf{u}, \mathbf{u}, t) = \begin{pmatrix} \mathbf{f}_1(\mathbf{p}, \mathbf{q}) \\ \mathbf{f}_2(\mathbf{p}, \mathbf{q}) \end{pmatrix} = \begin{pmatrix} -\frac{\partial H}{\partial \mathbf{q}}(\mathbf{p}, \mathbf{q}) \\ \frac{\partial H}{\partial \mathbf{p}}(\mathbf{p}, \mathbf{q}) \end{pmatrix}. \tag{5.130}$$

The well-known Picard or Waveform-relaxation scheme, see [49], for system (5.129) has the form

$$\mathbf{u}'^{i+1} = \mathbf{f}(\mathbf{u}^{i+1}, \mathbf{u}^i, t), \ \mathbf{u}^{i+1}(0) = \mathbf{u}_0, \tag{5.131}$$

where $x^0(t)$ is an initial iteration and the nonlinear splitting function $\mathbf{f} : (\mathbb{R}^m)^2 \times [0, T] \to \mathbb{R}^m$.

- The semi-implicit Euler scheme, see [16], is applied with the difference approximation and is given as

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \ \mathbf{f}(\mathbf{u}^{n+1}, \mathbf{u}^n, t^n). \tag{5.132}$$

The exact integrator is given as

$$(J\mathbf{u}')(t) = u(0) + \int_0^t \mathbf{u}'(s) \, ds. \tag{5.133}$$

The semi-implicit Euler integrator, with respect to the Hamiltonian structure is given as

$$(J_{\text{implicit Euler}}\mathbf{u}')(t) = t\mathbf{u}(t). \tag{5.134}$$

- The semi-implicit Lobatto IIIA-IIIB pairs scheme is applied with the difference approximation and is given as

$$k_1^{i+1} = \mathbf{f}_1\left(\mathbf{p}^n, \mathbf{q}^n + \frac{\Delta t}{6}\left(l_1^i - l_2^i\right)\right), \tag{5.135}$$

$$k_2^{i+1} = \mathbf{f}_1\left(\mathbf{p}^n + \frac{\Delta t}{24}\left(5k_1^i + 8k_2^i - k_3^i\right), \mathbf{q}^n + \frac{\Delta t}{6}\left(l_1^i + 2l_2^i\right)\right), \tag{5.136}$$

$$k_3^{i+1} = \mathbf{f}_1\left(\mathbf{p}^n + \frac{\Delta t}{6}\left(k_1^i + 4k_2^i + k_3^i\right), \mathbf{q}^n + \frac{\Delta t}{6}\left(l_1^i + 5l_2^i\right)\right), \tag{5.137}$$

$$l_1^{i+1} = \mathbf{f}_2\left(\mathbf{p}^n, \mathbf{q}^n + \frac{\Delta t}{6}\left(l_1^i - l_2^i\right)\right), \tag{5.138}$$

$$l_2^{i+1} = \mathbf{f}_2\left(\mathbf{p}^n + \frac{\Delta t}{24}\left(5k_1^i + 8k_2^i - k_3^i\right), \mathbf{q}^n + \frac{\Delta t}{6}\left(l_1^i + 2l_2^i\right)\right), \tag{5.139}$$

$$l_3^{i+1} = \mathbf{f}_2\left(\mathbf{p}^n + \frac{\Delta t}{6}\left(k_1^i + 4k_2^i + k_3^i\right), \mathbf{q}^n + \frac{\Delta t}{6}\left(l_1^i + 5l_2^i\right)\right), \tag{5.140}$$

$$\mathbf{p}^{n+1,i+1} = \mathbf{p}^n + \frac{\Delta t}{6}\left(k_1^i + 4k_2^i + k_3^i\right), \tag{5.141}$$

$$\mathbf{q}^{n+1,i+1} = \mathbf{q}^n + \frac{\Delta t}{6}\left(l_1^i + 4l_2^i + l_3^i\right). \tag{5.142}$$

where $k_1^0 = k_2^0 = k_3^0 = \mathbf{p}^n$ and $l_1^0 = l_2^0 = l_3^0 = \mathbf{q}^n$.

**Corollary 5.1** *The system (5.129) has a unique solution* $\mathbf{u}'^*$ *and the sequence* $\{\mathbf{u}^i\}$ *applied by the algorithm (5.131) converge to* $\mathbf{u}^*$*, where* $\mathbf{u}^* = J\mathbf{u}'^*$ *is the unique solution of (5.129).*

*Proof* The proof is following [50].

The iterative Lobatto III A method is given in Algorithm 5.4. The same idea can be done with the Lobatto III B method.

**Algorithm 5.4** We compute the time steps $n = 1, 2, 3, \ldots, N$ and the starting point is $u_0^{n+1} = u^n$, the time step is given with $\Delta t$, error bound: $\varepsilon = 10^{-5}$.

1. Initialization $i = 0$

$$\mathbf{u}^i(t^{n+1}) = \mathbf{u}(t^n), \tag{5.143}$$

2. Iterative Steps

$$\begin{aligned}\mathbf{u}^{i,n+1} = \mathbf{u}^n &+ \frac{\Delta t}{6}\left(\mathbf{f}(\mathbf{u}^n, t^n) + \mathbf{f}(\mathbf{u}^{i-1,n+1}, t^n)\right) \\ &+ \frac{2\Delta t}{3}\,\mathbf{f}\left(\frac{1}{2}(\mathbf{u}^n + \mathbf{u}^{i-1,n+1})\right. \\ &\left. + \frac{\Delta t}{8}\left(\mathbf{f}(\mathbf{u}^n, t^n) - \mathbf{f}(\mathbf{u}^{i-1,n+1}, t^{n+1})\right), t^{n+1/2}\right).\end{aligned} \tag{5.144}$$

3. Stopping Criterion: If $i = I$ or the error is given as

$$||\mathbf{u}^{i,n+1} - \mathbf{u}^{i-1,n+1}|| \leq \varepsilon, \tag{5.145}$$

we have $\mathbf{u}^{n+1} = \mathbf{u}^{i,n+1}$.
Else Goto step 1.

*Remark 5.21* The combination of the Labatto III A and Lobatto III B method allows to achieve a symplectic scheme, see [44]. Based on the iterative embedding, we could accelerate the schemes, while we apply explicitly computed informations of the previous steps.

An alternative approach is given in the following by the constraint Hamiltonian formulation of the problem.

### 5.3.4 Integrator with Lagrangian Multiplier (Constraint Hamiltonian)

We have the following equation, see the paper [51]:

$$H(q,p) = \frac{1}{2} p^t M^{-1} p + U(q), \tag{5.146}$$

where the constraint is given as $\mathscr{M} = \{(q,p);\ g(q) = 0, G(q)M^{-1}p = 0\}$.

To solve this system, we apply the Rattle algorithm:

$$p_{n+1/2} = p_n - \frac{\Delta t}{2}(\nabla U(q_n) + G(q_n)^t \lambda_{1,n}), \tag{5.147}$$

$$q_{n+1} = q_n + \Delta t\ M^{-1} p_{n+1/2}, \tag{5.148}$$

$$0 = g(q_{n+1}), \tag{5.149}$$

$$p_{n+1} = p_{n+1/2} - \frac{\Delta t}{2}(\nabla U(q_{n+1}) + G(q_{n+1})^t \lambda_{2,n+1}), \tag{5.150}$$

$$0 = G(q_{n+1}) M^{-1} p_{n+1}. \tag{5.151}$$

Further we have to solve the Lagrangian multipliers as

$$\Lambda^{l+1} = \Lambda^l - \mathbf{J}_\sigma^{-1} \underline{\sigma}(t^n + \Delta t), \tag{5.152}$$

where $\Lambda = (\lambda_1, \lambda_2)^t$, $\mathbf{J}_\sigma$ is the Jacobian of the equations $\sigma = \big(g(q), G(q)M^{-1}p\big)$

$$\mathbf{J} = \begin{pmatrix} \frac{\partial \sigma_1}{\partial \lambda_1} & \frac{\partial \sigma_1}{\partial \lambda_2} \\ \frac{\partial \sigma_2}{\partial \lambda_1} & \frac{\partial \sigma_2}{\partial \lambda_2} \end{pmatrix}. \tag{5.153}$$

*Remark 5.22* We solve the Rattle algorithm in the following manner:

(1) We set Eq. (5.147) in Eq. (5.148). Then Eq. (5.148) in Eq. (5.149) and we obtain a nonlinear equation for $\lambda_{1,n}$. Such an equation, we can solve via a Newton or Newton-like method.

(2) The we set Eq. (5.150) in Eq. (5.151).We obtain a linear equation for $\lambda_{2,n+1}$, which we can solve directly.

### 5.3.5 Numerical Experiments

The model equations are the following [48]:

$$H = \frac{1}{2m}\left(p_x^2 + p_y^2 + p_z^2\right) + \frac{p_\theta^2}{2A} + \frac{p_\psi^2}{2C} + \frac{(p_\phi - p_\psi \sin\theta)^2}{2A\cos^2\theta}$$
$$+ \; mgz - \mu\rho\left(\frac{1}{2}\Phi_2(z)(x \;\sin\theta + y\cos\theta\sin\phi)\right.$$
$$\left. + \; (-\Phi_1(z) + \frac{1}{4}(x^2+y^2)\Phi_3(z))\cos\theta\cos\phi\right). \tag{5.154}$$

where the higher order $\Phi_i(z)$ are defined as $\Phi_i(z) = \frac{\partial\Phi_{i-1}(z)}{\partial z}$.

We resolve the nonlinearity of the Hamiltonian with asymptotic symplecticity, due to the fact of the symplectic kernel.

While shorter times in the range of $20 \le t \le 80$ s can be reached with sufficient energy conservation by explicit schemes (e.g. Runge–Kutta fourth-order schemes), the overall benefit of the novel schemes are long-time conservations, see Fig. 5.25. For long-time studies, the Levitron moved into the stable attractor $x = 0, y = 0, z = z_s$, which is the stable point and we see only small perturbations.

For moderate time intervals, the improvement of the explicit integrators can be done by iterative or extrapolated algorithms, see [52]. The computational time of

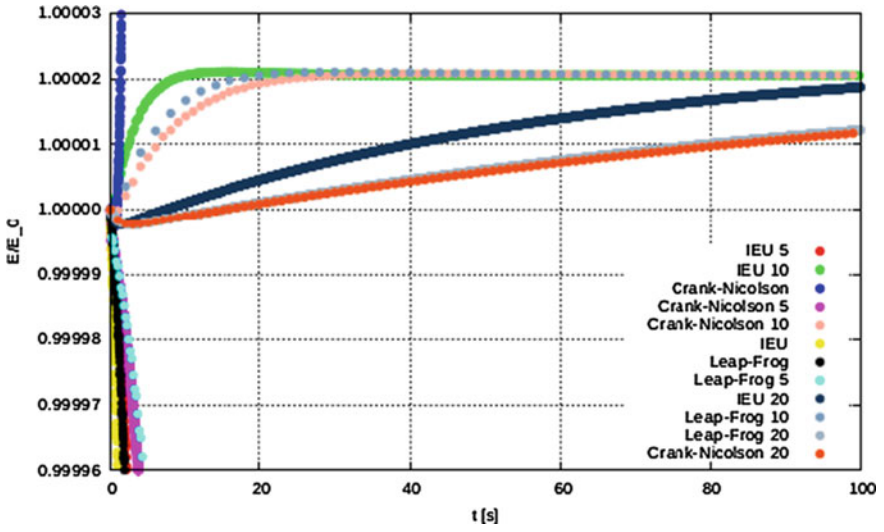



**Fig. 5.25** Long energy computations with semi-implicit methods in the time interval $t = [0, 100]$ s ($10^6$ timesteps)

**Table 5.1** Computational time for $10^6$ time steps of explicit and implicit schemes

| Explicit schemes | Comput. time in s | Extrapol. Verlet schemes | Comput. time in s | Semi-implicit schemes | Comput. time in s |
|---|---|---|---|---|---|
| Euler | 12.5 | Ord. 4 | 66.8 | it. Euler $i = 1$ | 12.6 |
| RK4 | 53.6 | Ord. 6 | 112.4 | it. Euler $i = 5$ | 62.65 |
| Verlet | 24.9 | Ord. 8 | 177.6 | it. Euler $i = 10$ | 124.6 |
| Time dep. Verlet | 62 | Ord. 10 | 249.4 | | |

such schemes is increased, see Table 5.1, such that an application of the iterative implicit Euler is more efficient.

In summary, the iterative implicit Euler schemes are more effective, because they can resolve the nonlinear Hamiltonian structure and gain asymptotic symplecticity with at least 5–10 iterations. We overcome the restrictions of the small time steps as known for the explicit schemes, e.g. Courant–Friedrichs–Levy condition. Compared with the fourth-order Runge–Kutta scheme only twice computational time is needed. In fact the method is computationally even more efficient, see also Table 5.1, because to obtain results with the same small error in energy conservation for explicit schemes a very large number of iterations with very small time steps are needed.

In general applications, the choice of the integrator will be determined by the question to be addressed: if the problem does not require long times to be studied a relatively simple integrator can be the most efficient, because it requires very little computational effort. For long-time analysis like the stability problem of the Levitron a necessary prerequisite for a correct solution is a very small error in energy conservation for this system. Therefore, only the more complex iterative solver is a good choice for this. Systems where conservation of energy or momentum are not guaranteed analytically, e.g. systems with viscous forces, will not benefit from the higher order schemes and simpler integrators can be used.

### *5.3.6 Conclusions and Discussions*

We discussed the explicit and implicit time integrators for nonlinear problems with non-separable Hamiltonians. As a test problem the Levitron is used to deal with a multiscale problem. Explicit methods are fast but reach results with small errors in energy conservation for time intervals of about $t \leq 50\,\mathrm{s}$. In contrast, implicit methods are more expensive due to their additional iterative cycles, but approach asymptotic symplecticity and therefore preserves the energy with quite small errors. An iterative implicit Euler scheme, which is long-time stable and efficient in the iterative cycles, was constructed. We avoid expensive inversion of matrices, which is necessary for fully implicit methods, by using waveform relaxation schemes. This new class of semi-implicit Euler schemes was able to demonstrate long-time stability of about 20 h for one test trajectory with only twice the computational effort compared

with standard fourth-order Runge–Kutta. This approach allows to design also higher order schemes and overcome the multiscale behaviour of such nonlinear dynamical systems.

## 5.4 Particle Method as Multiscale Problem: Adaptive Particle in Cell with Numerical and Physical Error Estimates

**Abstract** Particle methods are in general multiscale methods, while dealing with different spatial- and timescales. In this section, we discuss an adaptive particle in cell method based on multidimensional problems. Here, we concentrate on discussing the numerical and physical errors of this method. The motivation arose to the fact, that the reduction of computational time of particle methods is important and only modified methods, combining grid-based and grid-free methods with adaptive extension can overcome such problems, see [53, 54]. Form a multiscale viewpoint, we deal with different scales, e.g. near- and fare-field problems, means microscopic problems related to the particles and macroscopic problems related to the underlying electromagnetic fields, see [55]. The idea is to accelerate the solver processes with respect to the adaptivity and resolve coarser grids, while the finer scales are averaged via such larger scales, e.g. with implicit schemes or we neglect such no important scales. The main problem of the adaptive or nonuniform grids are errors in the numerical schemes and also errors in the physical setting. We consider a multiscale error analysis of particle in cell (PIC), while we couple to different scales:

- partial differential equations of the field, e.g. Maxwell's equation or Poisson's equation,
- ordinary differential equations of the particles, e.g. Newton's equation of motion for each particles,
- collision equations of the particles, e.g. Stochastic equations or binary collision equations, see ideas in [5].

We concentrate on the first two equations. Both equations can influence the error of each other systems. While the parts are coupled via interpolation functions, e.g. spline functions, we have taken into account a full cycle of PIC to localize the errors. In this section, we taken into account to the numerical and physical error of the PIC method with respect to an adaptation of the schemes. Based on the statistical influence of the scheme, while averaging and variances are important to obtain nearly stationary solutions, we also consider expectations and variances of the multiple runs of the scheme.

### *5.4.1 Introduction*

We are motivated to estimate the errors of a PIC schemes with different spatial grids.

In the following, we discussed the improved PIC cycles based on improving all parts of the cycle, see Fig. 5.26.

The following three parts of the PIC scheme are involved to the error estimates:

- Pusher (scheme to solve the mesh-free equation of motions).
- Solver (scheme to solve the mesh-based potential equations).
- Interpolation (approximation schemes to couple the mesh-free parameters with the mesh parameters)

First all three parts are important and we have to deal with their numerical approximation. Second, the physical constraints, as conservation of mass, momentum and energy are important to the physical experiments and should be conserved by the underlying schemes.

By the way it is not enough to couple higher order schemes of all the three parts together and resume to have a higher order computation of the cycle, while the combination of the parts did not conserve the physical constraints.

In this paper, we discuss the error estimates for different schemes and their relation to the conservation constraints.

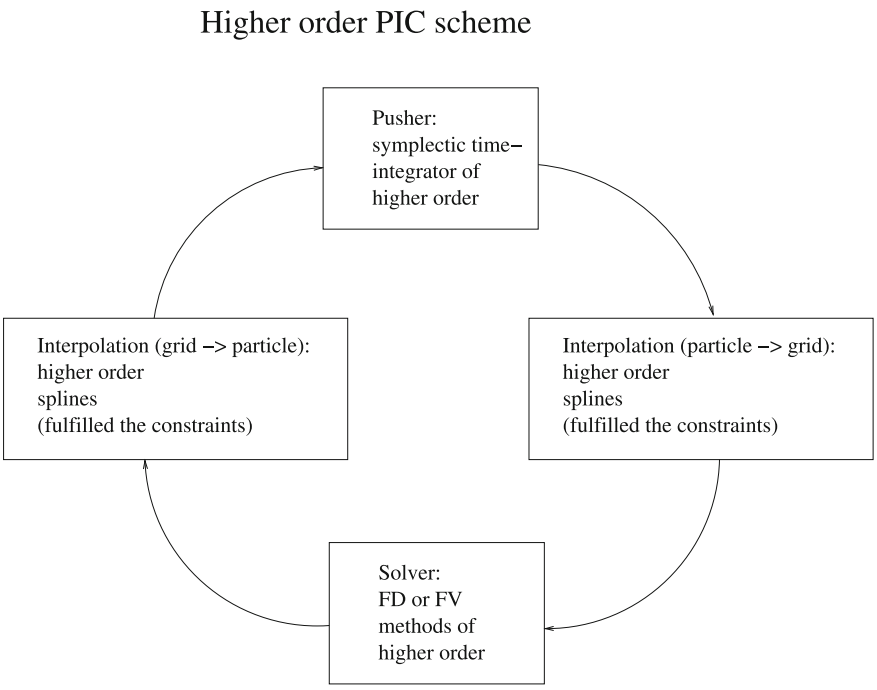



**Fig. 5.26** Improved PIC cycles for adaptive PIC

Based on the discretization schemes, we have the following assumptions:

**Assumption 5.5** Assumptions based on the one-dimensional discretization schemes, see [55]:

- $\Delta x \leq \lambda_D$,
- $\omega_p \Delta t \leq 2$,
- $L \geq \lambda_D$,
- $N_p \lambda_D \geq L$,

where $L$ is the domain length, $\omega_p$ is the plasma frequency, $\lambda_D$ is the Debye length, $\Delta x$ is uniform the spatial grid length and $\Delta t$ is the time step.

### *5.4.2 Mathematical Model*

In the following, we derive the mathematical model, while starting form a collision less system, which is related to the Vlasov equation, to a particle model based on superparticles (superposition of particles), see Fig. 5.27.

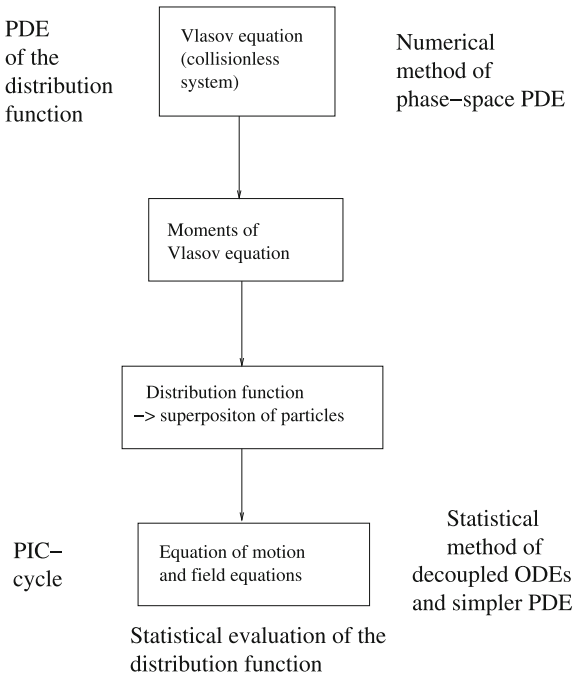



**Fig. 5.27** From Vlasov equation to PIC cycles

The governed equation is the Vlasov equation

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \mathbf{v} \cdot \frac{\partial f}{\partial \mathbf{x}} + \frac{\mathbf{F}}{m} \cdot \frac{\partial f}{\partial \mathbf{v}} = 0 \tag{5.155}$$

where $\mathbf{F} = q\ \mathbf{E} = -q\nabla\phi$,

Further the electric field in the electrostatic limit is described by the Poisson's equation

$$\nabla \cdot \nabla\phi = -\frac{\rho}{\varepsilon_0}, \tag{5.156}$$

where the net charge density is computed from the distribution function as

$$\rho(\mathbf{x}, t) = \sum_s q_s \int f(\mathbf{x}, \mathbf{v}, t)\ d\mathbf{v}, \tag{5.157}$$

The numerical approach to PIC is given as

$$f(\mathbf{x}, \mathbf{v}, t) = \sum_{p=1}^{N_p} f_p(\mathbf{x}, \mathbf{v}, t), \tag{5.158}$$

while the distribution function of each species is given as a superposition of several elements

$$f_p(\mathbf{x}, \mathbf{v}, t) = N_p S_{\mathbf{x}}(\mathbf{x} - \mathbf{x}_p(t)) S_{\mathbf{v}}(\mathbf{v} - \mathbf{v}_p(t)) \tag{5.159}$$

$S_{\mathbf{x}}$ and $S_{\mathbf{v}}$ are shape functions (e.g. B-splines) for the compuational particles.

**Moments of the Vlasov Equation**

We have the following moments of the Vlasov equation, which are the underlying equation of motion for our PIC cycles:

- 0-Moment : $\frac{dN_p}{dt} = 0$,
  (conservation of the number of physical particles),
- $1_{\mathbf{x}}$-Moment: $\frac{d\mathbf{x}_p}{dt} = \mathbf{v}_p$,
  $1_{\mathbf{v}}$-Moment: $\frac{d\mathbf{v}_p}{dt} = \frac{q}{m}\mathbf{E}_p$,
  (equation of motions for the physical particles)

We assume the following shape functions (interpolation), which map between grid to particles and fields to grid:

$$S_{\mathbf{v}}(\mathbf{v} - \mathbf{v}_p) = \delta(\mathbf{v} - \mathbf{v}_p), \tag{5.160}$$

$$S_{\mathbf{x}}(\mathbf{x} - \mathbf{x}_p) = \frac{1}{\Delta_p} B_l \left( \frac{\mathbf{v} - \mathbf{v}_p}{\Delta_p} \right), \tag{5.161}$$

where $B_l$ is a B-spline of order l, e.g. $l = 1$ is the known CIC shape function. Further $\Delta_p$ is the scale length of the support of the compuational particle.

PIC-cycle:

- Approximation (Grid to Particle):

$$\mathbf{E}_p = \sum_i \mathbf{E}_i W(\mathbf{x}_i - \mathbf{x}_p) \tag{5.162}$$

$W(\mathbf{x}_i - \mathbf{x}_p) = \int S_{\mathbf{x}}(\mathbf{x} - \mathbf{x}_p) B_l \left( \frac{\mathbf{x} - \mathbf{x}_p}{\Delta \mathbf{x}} \right)$,

- Equation of motion:

$$\frac{d\mathbf{x}_p}{dt} = \mathbf{v}_p, \tag{5.163}$$

$$\frac{d\mathbf{v}_p}{dt} = -\frac{q_s}{m_s} \mathbf{E}_p, \tag{5.164}$$

- Approximation (Particle to Grid)

$$\rho_i = \sum_p \frac{q_p}{\Delta \mathbf{x}} W(\mathbf{x}_i - \mathbf{x}_p), \tag{5.165}$$

- Field equation:

$$\Delta_h \phi_i = -\frac{\rho_i}{\varepsilon_0}, \tag{5.166}$$

$$\mathbf{E}_i = -\nabla_h \phi_i, \tag{5.167}$$

where $\Delta_h$ is the discrete second-order spatial operator and $\nabla_h$ is the discrete first order spatial operator.

### *5.4.3 Numerical Errors*

In the following, we discuss the numerical approximation errors, which is given by the numerical schemes:

- Finite Difference method or Finite-Volume Method,
- Direct solvers or iterative solvers,
- Adaptive Discretization (adaptive error of the finite difference or finite volume methods).

**Error Estimates for the Full PIC Cycle (Uniform Grid)**

We have to discuss the following elements of the cycle an estimate their errors.

Parts of the cycle:

(1) Pusher

$$\frac{dx_p}{dt} = v_p, \frac{dv_p}{dt} = F_p = \frac{e_p}{m_p} E(x_p), \tag{5.168}$$

with $p = 1, \ldots, P$ are the particles in the cycle and $q_p = \frac{e_p}{m_p}$ is the charge of the particle $p$.
The numerical scheme is given as a second order in time for one particle to time $t_{k+1}$:

$$x_{k+1} = x_k + \Delta t \; v_{k+1/2}, \tag{5.169}$$

$$v_{k+1/2} = v_{k-1/2} + 2\Delta t \; q \; E_k, \tag{5.170}$$

with $p = 1, \ldots, P$ are the particles in the cycle.

(2) Interpolation I (particle position to grid)

$$\rho_i = \sum_{j=1}^{J} q_j S(x_i - x_j(t_{k+1})), \tag{5.171}$$

where $x_j(t_{k+1})$ and $q_j(t_{k+1})$ are the position and charges of particle $j$ to time $t_{k+1}$.

(3) Solver and Interpolation II (grid to particle position)

$$E(x(t_{k+1})) = \sum_{i=1}^{I} E_i S(x_i - x(t_{k+1})), \tag{5.172}$$

and

$$E(x(t_{k+1})) = \sum_{i=1}^{I} \left( \sum_{k=1}^{K} g_{ik} \rho_k \right) S(x_i - x(t_{k+1})), \tag{5.173}$$

$$E(x(t_{k+1})) = \sum_{i=1}^{I} \left( \sum_{k=1}^{K} g_{ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1})) S(x_i - x(t_{k+1})) \right). \tag{5.174}$$

**Theorem 5.6** *For one PIC cycle, with CIC as Interpolation and second order in space for the solver and second order in time for the pusher, we assume*

- *Local Error of the Interpolation I:*

$$||E(x) - E(x_i)|| \leq O(\Delta x), \tag{5.175}$$

- *Local Error of the Solver scheme:*

$$||g_{exact} - g_{i,k}|| \leq O(\Delta x^2), \tag{5.176}$$

- *Local Error of the Interpolation II:*

$$||\rho(x) - \rho(x_i)|| \leq O(\Delta x), \tag{5.177}$$

- *Local Error of the Pusher scheme (Time-integrator):*

$$||E(x(t_{k+1})) - E(x, t_{k+1})|| \leq O(\Delta t^2), \tag{5.178}$$

$$||E^{int}(x(t_{k+1})) - E^{int}(x, t_{k+1})|| + ||E^{ext}(x(t_{k+1})) - E^{ext}(x, t_{k+1})|| \leq O(\Delta t^2), \tag{5.179}$$

*where $E = E^{int} + E^{ext}$ and $E^{int}$ is the internal and $E^{ext}$ is the external component.*

*Then the local error estimates is given as*

$$err_{local,PIC} = ||E - E_{num}|| \leq O(\Delta x) + O(\Delta t^2), \tag{5.180}$$

*where E is the exact electrical field and $E_{num}$ the numerical approximated electrical field with $\Delta x$ as spatial grid size and $\Delta t$ as time step.*

*Proof* We deal with the following time and space error estimates:

$$||E(x(t_{k+1})) - E(x, t_{k+1}) + E(x, t_{k+1}) - E(x_i, t_{k+1})|| \tag{5.181}$$

$$\leq ||E(x(t_{k+1})) - E(x, t_{k+1})|| + ||E(x, t_{k+1}) - E(x_i, t_{k+1})||. \tag{5.182}$$

The first part of the error estimates is the approximation error in time, while the second part is the approximation in space.

The first part is estimated based on the assumption of a second-order time-integration scheme:

$$||E(x(t_{k+1})) - E(x, t_{k+1})|| \leq O(\Delta t^2). \tag{5.183}$$

The second part is given as

$$\begin{aligned} &||E(x_{approx}, t_{k+1}) - E(x_i, t_{k+1})|| \\ &= || \left( \sum_{k=1}^{K} g_{,exact,ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1})) \right) \\ &\quad - \left( \sum_{k=1}^{K} g_{ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1})) \right) || \leq O(\Delta x^2). \end{aligned} \tag{5.184}$$

$$||E(x, t_{k+1}) - E(x_{approx}, t_{k+1})|| \leq O(\Delta x). \tag{5.185}$$

We combine all the results and obtain the local error estimates as

$$err_{local,PIC} = ||E - E_{num}|| \leq O(\Delta x) + O(\Delta t^2). \tag{5.186}$$

*Remark 5.23* The numerical error of the uniform PIC cycle is a combination of the spatial- and time-approximations. Means the higher the numerical approaches of each individual element of the cycle, the higher approach is also the underlying full error of the cycle, see further ideas of adaptation and their underlying errors [56].

**Error Estimates for Adaptive Grids**

In the following, we discuss the numerical errors for the adaptive grids.

For the adaptive grids, we have the following errors:

- Numerical errors (approximation errors to the numerical schemes).
- Physical errors (approximation errors to the physical constraints, e.g. self-force, inter-particle forces).

**Numerical Error for Adaptive Schemes Based on Physical Constraints**

In the literature there exists different example to improve standard PIC to adaptive PIC.

Here the problem are often that simple coupling ideas without deriving correct error estimates lacked.

In the following, we proof that only standard coupling of uniform discretization and standard shape functions, will produce large errors when concerning large time steps, e.g. $10^8$.

We have the following outline of the errors due to the PIC method:

Mathematical errors:

- Spatial symmetry is not correct
- Interpolation error of the standard CIC shape functions

**Spatial Symmetry is Neglected**

In the following, we derive the error of the neglected spatial symmetry without a corrected discretization

**Theorem 5.7** *The error of the self-forces for non-balanced nonuniform discretisation is given with*

$$err \leq O(\alpha_{\max} - \alpha_{\min}), \tag{5.187}$$

*while $\alpha_{max}$ is the maximum length of a grid cell and $\alpha_{min}$ is the minimum length of a grid cell.*

*Proof* Based on the idea to proof the D. Tskhakaya et al., we have the error of the self-force given as:

$$
\begin{aligned}
err &= \frac{e^2}{V_g}\sum_{i,k} g_{i,k} S(x_i - x) S(x_k - x) \\
&\quad - \frac{e^2}{V_g}\sum_{i,k} g_{k,i} S(x_k - x) S(x_i - x) && (5.188)\\
&\leq \frac{e^2}{V_g} |\sum_{i,k} (g_{i,k} - g_{k,i})| && (5.189)\\
&\leq \frac{e^2}{V_g} 2 N_g \left( \max_i^{N_g} \alpha_i - \min_k^{N_g} \alpha_k \right) && (5.190)\\
&\leq C\ O(\alpha_{\max} - \alpha_{\min}), && (5.191)
\end{aligned}
$$

where we assume $g_{i,k} \neq g_{k,i}$ and $||S(x_i - x)|| \leq 1$, $\alpha_{\max} = \max_i^{N_g} \alpha_i$ and $\alpha_{\min} = \min_i^{N_g} \alpha_i$.

*Remark 5.24*
- The most delicate case is given if the maximum and minimum length of a cell are very different:
  $\alpha_{\max} \gg \alpha_{\min}$,
  then the error is given with the scale of the largest cell:
  $err \leq O(\alpha_{\max})$.
- A further delicate case is given if we try to smooth the error over a long spatial scale, means the maximum and minimum of the scales are "nearly" the same:
  $\alpha_{\max} \approx \alpha_{\min}$,
  but here we have to taken into account the long-time stability:

*Example 5.3* We assume that the difference between the two scales are given as:
$diff = \alpha_{\max} - \alpha_{\min}$,
further we have about $N_G = 10^6$ cells and $N_r = 10^{10}$ runs (repetition of the method), then the difference between the cells have to be:

$$
\begin{aligned}
diff &\leq \frac{1}{N_g N_r}, && (5.192)\\
&\leq 10^{-16}, && (5.193)
\end{aligned}
$$

so we have to deal with a very small difference and the smoothing zone is very large.

- The optimal case is given if the maximum and minimum length of a cell are the same if we only use standard discretization schemes:
  $\alpha_{\max} = \alpha_{\min}$,
  then the error is zero:
  $err = 0$.

Without any correction, the self-force is going to infinity with very spatial scales, it means if we do not balance the discretization schemes and shape function, we have large errors in the computations.

The same can also be proved with two particle interaction forces.

#### 5.4.3.1 Higher Order Error Estimates with Adaptive Schemes

In the following, we discuss the adaptive PIC based on:

- Balanced discretization methods
- Weighted shape functions

Based on the balanced discretization method, we could show, that the self-force and inter-particle force are fulfilled, therefore, we also fulfil the momentum conservation, see [57].

The error estimates are only given with respect to the discretization error and interpolation errors.

**Theorem 5.8** *The error estimates is therefore given with respect to the discretization error and the interpolation error.*

$$err_{disc} = ||E - E_{num}|| \leq O((\Delta x)^{k_1}), \tag{5.194}$$

*where* $k_1$ *is the order of the discretization scheme and* $\Delta x$ *is the maximal grid step.*

$$err_{disc} = ||\rho - \rho_{inter}|| \leq O((\Delta x)^{k_2}), \tag{5.195}$$

*where* $k_2$ *is the order of the interpolation scheme* $S^k$*, while* $k = 0$ *is the NGP,* $k = 1$ *is the CIC and* $k = 2$ *is the quadratic B-spline shape function.*

*Proof* Based on the balanced discretization and the weighted shape functions, the self-forces and inter-particle forces are fulfilled.

#### 5.4.3.2 Error Estimates for the Full PIC Cycle for Adaptive Schemes

We assume to have the following approximation errors for our underlying numerical schemes:

- Pusher: $O(\Delta t^2)$
- Solver: $O(\Delta x_{\max}^2)$, where $\Delta x_{\max} = \max_{i=1}^{I}(\Delta x_i)$.
- Interpolation (adaptive CIC): $O(\Delta x_{\max})$, where $\Delta x_{\max} = \max_{i=1}^{I}(\Delta x_i)$.

**The Parts of the PIC Cycle, Like in the Uniform Case**

Parts of the cycle:

(1) Pusher

$$\frac{dx_p}{dt} = v_p, \frac{dv_p}{dt} = F_p = \frac{e_p}{m_p} E(x_p), \tag{5.196}$$

with $p = 1, \ldots, P$ are the particles in the cycle and $q_p = \frac{e_p}{m_p}$ is the charge of the particle $p$.
The numerical scheme is given as a second order in time for one particle to time $t_{k+1}$:

$$x_{k+1} = x_k + \Delta t \; v_{k+1/2}, \tag{5.197}$$

$$v_{k+1/2} = v_{k-1/2} + 2\Delta t \; q \; E_k, \tag{5.198}$$

with $p = 1, \ldots, P$ are the particles in the cycle.

(2) Interpolation I (particle position to grid)

$$\rho_i = \sum_{j=1}^{J} q_j S(x_i - x_j(t_{k+1})), \tag{5.199}$$

where $x_j(t_{k+1})$ and $q_j(t_{k+1})$ are the position and charges of particle $j$ to time $t_{k+1}$.

(3) Solver and Interpolation II (grid to particle position)

$$E(x(t_{k+1})) = \sum_{i=1}^{I} E_i S(x_i - x(t_{k+1})), \tag{5.200}$$

and

$$E(x(t_{k+1})) = \sum_{i=1}^{I} \left( \sum_{k=1}^{K} g_{ik} \rho_k \right) S(x_i - x(t_{k+1})), \tag{5.201}$$

$$E(x(t_{k+1})) = \sum_{i=1}^{I} \left( \sum_{k=1}^{K} g_{ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1})) S(x_i - x(t_{k+1})) \right). \tag{5.202}$$

**Theorem 5.9** *For one PIC cycle, with CIC as Interpolation and second order in space for the solver and second order in time for the pusher, we assume:*

- *Local Error of the Interpolation I:*

$$||E(x) - E(x_i)|| \leq O(\Delta x_{max}), \tag{5.203}$$

- *Local Error of the Solver scheme:*

$$||g_{exact} - g_{i,k}|| \leq O(\Delta x_{\max}^2), \tag{5.204}$$

- *Local Error of the Interpolation II:*

$$||\rho(x) - \rho(x_i)|| \leq O(\Delta x_{\max}), \tag{5.205}$$

- *Local Error of the Pusher scheme (Time-integrator):*

$$||E(x(t_{k+1})) - E(x, t_{k+1})|| \leq O(\Delta t^2), \tag{5.206}$$

*where $\Delta x_{max}$ is the maximal grid size of the adaptive grid.*

*Then the local error estimates is given as*

$$err_{local,PIC} = ||E - E_{num}|| \leq O(\Delta x_{\max}) + O(\Delta t^2), \tag{5.207}$$

*where E is the exact electrical field and $E_{num}$ the numerical approximated electrical field with $\Delta x$ as spatial grid size and $\Delta t$ as time step.*

*Proof* We use the same arguments as for the uniform case and deal with the following time and space error estimates:

$$\begin{aligned} &||E(x(t_{k+1})) - E(x_{approx}, t_{k+1}) + E(x_{approx}, t_{k+1}) - E(x_i, t_{k+1})|| \\ &\quad \leq ||E(x(t_{k+1})) - E(x_{approx}, t_{k+1})|| + ||E(x_{approx}, t_{k+1}) - E(x_i, t_{k+1})||. \end{aligned} \tag{5.208}$$

The first part of the error estimates is the approximation error in time, while the second part is the approximation in space.

The first part is estimated based on the assumption of a second-order time-integration scheme:

$$||E(x(t_{k+1})) - E(x, t_{k+1})|| \leq O(\Delta t^2). \tag{5.209}$$

The second part is given as:

$$\begin{aligned} &||E(x_{approx}, t_{k+1}) - E(x_i, t_{k+1})|| \\ &\quad = || \left( \sum_{k=1}^{K} g_{,exact,ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1})) \right) \\ &\qquad - \left( \sum_{k=1}^{K} g_{ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1})) \right) || \leq O(\Delta x_{\max}^2). \end{aligned} \tag{5.210}$$

$$||E(x, t_{k+1}) - E(x_{approx}, t_{k+1})|| \leq O(\Delta x_{\max}). \tag{5.211}$$

We combine all the results and obtain the local error estimates as

$$err_{local,PIC} = ||E - E_{num}|| \leq O(\Delta x_{\max}) + O(\Delta t^2), \tag{5.212}$$

In the next subsection, we taken into account the underlying errors, if we consider the constraints of the self-forces, means that we should not have additional forces from the numerical scheme, see [58].

#### 5.4.3.3 Error Estimates for the Self-force in a Full PIC Cycle for Adaptive Schemes

We have the following theorem for the error estimates of the self-forces:

**Theorem 5.10** *For one PIC cycle, with CIC as Interpolation and second order in space for the solver and second order in time for the pusher, we assume*

*Then the local error estimates is given as*

$$\begin{aligned} err_{local,PIC,self-force} &= ||F_{self} - F_{self,num}|| \\ &\leq O(\Delta x_{\max}) + O(\alpha_{\max} - \alpha_{\min}) + O(\Delta t^2), \end{aligned} \tag{5.213}$$

*where $F_{self}$ is the exact electrical field for the self-force and $F_{self,num}$ is the numerical approximated electrical field for the self-force with $\Delta x_{max}$ as spatial grid size and $\Delta t$ as time step.*

*The error of the numerical scheme related to the non-translation invariant solver scheme (used as a constraint to the self-force) is given as*

$$||g_{ik,adapt} - g_{ki,adapt}|| \leq O(\alpha_{\max} - \alpha_{min}). \tag{5.214}$$

*Proof* We use the same arguments as for the uniform case and deal with the following time and space error estimates:

$$\begin{aligned} &||F_{self,exact}(x(t_{k+1})) - F_{self,corrected}(x, t_{k+1}) \\ &\quad + F_{self,corrected}(x, t_{k+1}) - F_{self,num}(x_i, t_{k+1})|| \\ &\leq ||F_{self,exact}(x(t_{k+1})) - F_{self,corrected}(x, t_{k+1})|| \\ &\quad + ||F_{self,corrected}(x, t_{k+1}) - F_{self,num}(x_i, t_{k+1})||. \end{aligned} \tag{5.215}$$

The first part of the error estimates is the approximation error in time, while the second part is the approximation in space.

The first part is estimated based on an assumed correct $g_{ik,correct}$ such that the error is only related to the numerical errors of the spatial grid

$$\begin{aligned} &||F_{self,exact}(x(t_{k+1})) - F_{self,corrected}(x, t_{k+1})|| \\ &||eE_{exact}(x(t_{k+1})) - eE_{corrected}(x, t_{k+1})|| \leq O(\Delta t^2) + O(\Delta x_{\max}). \end{aligned} \tag{5.216}$$

The second part is given as the self-force error, while the approximated numerical solution

$$\begin{aligned} &||F_{self,corrected}(x, t_{k+1}) - F_{self,num}(x_i, t_{k+1})|| \\ &= ||eE_{corrected}(x, t_{k+1}) - eE_{num}(x_i, t_{k+1})|| \leq O(\alpha_{\max} - \alpha_{\min}). \end{aligned} \tag{5.217}$$

We combine all the results and obtain the local error estimates as

$$\begin{aligned} err_{self,local,PIC} &= ||F_{self} - F_{self,num}|| \\ &\leq O(\Delta x_{\max}) + O(\Delta t^2) + O(\alpha_{\max} - \alpha_{\min}). \end{aligned} \tag{5.218}$$

*Remark 5.25* The error is splitted into two parts:

- Numerical approximation errors of the underlying PIC schemes, e.g. Solver, Pusher, Interpolation scheme. Such error can be reduced by applying higher order schemes, e.g. fourth-order discretization scheme for the solver.
- Constraint approximation errors (errors from the physical constraints), e.g. self-force constraint. Such errors are related to an invariance of the underlying scheme, e.g. translation invariance to the solver, see [55]. Such constraints are only fulfilled for equidistant grids and using adaptive grids neglect such invariances. To overcome such constraint errors, we have to optimize or embed the constraints to our PIC schemes, see ideas in [59, 60].

#### 5.4.3.4 Finite Difference Error Estimates for the Nonuniform Grid

Based on the nonuniform grid, we have to estimate the difference between the uniform and nonuniform error of the finite difference schemes.

The field equation is given as

$$\nabla^2 \phi_p = -\frac{\rho_p}{\varepsilon_0} \tag{5.219}$$

with Dirichlet conditions.

The optimal error estimates for the uniform grid is given for the finite difference scheme:

We have the discrete solution $\phi_{p,\Delta x} = L_{\Delta x}^{-1} \frac{\rho_{p,\Delta x}}{\varepsilon_0}$ while $\phi_{p,\Delta x}^* = R_{\Delta x} u$ is the restriction of the exact solution $\phi_p = L^{-1} \frac{\rho_p}{\varepsilon_0}$ (related to the shape functions)

$$\phi_{p,\Delta x} - \phi_{p,\Delta x}^* = L_{\Delta x}^{-1} \frac{\rho_{p,\Delta x}}{\varepsilon_0} - R_{\Delta x} \phi_p, \tag{5.220}$$

$$= L_{\Delta x}^{-1} \left( \frac{\rho_{p,\Delta x}}{\varepsilon_0} - \tilde{R}_{\Delta x} \frac{\rho_p}{\varepsilon_0} \right) - L_{\Delta x}^{-1} \left( L_{\Delta x} R_{\Delta x} - \tilde{R}_{\Delta x} L \right) \frac{\rho_p}{\varepsilon_0}. \tag{5.221}$$

**Theorem 5.11** *We have ($\phi_p \in H^2(\Omega)$ for the solution of $L\phi_p = \frac{\rho_p}{\varepsilon_0}$. The right-hand side of the equation $L_{\Delta x}\phi_{p,\Delta x} = \frac{\rho_{p,\Delta x}}{\varepsilon_0}$ is chosen with:*

$$|\left(\frac{\rho_{p,\Delta x}}{\varepsilon_0} - \tilde{R}_{\Delta x}\frac{\rho_p}{\varepsilon_0}\right)|_0 \leq C_f \Delta x^m, \tag{5.222}$$

*while m is the order of the shape function.*

*Proof* The approximation of the right-hand side to the grid is given as

$$|\rho_{p,\Delta x} - \sum_{j=1}^{J} q_j S(x_i - x_j)|_0 \leq O(\Delta x^m), \tag{5.223}$$

where $m$ is the order of the shape function, e.g. $m = 1$ for a CIC approximation.

The consistency of the uniform grid is given as

$$|L_h R_{\Delta x} - \tilde{R}_{\Delta x} L|_{-2 \leftarrow 2} \leq C \Delta x^n, \tag{5.224}$$

where $n = 2$ an the order of the discretization schemes.

**Error Between a Fine and Coarse Grid**

The error estimates between the adaptive solutions are given as

**Theorem 5.12** *We have two difference schemes given with the solutions of the equations $L_{\Delta x}\phi_{p,\Delta x} = \frac{\rho_{p,\Delta x}}{\varepsilon_0}$ and $L_{\Delta \tilde{x}}\phi_{p,\Delta \tilde{x}} = \frac{\rho_{p,\Delta \tilde{x}}}{\varepsilon_0}$ the error estimates is given as*

$$\begin{aligned} ||\phi_{p,\Delta x} - \phi_{p,\Delta \tilde{x}}||_0 &\leq \max\{\Delta x, \Delta \tilde{x}\} \sum_{i=1}^{I} \left\| \frac{\partial \phi_{p,\Delta x,i}}{\partial x}|_{\Delta x} - \frac{\partial \phi_{p,\Delta \tilde{x},i}}{\partial x}|_{\Delta \tilde{x}} \right\|_0 \\ &\leq C \max\{\Delta x, \Delta \tilde{x}\}^2 ||\phi_{p,\min\{\Delta x, \Delta \tilde{x}\}}||_0, \end{aligned} \tag{5.225}$$

*while I is the number of the grid cells in the coarse discretization.*

*Proof* We apply the idea of the finer grid as a reference solution. The difference to the coarse grid is given with respect to the consistency and right hand side error, see also [61].

*Remark 5.26* To apply the error estimates at the adaptive interface, we have to assume an error bound $\varepsilon$ and compute:

$$\sum_{i=1}^{I} \left\| \frac{\partial \phi_{p,\Delta x,i}}{\partial x}|_{\Delta x} - \frac{\partial \phi_{p,\Delta \tilde{x},i}}{\partial x}|_{\Delta \tilde{x}} \right\|_0 \leq E, \tag{5.226}$$

if $E \leq \varepsilon$, then we are in the tolerance of the error estimates, if $E \geq \varepsilon$, then we have to deal with a finer discretization.

*Remark 5.27* To apply such an error estimates, we have to assume an error bound, e.g. $err = 10^6$.

Then, we accept the coarsening,

if we have

$$\max\{\Delta x, \Delta\tilde{x}\} \sum_{i=1}^{I} \left\| \frac{\partial \phi_{p,\Delta x,i}}{\partial x}|_{\Delta x} - \frac{\partial \phi_{p,\Delta\tilde{x},i}}{\partial x}|_{\Delta\tilde{x}} \right\|_0 \leq err, \tag{5.227}$$

else we apply the finer grids.

### 5.4.4 Absolute Error Based on the Initialization and Right-Hand Side

The absolute errors are given with respect to the initialization and right-hand side values.

**Theorem 5.13** *For one PIC cycle, with CIC as Interpolation and second order in space for the solver and second order in time for the pusher, we assume:*

- *Local Error of the Interpolation I:*

$$||E(x) - E(x_i)|| \leq C_4 \Delta x_{\max} ||E_{init}||, \tag{5.228}$$

- *Local Error of the Solver scheme:*

$$||g_{exact} - g_{i,k}|| \leq C_3 \Delta x_{\max}^2 ||\rho_{init}||, \tag{5.229}$$

- *Local Error of the Interpolation II:*

$$||\rho(x) - \rho(x_i)|| \leq C_3 \Delta x_{\max} ||\rho_{init}||, \tag{5.230}$$

- *Absolute Error of the Pusher scheme (Time-integrator):*

$$||E(x(t_{k+1})) - E(x, t_{k+1})|| \leq C_2 \Delta t^2) ||E(x(t_0))||, \tag{5.231}$$

*where $\Delta t$ is the local time step and $t_0$ is the starting time, we have $t_k = k\Delta t + t_0$.*

*Then the local error estimates is given as*

$$err_{local,PIC} = ||E - E_{num}|| \leq C_1 \Delta x \rho_{init} + C_2 \Delta t^2 E(x(t_0)), \tag{5.232}$$

*where $\rho(x, t_0)$ is initial charge density and the exact electrical field and $E_{num}$ the numerical approximated electrical field with $\Delta x$ as spatial grid size and $\Delta t$ as time step.*

*Proof* We use the same arguments as for the uniform case and deal with the following time and space error estimates:

$$\begin{aligned}&||E(x(t_{k+1})) - E(x_{approx}, t_{k+1}) + E(x_{approx}, t_{k+1}) - E(x, t_{k+1})|| \\ &\quad \leq ||E(x(t_{k+1})) - E(x_{approx}, t_{k+1})|| + ||E(x_{approx}, t_{k+1}) - E(x, t_{k+1})||. \end{aligned} \tag{5.233}$$

The first part of the error estimates is the approximation error in time, while the second part is the approximation in space.

The first part is estimated based on the assumption of a second-order time-integration scheme:

$$\begin{aligned}&||E(x(t_{k+1})) - E(x, t_{k+1})|| \\ &\quad = ||D_{exact}E(x(t_k)) - D_{num}E(x, t_k)|| \\ &\quad = ||\sum_{\nu=1}^{k} D_{exact}^{k-\nu}(D_{exact} - D_{num})E(x(t_0))|| \leq C\Delta t^2||E(x(t_0))||, \end{aligned} \tag{5.234}$$

where $D_{exact}$ is the exact time-integrator and $D_{num}$ is the numerical time-integrator and the convergence is given as $||D_{exact} - D_{num}|| \leq O(\Delta t^2)$.

The second part is given as

$$\begin{aligned}&||E(x_{approx}, t_{k+1}) - E(x_i, t_{k+1}) + E(x_i, t_{k+1}) - E(x, t_{k+1})|| \\ &\quad = ||\left(\sum_{k=1}^{K} g_{,exact,ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1}))\right) \\ &\qquad - \left(\sum_{k=1}^{K} g_{ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1}))\right) \\ &\qquad + \left(\sum_{k=1}^{K} g_{ik} \sum_{\tilde{i}=1}^{J} q_j S(x_{\tilde{i}} - x_j(t_{k+1}))\right) \\ &\qquad - \left(\sum_{k=1}^{K} g_{ik}\rho_i\right)|| \leq (C_2\Delta x_{\max}^2 + C_3\Delta x_{\max})\rho_{init}, \end{aligned} \tag{5.235}$$

where $\rho_{init} = \rho(x_i, t^{k+1})$ for all $i \in I$.

### 5.4.5 Error Reduction with Respect to SPDE (Stochastic Partial Differential Equations)

Based on the perturbations and statistical influence of the fields with respect to the densities, the PIC algorithm, based on the Pusher, Solver and Approximation parts, can be rewritten to stochastic partial differential equations.

We can obtain in a first approximation the method as a stochastic heat equation (or in the stationary case as a elliptic stochastic equation)

$$dX - \Delta X dt = dW, \text{ in } \Omega \times \mathbb{R}^+, \tag{5.236}$$

$$X = 0, \text{ on } \partial\Omega \times \mathbb{R}^+, \tag{5.237}$$

$$X(\cdot, 0) = X_0, \text{ in } \Omega, \tag{5.238}$$

where $\Omega \subset \mathbb{R}^d$ is a convex polygonal domain and $\Delta = \sum_{k=1}^{d} \frac{\partial}{\partial x_k}$ is the Laplace operator. We have $H = L_2(\Omega)$ with the usual norm $||\cdot||$ and scalar product $\langle \cdot, \cdot \rangle$. Further $W$ is a $Q$-Wiener process or a Gaussian white noise.

For all convergence studies the approximation of the white noise and the necessary regularity have to be done, see

$$\eta_i = \frac{1}{\sqrt{\Delta x}} \int_{x_i}^{x_{i+1}} dW(x, t), \ i = 1, \ldots, N, \tag{5.239}$$

i.e. $\eta_i \approx N(0, 1)$

Example for a finite difference scheme (e.g. Method of lines with second-order two for the diffusion operator), we obtain an error estimates (only for the spatial discretized operators), see [62]:

$$\left( E \left[ \frac{1}{N} \sum_{j=1}^{N} (\hat{u}(x_j) - \hat{u}_j)^2) \right] \right)^{1/2} \leq \frac{C}{1 - \lambda_N^2} \Delta x \approx \mathcal{O}(\sqrt{\Delta x}), \tag{5.240}$$

where $\frac{1}{1-\lambda_N^2} \approx \sqrt{\Delta x}$ (approximation with respect to the regularity of the white noise).

#### 5.4.5.1 Reduction of the Error Estimates for the Local PIC

The influence of the statistical error based on the reformulated model problem as stochastic partial differential operator reduce also the error estimates of the local PIC.

**Theorem 5.14** *For PIC cycle, we apply the underlying model of a stochastic PDE. Therefore the influence to the solver scheme is given as*
*Local Error of the Solver scheme:*

$$||E(g_{exact} - g_{i,k})|| \leq O((\Delta x)^{1-\beta}), \tag{5.241}$$

*where* $\beta \in (0, 1]$.

*Then the local error estimates is given as*

$$err_{local,PIC} \leq O((\Delta x)^{1-\beta}) + O\left(\frac{\Delta t^2}{(\Delta x)^{\beta}}\right), \tag{5.242}$$

*where for the standard finite difference scheme* $\beta = 1/2$.

*Proof* The proof for the elliptic SPDE can be found in [62]. At least we reduce the error estimates with respect to the regularity of the Wiener process $dW$.

### 5.4.6 Algorithmic Ideas to Overcome the Self-Force Problems

In the adaptive grids, we cannot fulfil the invariance of the discretization schemes and have the following issues problems:

- Exact Potential, Self-force not zero: We compute with the exact Greens function or with the exact adaptive discretization schemes, but then we have an error in the self-forces: $||G_{i,j} - G_{i-s,j-s}|| \leq O(\alpha_{\max} - \alpha_{\min})$. That means the error corresponds to the difference of the maximal and minimal grid size.
- Non-Exact Potential, Self-force zero: We correct the exact Greens function or exact adaptive democratization schemes with the underlying shape functions and obtain an invariance of the discretization schemes (means Self-force is zero), but then we have an error between the exact and corrected Greens function or discretization scheme: $||G_{i,j} - G_{correct,i,j}|| \leq O(\max(\Delta x, \Delta \tilde{x}))$, where $\Delta x$ and $\Delta \tilde{x}$ are the different grid sizes.

We propose the following balanced scheme, between small self-forces and small error in the approximated discretization schemes.

Based on the work of Tskhakaya [57], we have the following potential generated by some particles located at $X$:

$$\phi(x) = e \sum_{i=1}^{m} S(x_i - X) G(x - x_i), \tag{5.243}$$

where $G(x - X)$ is the Greens function.

We expand $G(x - x_i)$ near $x - X$

$$\begin{aligned}\phi(x) &= e\sum_{i=1}^{m} S(x_i - X)G(x - X) + e\sum_{i=1}^{m} S(x_i - X)\sum_{n=1}^{\infty}\frac{(X - x_i)^n}{n!}\frac{\partial^n G(x - X)}{\partial x^n},\\ &= eG(x - X) + \delta\phi(x),\\ &= \phi_{exact}(x) + \delta\phi(x),\end{aligned} \tag{5.244}$$

while the first term represents the correct physical potential, the second term is an unphysical part based on the weighting. The term gets as small as possible for higher order shape functions which fulfils

$$\sum_{i=1}^{m} S(x_i - X)(x_i - X)^n = 0, \tag{5.245}$$

Based on the adaptive discretization schemes, we have the problem of the invariance of the Greens function, see [55].

The following problem is given:

$$E_d^0 = E_{adapt,d}^0 + E_{adapt,unphysical}^0, \tag{5.246}$$

while the second term goes to zero for $n \to \infty$, means with higher order shape functions, we have the problem of the not fulfilled invariance of the Greens function:

$$G_{i,j} \neq G_{i-\Delta x, j-\Delta\tilde{x}}, \tag{5.247}$$

while $\Delta x \neq \Delta\tilde{x}$.

To fulfil the invariance of the discretization, we perturb the Greens function, such that we have

$$G_{correct,i,j} = G_{correct,i-\Delta x, j-\Delta\tilde{x}}, \tag{5.248}$$

while $\Delta x \neq \Delta\tilde{x}$.

Therefore, we have the following corrections:

$$E_d^0 = E_{adapt,d}^0 + E_{adapt,correct,d}^0 - E_{adapt,correct,d}^0 + E_{adapt,unphysical}^0, \tag{5.249}$$

while of $E_{adapt,d}^0 + E_{adapt,correct,d}^0$ it correspond the perturbed Greens function $G_{correct,i,j}$.

Here we have to optimize the perturbation of the approximated Greens function via a variational minimization problem, see [63–67].

We have to minimize the error based on the energy norm:

$$||E||_0^2 = \int_{\Omega_i} (Q_x)\, dx \leq err, \tag{5.250}$$

where $Q_x = \frac{\partial E}{\partial x}$ and $E = G_{i,j} - G_{correct,i,j}$.

$G_{i,j}$ is the exact Greens function and $G_{correct,i,j}$ is the approximated and corrected Greens function.

#### 5.4.6.1 Balance of the Adaptive Errors

In the following, we discuss the minimization problem in the following steps:

- Uniform Grid,
- Adaptive Grid,
- Correction and Minimization of the adaptive grid errors.

**Uniform Grid**

In the uniform grid, we have the following case for the potential, when applying discrete schemes and shape functions:

$$\phi_{uniform}(x) \ = \phi_{exact}(x) + \delta\phi_{uniform}(x), \tag{5.251}$$

while $\delta\phi_{uniform}(x)$ are the unphysical potential due to the numerical schemes and we assume

$$\delta\phi_{uniform}(x) \to 0, \tag{5.252}$$

if we apply higher order discretization schemes and higher momentum shape functions.

Further for uniform discretization schemes, we have fulfilled the invariance of the solver (here especially for the Greens function):

$$g_{i,j} = g_{i-\Delta x, j-\Delta x}, \tag{5.253}$$

while based on this constraint, the self-forces are zero, see [55].

**Adaptive Grid**

In the adaptive grid, we have for the potential the same idea as for the uniform case, when applying discrete schemes and shape functions

$$\phi_{adaptive}(x) \ = \phi_{exact}(x) + \delta\phi_{adaptive}(x), \tag{5.254}$$

while $\delta\phi_{adaptive}(x)$ are the unphysical potential due to the numerical schemes of the adaptive errors and we assume

$$\delta\phi_{adaptive}(x) \to 0, \tag{5.255}$$

if we apply higher order discretization schemes and higher momentum shape functions.

Further for adaptive discretization schemes, we have an additional error, while we deal with solver schemes applied to non-symmetric grids (here especially for the Greens function):

$$g_{i,j} = g_{i-\Delta_i x, j-\Delta_j x}, \tag{5.256}$$

and $\Delta_i x \neq \Delta_j x$, and therefore based on this constraint, the self-forces are not zero, see [55] and the error is given as

$$||g_{i,j} - g_{i-\Delta_i x, j-\Delta_j x}|| \leq O(\max_{i=1,\ldots,I}(\alpha_i)), \tag{5.257}$$

while $\Delta_i x = \alpha_i \Delta x$ and $\alpha_i \in \mathbb{R}^+,\ i = 1, \ldots, I$, $I$ are the number of cells and $\Delta x$ is the uniform grid size.

**Minimization of the Adaptive Grid Errors**

To minimize the adaptive grid errors with respect to the self-force constrain, we have to balance with the shape functions.

$$g_{i,j} = g_{i-\Delta_i x, j-\Delta_j x} + O\left(\frac{\partial^m}{\partial x^m} g_{i,j}\right), \tag{5.258}$$

and are derivatives of the Greens functions $\frac{\partial^m}{\partial x^m} g_{i,j}$, while $m = 2, 4, \ldots$ is an even number.

For this case, we can fulfil the constraint to be zero.

To apply such we have to correct the potential as

$$\phi_{adaptive}(x) \quad = \phi_{exact}(x) + \delta\phi_{correct}(x) - \delta\phi_{correct}(x) + \delta\phi_{adaptive}(x), \tag{5.259}$$

while $\delta\phi_{correct}(x)$ is used to fulfil the constraints and we assume $\delta\phi_{adaptive}(x) \to 0$ for higher order discretization schemes and shape functions.

By the way, now we have to deal with minimization problem:

- Minimization of the unphysical potential: $||\delta\phi_{correct}(x)|| \to 0$.
- Minimization of the error in the invariance of the solver $||g_{i,j} - g_{i-\Delta_i x, j-\Delta_j x}|| \to 0$.

The idea is to shift the error to higher order moments of the shape functions, e.g. $m = 2, 4, \ldots$ and apply higher order shape functions. With the higher order shape functions, we can correct the potential and fulfil the self-forces.

By the way, we have to be sure, that a balance of the errors are necessary and that we obtain an average of the errors in long-term computations (statistical averaging).

Such ideas to smear out the error in the statistical manner is discussed in the next subsection.

### 5.4.7 Absolute and Statistical Errors

In the following, we contribute an absolute error based on the idea to deal with uniform and nonuniform grids.

While the error of the uniform grid are socalled reference solutions, the adaptive or nonuniform grids are the numerical solutions.

Based on the fact, that we have statistical errors, we have to define expected values $E$ and their variance $Var$. The $\sqrt{Var}$ is assumed to be the error of the expected values.

**Proposition 5.1** *We assume to have a series of discrete computed the potential $\phi$ and electric field $E$:*
$\phi_1^1, \ldots, \phi_m^n$
*and*
$E_1^1, \ldots, E_m^n$
*Then the statistical error is given as*

$$\delta\tilde{\phi} \leq \sum_{i=1}^{m} \sqrt{Var(\tilde{\phi}_i)}, \tag{5.260}$$

$$\delta\tilde{E} \leq \frac{m}{\min_{i=1}^{m}(\Delta x_i)} \max_{i=1}^{m}(\sqrt{Var(\tilde{\phi}_i)}). \tag{5.261}$$

*Proof* We have the following numerical dates for discrete computed the potential *phi*:
$\phi_1^1, \ldots, \phi_m^n$
while $i = 1, \ldots, m$ are the spatial coordinates and $j = 1, \ldots, n$ are the time coordinates.

The expected value is given as

$$E(\tilde{\phi}_i) = \sum_{j=1}^{n} \frac{1}{n} \phi_i^j, \tag{5.262}$$

we assume the same probability of $\frac{1}{n}$.

So that the variance is given as

$$Var(\tilde{\phi}_i) = \sum_{j=1}^{n} \frac{1}{n} (\phi_i^j)^2 - (E(\tilde{\phi}_i))^2, \tag{5.263}$$

we assume to have independent random variables $\phi_i^j$.

The statistical error is given as:

$$\delta\tilde{\phi}_i = \sqrt{Var(\tilde{\phi}_i)}. \tag{5.264}$$

Further we have the following numerical dates for discrete computed the E-field $E$:

$E_1^1, \ldots, E_m^n$

while $i = 1, \ldots, m$ are the spatial coordinates and $j = 1, \ldots, n$ are the time coordinates.

The expected value is given as

$$E(\tilde{E}_i) = \sum_{j=1}^{n} \frac{1}{n} E_i^j, \tag{5.265}$$

we assume the same probability of $\frac{1}{n}$ and $E = f(\phi)$ in the discrete notations, the function $f$ is given as

$E_i^j = \frac{\phi_{i+1}^j - \phi_{i-1}^j}{2\Delta x}$ for uniform grids

$E_i^j = \frac{\tilde{\phi_{i+1}^j} - \phi_{i-1}^j}{2\Delta x_{i-1}}$ and $\tilde{\phi_{i+1}^j} = \frac{\Delta x_i - \Delta x_{i-1}}{\Delta x_i}\phi_i^j + \frac{\Delta x_{i-1}}{\Delta x_i}\phi_{i+1}^j$ for corrected adaptive interfaces

$E_i^j = \frac{\tilde{\phi_{i+1}^j} - \phi_{i-1}^j}{\Delta x_{i-1} + \Delta x_{i+1}}$ for uncorrected adaptive interfaces

So that the variance is given as

$$Var(\tilde{E}_i) = Var(f(\tilde{\phi}_i)) \leq \frac{m}{\min_{i=1}^m(\Delta x_i)} \max_{i=1}^m(Var(\tilde{\phi}_i)), \tag{5.266}$$

The statistical error is given as

$$\delta\tilde{E}_i = \sqrt{Var(\tilde{E}_i)}, \tag{5.267}$$

The statistical a posteriori error estimates are given as

**Proposition 5.2** *We assume to have a two different series of discrete computed the potential $\phi_{uniform}, \phi_{adaptive}$ and electric field $E_{uniform}, E_{adaptive}$, where we have the spatial and time coordinates*

$$\phi_{uniform} = (\phi_{uniform,1}^1, \ldots, \phi_{uniform,m}^1, \ldots, \phi_{uniform,1}^n, \ldots, \phi_{uniform,m}^n)^t,$$
$$\phi_{adaptive} = (\phi_{adaptive,1}, \ldots, \phi_{adaptive,m}, \ldots, \phi_{adaptive,1}, \ldots, \phi_{adaptive,m})^t,$$

*and*

$$E_{uniform} = (E^1_{uniform,1}, \ldots, E^1_{uniform,m}, \ldots, E^1_{uniform,1}, \ldots, E^1_{uniform,m})^t,$$
$$E_{adaptive} = (E^1_{adaptive,1}, \ldots, E^1_{adaptive,m}, \ldots, E^n_{adaptive,1}, \ldots, E^n_{adaptive,m})^t.$$

*Then the statistical a posteriori error estimates is given as*

$$\delta(\tilde{\phi}_{uniform} - \tilde{\phi}_{adaptive}) \leq \sum_{i=1}^{m} \sqrt{Var(\tilde{\phi}_{uniform,i} - Var(\tilde{\phi}_{adaptive,i})}, \tag{5.268}$$

$$\delta(\tilde{E}_{uniform} - \tilde{E}_{adaptive}) \leq \frac{m}{\min_{i=1}^{m}(\Delta x_i)} \max_{i=1}^{m}(\sqrt{Var(\tilde{\phi}_{uniform,i} - Var(\tilde{\phi}_{adaptive,i})}). \tag{5.269}$$

*Proof* See the ideas in the proof of Proposition 5.1 with respect to the additivity of the expected values and variances.

*Remark 5.28* The expectation $E(\phi_{uni} - \phi_{adapt})$ is the underlying sensitivity, which relates the adaptive solution to the uniform (reference) solution.

For $E(\phi_{uni} - \phi_{adaptiv}) \approx 0$ we have only small derivations and the errors are small.

For $E(\phi_{uni} - \phi_{adaptiv}) \gg 0$ the derivations are large and the errors to the uniform grid high.

For larger computations it is often not possible to compute an uniform (reference) solution. Here, we compare adaptive solutions, with different fine grids $\Delta x$, $\Delta x/2, \ldots$, such that we allow to derive an numerical error rate to the used adaptive grids.

If the error is strongly variating from one to another grid, we assume that we did not reach a convergent solution and refine more in the underlying regions, since our error is small enough.

### 5.4.8 Scaling of the Error and Analytical Error

In the last section, we have derived the statistical errors of the uniform and adaptive solutions.

We apply the statistical error estimate

$$\delta\tilde{E}_{k,uniform} = \frac{1}{2\Delta x}(\tilde{\phi}_{k+1,uniform} + \tilde{\phi}_{k-1,uniform}) \tag{5.270}$$

$$\delta\tilde{E}_{k,no-corr,adaptive} = \frac{1}{\Delta x_{k-1} + \Delta x_{k+1}}(\tilde{\phi}_{k+1,adaptive} + \tilde{\phi}_{k-1,adaptive}) \tag{5.271}$$

$$\delta\tilde{E}_{k,corr,adaptive} = \frac{\Delta x_k - \Delta x_{k-1}}{2\Delta x_{k-1}\Delta x_k}\tilde{\phi}_{k,adaptive} + \frac{1}{2\Delta x_k}\tilde{\phi}_{k+1,adaptive}\frac{1}{2\Delta x_{k-1}}\tilde{\phi}_{k-1,adaptive}, \tag{5.272}$$

Further we assume based on the discretization and solver schemes that the discretization is scaled quadratic (quadratical error) $\mathcal{O}(\Delta x^2)$ and the solver is linear scaled (linear error) $\mathcal{O}(\Delta x)$. Based on the consecutive application of discretization and solver schemes, the errors are scaled linear $\mathcal{O}(\Delta x)$.

**Assumption 5.15** The potential errors are linear:

$$|\tilde{\phi}_{k,adaptive}| \leq \mathcal{O}(\Delta x), \tag{5.273}$$

$$|\tilde{\phi}_{k,uniform}| \leq \mathcal{O}(\Delta x), \tag{5.274}$$

where the different grid-scales are given as $\Delta x_k = \Delta x\alpha_k$, and $\alpha_k \in (0, 1]$. $\Delta x$ is the uniform grid-scale.

**Proposition 5.3** *The errors between the uniform and adaptive solutions are scales with* $\mathcal{O}(\Delta\alpha_{\max})$ *and* $\Delta\alpha_{\max} = \max_{k=1}^{K}\alpha_k - 1$, *while* $1, \ldots, K$ *are the grid points.*

*Proof* We have to estimate the error of the different grids:

$$\begin{aligned} &|\delta\tilde{E}_{k,uniform} - \delta\tilde{E}_{k,no-corr,adaptive}| \\ &\leq |\frac{1}{\Delta x} - \frac{1}{\Delta x_{k-1} + \Delta x_{k+1}}|\mathcal{O}(\Delta x_{\max}), \\ &\leq \mathcal{O}(\Delta\alpha_{\max}), \end{aligned} \tag{5.275}$$

where $\Delta\alpha_{\max} = \max_{k=1}^{K}\alpha_k - 1$.

The same can be done with the corrected adaptive version.

#### 5.4.8.1 Analytical Errors

The analytical error is based on a 1D solution at the interface of the different grids.

We assume different grids for the solver, which can be estimated as errors based on the statistical variance.

The pusher can be reformulated as oscillator of the errors of the solved Poisson's equation:

$$\frac{\partial^2\varepsilon}{\partial t^2} = -\omega^2\varepsilon \tag{5.276}$$

while $\omega$ is the frequency of the oscillator and $\varepsilon_k = x_k - X_k$, where $x_k$ is the analytical and $X_k$ the numerical solution of the pusher at the grid point $k$.

By deriving the truncation error of the pusher, we have the Taylor expansion of the underlying scheme

$$-\frac{e}{m}E(X_k+\varepsilon_k)-E(X_k) = -\frac{e}{m}\varepsilon_k\frac{\partial E}{\partial x}|_{X_k+\Delta x)} \approx -\frac{e}{m}\varepsilon_k|\frac{\partial E}{\partial x}|_{\max} \quad (5.277)$$

Such an estimation of the maximum of the divergence of the $E$-field is the Laplacian of the potential, means the maximum errors of the potential.

$$\frac{\partial^2\varepsilon}{\partial t^2} = -\omega^2\varepsilon \quad (5.278)$$

where $\omega^2 = \frac{e}{m}|\frac{\partial E}{\partial x}|_{\max}$.

While the solution of the oscillator is given as

$$\varepsilon = \cos(\omega\Delta t) \approx= 1 - \frac{(\omega\Delta t)^2}{2}, \quad (5.279)$$

Now we compare the different grids of the Poisson's equation

**Proposition 5.4** *The errors uniform and adaptive solutions are given as* $|\frac{\partial E}{\partial x}|_{max,uniform}$ *and* $|\frac{\partial E}{\partial x}|_{max,adaptive}$*,*
*then the analytical error of the different oscillator grids are given as:*

$$\Delta\varepsilon = |\frac{(\omega_{uniform}\Delta t)^2}{2} - \frac{(\omega_{adaptive}\Delta t)^2}{2}|, \quad (5.280)$$

*where* $\omega^2_{uniform} = \frac{e}{m}|\frac{\partial E}{\partial x}|_{max,uniform}$ *and* $\omega^2_{adaptive} = \frac{e}{m}|\frac{\partial E}{\partial x}|_{max,adaptive}$*.*

*Proof* The results of the different error estimates are inserted into the oscillator.

### *5.4.9 Numerical Results*

In the following, we discuss a microscopic Test problem of one particle to see the error for the different methods.

Therefore, we separate dominant errors, e.g. energy error of non-conservation, which arose of many particle systems. Here, we carefully discuss the single error for smaller systems:

- 1 Electron : We test only the self-force (should be zero)
- 1 Electron and 1 Ion: We test the self-force and inter-particle force (all the error should be skipped)

**Table 5.2** Experimental setup of the one-particle problem

| | |
|---|---|
| Domain $\Omega$ | Total domain length $= 100\lambda_{De}$ |
| Fine grid size | $\Delta x_1 = 0.1\lambda_{De}$ |
| Interface (domain cut) | $x_{inter} = 50\lambda_{De}$ |
| Coarsening | $\Delta x_1 \rightarrow 50\Delta x_1 = \Delta x_2$ |
| Time step | $\Delta t = 0.002$ |
| End point | $t_{\max} = 30$ |
| Averaging | $t_{start} = 10$, $out = 0.2$ , $\Delta t_{ion} = 1$ |
| Boundary conditions | $\Phi(0) = 0,\ \Phi(L) = \Phi_{analytic}(x_e)$ |
| Initialization | El. starts at $x = 50\lambda_{De}$, $v0 = 0 * V_{th,e}$ |

**Table 5.3** Numerical errors of the adaptive grid without corrections

| $\Delta_x 2/\Delta_x 1$ | $err_{uncorr}$ | $err_{corr}$ | $abserr_{uncorr}$ | $abserr_{corr}$ |
|---|---|---|---|---|
| x-Coordinate | y-Coordinate | y-Coordinate | y-Coordinate | y-Coordinate |
| 1 | 0 | 0 | 0 | 0 |
| 2 | −2.683191 | 3.572447e-13 | 2.683191 | 3.572447e-13 |
| 4 | −4.874832 | 0 | 4.874832 | 0 |
| 6 | −5.826364 | 0 | 5.826364 | 0 |
| 10 | −6.697762 | −1.786227e-12 | 6.697762 | 1.786227e-12 |
| 20 | −7.428628 | −3.572447e-13 | 7.428628 | 3.572447e-13 |
| 50 | −7.903834 | 3.572447e-13 | 7.903834 | 3.572447e-13 |

Here we have the following experimental setup, see Table 5.2.

The one-particle problem is done with 1 electron with $v = 0$ is placed at cut-gridpoint (fine–coarse grid interface) and we have a wrong E-Field, based on the standard discretization. Later we switch to the correction and apply a logical uniform grid to correct the wrong E-field.

In the following, we present the relative and absolute errors of the one-particle experiment with standard FD and corrected FD schemes, see Table 5.3.

*Remark 5.29* Based on the uncorrected scheme (standard Finite Difference scheme), we have the problem of reduction of the suggested convergence order of $\mathcal{O}(h^{2\beta})$ to a lower order of $\mathcal{O}(h^{\beta}\log(h))$, where $0 < \beta \leq 1$. For example we will obtain $\beta = \frac{1}{2}$ at the interface for deterministic PDEs. Here we deal with stochastic PDEs, therefore we lose convergence order, see [68, 69].

## 5.4.10 Conclusion

We discuss the multiscale problems of using adaptive grids for Particle in Cell methods. While the PIC methods are constructed for uniform meshes based on the correct

spatial symmetry of the field solvers, see [55, 57], we perturbed the symmetry with adaptive meshes and apply non-symmetric schemes (or adaptive schemes). We derive error estimates to see the errors in the adaptive scheme, e.g. error in the self-force. Taken into account the physical assumption in the decreasing charge density, we can reduce and nearly circumvent such errors. In the numerical examples, we could present the advantages of the adaptive schemes and reduce the computational costs. A detailed study of a so-called microscopic problem with one and two particles allows a more precise analysis of the errors. Such that we can apply many particle problems and understand their different errors.

## 5.5 A Multicomponent Transport Model for Plasma and Particle Transport: Multicomponent Mixture

**Abstract** In this paper we present a model based on a multicomponent transport regime. Such models can be applied in plasma simulations, e.g. a local thermodynamic equilibrium, weakly ionized plasma mixture. Such plasma mixtures are applied in medical sterilization and technical etching processes, see [70]. A further application of multicomponent models are applications in complex fluid problem, e.g. viscoelasticity equations (based on the Oldroyd B constitutive equation) related with multicomponent fluid flow, see [71]. While the most classical description of the diffusion phenomenon is based on the Fickian's approach, see [72, 73], we discuss here the more detailed Maxwell–Stefan model, which covers the binary reciprocal interactions of the gas molecules, see [74, 75]. Such a more detailed description resolves into a system of coupled nonlinear partial differential equations and the diffusion is more complex as in the Fickian's approach. Here, we present a ternary gas mixture and study the problems of the numerical approaches. We present a explicit solver methods and discuss more improved results of the mixtures.

### *5.5.1 Introduction*

Multicomponent transport models are nowadays important to understand complex fluid mixtures, e.g. plasma transport, multiscale modelling of fluids, population dynamics, complex fluids, etc., see [76, 77]. While diffusion is a time-dependent process, which has his origin by the motion of the species that spread in space, the understanding of such process is important for the modelling. We assume to have diffusion driven processes such that classical description, like Fickian's approach failed.

The Fickian's approach has the underlying idea, that fluxes from regions with high concentrations go to regions of low concentrations, while their magnitude is proportional to the gradients, see [72, 73]. Such a direct relation between flux and

concentration gradient is some models sufficient but failed by more delicate multicomponent mixtures, see [78, 79]. The more extended model is the Maxwell–Stefan approach, see [74, 75], which allow to model the advanced phenomenas, e.g. cross-diffusion model. The idea is to see the processes as binary reciprocal interactions of the gas molecules and results in more complex coupled equation systems as the phenomenological approach with Fickian's law.

Because of a nonlinear coupled differential equation system, it is more delicate to solver such evolution equations, see [80]. We have to overcome additional constraints, e.g. total sum of the diffusive fluxes are zero. This additional contribution results have taken into account by the inversion of the flux–force relation. Here, we can extend and stabilize the inversion and obtain a well-posedness of the Stefan–Maxwell equations, see [80].

In smaller mixture regimes, e.g. 3 or 4 species, we could also reformulate the problem into an equation system of 2 or 3 species. Here, we apply the condition of the summation of the mole fractions $\sum_{i=1}^{n} \xi_i = 1$ to the equation system and reformulate into a smaller system of $(n-1)$-species without the constraints, see also [81].

For the numerical schemes, we deal with such a reformulation in a lower dimensional coupled nonlinear equation system and apply the discretization and linearisation schemes.

In the following, we discuss a complicate plasma model, which can be applied for plasma mixture problems.

## *5.5.2 Mathematical Model for Plasma Mixture Problem*

In the following, we discuss a multicomponent transport model, which can be applied in plasma simulations. The model is motivated in the literature for multicomponent simulation models, see [82, 83].

Often in the modelling, we discuss the different scale regimes. We consider the Knudsen Number, which is the ratio of the mean free path $\lambda$ over the typical domain size $L$.

We have the different model regimes:

- For small Knudsen Numbers $Kn \approx 0.01$ we deal with a Navier–Stokes equation,
- and for large Knudsen Numbers $Kn \geq 1.0$ we deal with the Boltzmann equations.

In the first section we describe the modelling of the plasma, where we use the velocity in the impulse conservation for the transport of the species.

### 5.5.2.1 Plasma Model for Atmospheric Regimes

The model assumes that the neutral particles can be described as fluid-dynamical model, where the elastic collision define the dynamics and few inelastic collisions are, among other reasons, responsible for the chemical reactions.

To describe the individual mass densities, as well as the global momentum and the global energy as the dynamical conservation quantities of the system, corresponding conservation equations are derived from Boltzmann equations.

The individual character of each species is considered by mass conservation equations and the so-called difference equations.

The extension of the non-mixtured multicomponent transport model, [82] is done with respect to the collision integrals related to the right-hand side sources of the conservation laws.

The conservation laws of the neutral elements are given as

$$\begin{aligned}
\frac{\partial}{\partial t}\rho_s + \frac{\partial}{\partial \boldsymbol{r}} \cdot \rho_s \boldsymbol{u}_s &= m_s Q_n^{(s)}, \\
\frac{\partial}{\partial t}\rho \boldsymbol{u} + \frac{\partial}{\partial \boldsymbol{r}} \cdot \left( \underline{\underline{P}}^* + \rho \boldsymbol{u}\boldsymbol{u} \right) &= -Q_m^{(e)}, \\
\frac{\partial}{\partial t}\mathscr{E}_{\text{tot}}^* + \frac{\partial}{\partial \boldsymbol{r}} \cdot \left( \mathscr{E}_{\text{tot}}^* \boldsymbol{u} + \boldsymbol{q}^* + \underline{\underline{P}}^* \cdot \boldsymbol{u} \right) &= -Q_{\mathscr{E}}^{(e)},
\end{aligned}$$

where $\rho_s$ is the density of species $i$, with the total density $\rho = \sum_{i=1}^{N} \rho_i$ and $\boldsymbol{u}$ is the velocity. Further $\mathscr{E}_{\text{tot}}^*$ is the total energy of the neutral particles.

Further the variable $Q_n^{(s)}$ is the collisional term of the mass conservation equation, $Q_m^{(e)}$ is the collisional term of the momentum conservation equation and $Q_{\mathscr{E}}^{(e)}$ is the collisional term of the energy conservation equation.

We derive the collisional term with respect to the Chapmen–Enskog method, see [84], and achieve for the first derivates the following results:

$$m_s Q_n^{(s)} = -\nabla \cdot \left( \rho_i \sum_{j=0} \mathbf{V}_i^j \right), \tag{5.281}$$

$$Q_m^{(e)} = -\sum_{i=1}^{n_s} \rho_i F_i, \tag{5.282}$$

$$Q_{\mathscr{E}}^{(e)} = -\sum_{i=1}^{n_s} \rho_i \rho F_i \left( \mathbf{u} + \sum_{j=0} V_i^{(j)} \right), \tag{5.283}$$

where $i = 1, \ldots, n_s$, $F_i$ is an external force per unit mass (see Boltzmann equation), further the diffusion velocity is given as

$$\mathbf{V}_i^0 = 0, \tag{5.284}$$

$$\mathbf{V}_i^1 = -\sum_{j=1}^{N} D_{ij} \left( d_j + k_{T_j} \frac{\Delta T}{T} \right), \tag{5.285}$$

where $\sum_{i=1}^{N} d_i = 0$,

$$d_i = \nabla x_i + x_i \frac{\nabla p}{p} - \frac{\rho_i}{\rho} F_i, \tag{5.286}$$

$$d_i = d_i - y_i \sum_j d_j^*, \tag{5.287}$$

where $x_i = \frac{n_s}{n}$ is the molar fraction of species $i$.

We have an additional constraint based on the mass fraction of each species

$$\frac{\partial}{\partial t} y_i + \nabla y_i = R_i(y_1, \ldots, y_N), \tag{5.288}$$

where $y_i$ is the mass fraction of species $i$, $R_i$ is the net production rate of species $i$ due to his reactions.

*Remark 5.30* The model problem contains conservation equations and constraints related to the material properties, e.g. mass fraction which chanced by the reactions. Both equation parts, means conservation equation (which are related to macroscopic scales) and the constraint equation (which are related to microscopic scales) result into the multiscale model.

#### 5.5.2.2 Simplified Model of a Ternary Mixture Based on the Maxwell–Stefan Diffusion Equation

We simplified the delicate plasma model to a three-component gas mixture, e.g. hydrogen $H_2$ (species 1), nitrogen $N_2$ (species 2) and carbon oxide $CO_2$ (species 3), see also [85, 86]. We could also modify such a model equation to the multicomponent flow problem of an etching process of a reactive plasma, e.g. with oxygen $O_2$, chloride $Cl$ and nitrogen $N_2$, or other gaseous species.

We concentrate on the three-component system, which are first introduced, and solve such a system as a linear optimal problem (General Linear Optimal Problem). We deal with:

$$\partial_t \xi_i + \nabla \cdot N_i = 0, \ 1 \le i \le 3, \tag{5.289}$$

$$\sum_{j=1}^{3} N_j = 0, \tag{5.290}$$

$$\frac{\xi_2 N_1 - \xi_1 N_2}{D_{12}} + \frac{\xi_3 N_1 - \xi_1 N_3}{D_{13}} = -\nabla \xi_1, \tag{5.291}$$

$$\frac{\xi_1 N_2 - \xi_2 N_1}{D_{12}} + \frac{\xi_3 N_2 - \xi_2 N_3}{D_{23}} = -\nabla \xi_2, \tag{5.292}$$

where the domain is given as $\Omega \in \mathbb{R}^d$, $d \in \mathbb{N}^+$ with $\xi_i \in C^2$.

We could reduce to a simpler model problem as

$$\partial_t \xi_i + \nabla \cdot N_i = 0, \ 1 \leq i \leq 2, \tag{5.293}$$

$$\frac{1}{D_{13}} N_1 + \alpha N_1 \xi_2 - \alpha N_2 \xi_1 = -\nabla \xi_1, \tag{5.294}$$

$$\frac{1}{D_{23}} N_2 - \beta N_1 \xi_2 + \beta N_2 \xi_1 = -\nabla \xi_2, \tag{5.295}$$

where $\alpha = \left(\frac{1}{D_{12}} - \frac{1}{D_{13}}\right)$, $\beta = \left(\frac{1}{D_{12}} - \frac{1}{D_{23}}\right)$.

#### 5.5.2.3 Maxwell–Stefan Diffusion Equation as an Optimal Control Problem

From the mathematical point of view, the coupled equation system (5.293)–(5.295) can be seen as an optimal control problem. We rewrite the model equation (5.293)–(5.295) to a set of $s$ linearized states $U_0, U_1, \ldots, U_s$ by the linear system:

$$U'_{i+1} = J_i(t) U_{i+1} + \hat{B}(t) v, \tag{5.296}$$

where $J_i$ is the Jacobian of $B(U, t)$ and given in (5.296), the control operator is $\hat{B}(t) = \tilde{B}(t) - J_i$, and the system input is $v = U_i$.

Then, we can now apply the idea of a GLCS (general linear control system), see [87], using the following notations: $u = U_{i+1}$, $v = U_i$, $A_1(t) = J_i(t)$, $A_2(t) = \tilde{B}(t)$.

The GLCS is given as

$$\frac{du}{dt} = A_1(t) u + A_2(t) v, \tag{5.297}$$

$$\tilde{u} = C(t) u + D(t) v, \tag{5.298}$$

$$u(0) = u_0, \tag{5.299}$$

where the time-dependent operators are $A(t) \in \mathbf{X}^n \times \mathbf{X}^n$, $B(t) \in \mathbf{X}^n \times \mathbf{X}^m$, $C(t) \in \mathbf{X}^p \times \mathbf{X}^n$, $D(t) \in \mathbf{X}^p \times \mathbf{X}^m$, $v : \mathbf{X} \to \mathbf{X}^m$ denotes the system input, $\tilde{u} : \mathbf{X} \to \mathbf{X}^p$ is the system output and $u : \mathbf{X} \to \mathbf{X}^n$ denotes the state vector. Furthermore, $\mathbf{X}$ is an appropriate Banach space, e.g. $U$, a space of continuous or piecewise continuous functions.

The analytical solution of (5.297) and (5.298) is

$$u(t) = \exp\left(\int_0^t A_1(s) ds\right) u_0 + \int_0^t \exp\left(\int_s^t A_1(\tilde{s}) d\tilde{s}\right) A_2(s) v(s) ds, \tag{5.300}$$

$$\begin{aligned} \tilde{u}(t) = {} & C(t) \ \exp\left(\int_0^t A_1(s) ds\right) u_0 \\ & + \ C(t) \int_0^t \exp\left(\int_s^t A_1(\tilde{s}) d\tilde{s}\right) A_2(s) v(s) ds + D(t) v(t), \end{aligned} \tag{5.301}$$

where we apply the fast computation of the exponential integral matrices via the Magnus expansion, see [88–90], and discussed in the following.

*Remark 5.31* The rewriting into a control system can be important for large equation systems. Here, we concentrate on studying a ternary system and apply direct method as for example explicit or implicit time-discretization schemes.

### 5.5.3 Numerical Experiments

In the following, we deal with a ternary mixture, see [85], which simulates the mixture of three gaseous species.

We concentrate on the three component system:

$$\partial_t \xi_i + \partial_x N_i = 0, \ 1 \le i \le 3, \tag{5.302}$$

$$\sum_{j=1}^{3} N_j = 0, \tag{5.303}$$

$$\frac{\xi_2 N_1 - \xi_1 N_2}{D_{12}} + \frac{\xi_3 N_1 - \xi_1 N_3}{D_{13}} = -\partial_x \xi_1, \tag{5.304}$$

$$\frac{\xi_1 N_2 - \xi_2 N_1}{D_{12}} + \frac{\xi_3 N_2 - \xi_2 N_3}{D_{23}} = -\partial_x \xi_2, \tag{5.305}$$

where the spatial domain is given as $\Omega \in \mathbb{R}^d$, $d \in \mathbb{N}^+$, the time domain is given as $[0, T] \in \mathbb{R}_0^+$, while the solution is given in a sufficient smooth space, e.g. with $\xi_i \in C^2(\Omega \times [0, T])$.

The parameters and the initial and boundary conditions are given as

- $D_{12} = D_{13} = 0.833$ (means $\alpha = 0$) and $D_{23} = 0.168$ (Uphill diffusion, semi-degenerated Duncan and Toor experiment)
- $D_{12} = 0.0833, D_{13} = 0.680$ and $D_{23} = 0.168$ (asymptotic behaviour, Duncan and Toor experiment)
- $J = 140$ (spatial grid points)
- The time step restriction for the explicit method is given as $\Delta t \le \frac{(\Delta x)^2}{2 \max\{D_{12}, D_{13}, D_{23}\}}$
- The spatial domain is $\Omega = [0, 1]$, the time domain $[0, T] = [0, 1]$
- The initial conditions are:

1. Uphill example

$$\xi_1^{in}(x) = \begin{cases} 0.8 & \text{if } 0 \le x < 0.25 \\ 1.6(0.75 - x) & \text{if } 0.25 \le x < 0.75 \\ 0.0 & \text{if } 0.75 \le x \le 1.0 \end{cases}, \tag{5.306}$$

$$\xi_2^{in}(x) = 0.2, \ \text{for all } x \in \Omega = [0, 1], \tag{5.307}$$

2. Diffusion example (Asymptotic behaviour)

$$\xi_1^{in}(x) = \begin{cases} 0.8 \text{ if } 0 \leq x \in 0.5 \\ 0.0 \text{ else} \end{cases}, \tag{5.308}$$

$$\xi_2^{in}(x) = 0.2, \text{ for all } x \in \Omega = [0, 1]. \tag{5.309}$$

- The boundary conditions are of no-flux type:

$$N_1 = N_2 = N_3 = 0, \text{ on } \partial\Omega \times [0, 1], \tag{5.310}$$

We could reduce to a simpler model problem as

$$\partial_t \xi_i + \partial_x \cdot N_i = 0, \ 1 \leq i \leq 2, \tag{5.311}$$

$$\frac{1}{D_{13}} N_1 + \alpha N_1 \xi_2 - \alpha N_2 \xi_1 = -\partial_x \xi_1, \tag{5.312}$$

$$\frac{1}{D_{23}} N_2 - \beta N_1 \xi_2 + \beta N_2 \xi_1 = -\partial_x \xi_2, \tag{5.313}$$

where $\alpha = \left(\frac{1}{D_{12}} - \frac{1}{D_{13}}\right)$, $\beta = \left(\frac{1}{D_{12}} - \frac{1}{D_{23}}\right)$.

We rewrite into:

$$\partial_t \xi_1 + \partial_x \cdot N_1 = 0, \tag{5.314}$$

$$\partial_t \xi_2 + \partial_x \cdot N_2 = 0, \tag{5.315}$$

$$\begin{pmatrix} \frac{1}{D_{13}} + \alpha\xi_2 & -\alpha\xi_1 \\ -\beta\xi_2 & \frac{1}{D_{23}} + \beta\xi_1 \end{pmatrix} \begin{pmatrix} N_1 \\ N_2 \end{pmatrix} = \begin{pmatrix} -\partial_x \xi_1 \\ -\partial_x \xi_2 \end{pmatrix}, \tag{5.316}$$

and we have

$$\partial_t \xi_1 + \partial_x \cdot N_1 = 0, \tag{5.317}$$

$$\partial_t \xi_2 + \partial_x \cdot N_2 = 0, \tag{5.318}$$

$$\begin{pmatrix} N_1 \\ N_2 \end{pmatrix} = \frac{D_{13} D_{23}}{1 + \alpha D_{13} \xi_2 + \beta D_{23} \xi_1} \begin{pmatrix} \frac{1}{D_{23}} + \beta\xi_1 & \alpha\xi_1 \\ \beta\xi_2 & \frac{1}{D_{13}} + \alpha\xi_2 \end{pmatrix} \begin{pmatrix} -\partial_x \xi_1 \\ -\partial_x \xi_2 \end{pmatrix}. \tag{5.319}$$

The next step is to apply the semi-discretization of the partial differential operator $\frac{\partial}{\partial x}$.

We apply the first differential operator in Eqs. (5.317) and (5.318) as an forward upwind scheme given as

$$\frac{\partial}{\partial x} = D_+ = \frac{1}{\Delta x} \cdot \begin{pmatrix} -1 & 0 \dots & & 0 \\ 1 & -1 & 0 \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & & 1 & -1 & 0 \\ 0 & \dots & 0 & 1 & -1 \end{pmatrix} \in \mathbb{R}^{(J+1)\times(J+1)}, \tag{5.320}$$

and the second differential operator in Eq. (5.319) as an backward upwind scheme given as

$$\frac{\partial}{\partial x} = D_- = \frac{1}{\Delta x} \cdot \begin{pmatrix} -1 & 1 & 0 \dots & 0 \\ 0 & -1 & 1 & 0 \dots \\ \vdots & \ddots & \ddots & \ddots & \ddots \\ 0 & \dots & 0 & -1 & 1 \\ 0 & & \dots & 0 & -1 \end{pmatrix} \in \mathbb{R}^{(J+1)\times(J+1)}. \tag{5.321}$$

*Remark 5.32* We decided to apply finite difference scheme for the spatial discretization. One could also apply a variational formulation, e.g. finite element or finite volume schemes, see for example [91, 92]. Such a notation allows to apply the finite matrices $D_+$ and $D_-$ as abstract operators for the next step in the time-discretization schemes.

### 5.5.4 Iterative Scheme in Time (Global Linearization, Matrix Method)

We propose a iterative scheme to resolve the nonlinearity of the equation system with respect to the local time step. Means, we linearize the equation system with respect to the small time step. Based on the explicit time discretization (explicit Euler method), we are restricted by the CFL condition and therefore small time step approaches are necessary such that we overcome the nonlinear behaviour, see [14]. We solve the iterative scheme:

$$\xi_1^{n+1} = \xi_1^n - \Delta t \; D_+ N_1^n, \tag{5.322}$$

$$\xi_2^{n+1} = \xi_2^n - \Delta t \; D_+ N_2^n, \tag{5.323}$$

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} N_1^{n+1} \\ N_2^{n+1} \end{pmatrix} = \begin{pmatrix} -D_- \xi_1^{n+1} \\ -D_- \xi_2^{n+1} \end{pmatrix}, \tag{5.324}$$

for $j = 0, \ldots, J$ , where $\xi_1^n = (\xi_{1,0}^n, \ldots, \xi_{1,J}^n)^T$, $\xi_2^n = (\xi_{2,0}^n, \ldots, \xi_{2,J}^n)^T$ and $I_J \in \mathbb{R}^{J+1} \times \mathbb{R}^{J+1}$, $N_1^n = (N_{1,0}^n, \ldots, N_{1,J}^n)^T$, $N_2^n = (N_{2,0}^n, \ldots, N_{2,J}^n)^T$ and $I_J \in \mathbb{R}^{J+1} \times \mathbb{R}^{J+1}$, where $n = 0, 1, 2, \ldots, N_{end}$ and $N_{end}$ are the number of time steps, i.d. $N_{end} = T/\Delta t$.

The matrices are given as

$$A, B, C, D \in \mathbb{R}^{J+1} \times \mathbb{R}^{J+1}, \tag{5.325}$$

$$A_{j,j} = \frac{1}{D_{13}} + \alpha \xi_{2,j},\ j = 0 \ldots, J, \tag{5.326}$$

$$B_{j,j} = -\alpha \xi_{1,j},\ j = 0 \ldots, J, \tag{5.327}$$

$$C_{j,j} = -\beta \xi_{2,j},\ j = 0 \ldots, J, \tag{5.328}$$

$$D_{j,j} = \frac{1}{D_{23}} + \beta \xi_{1,j},\ j = 0 \ldots, J, \tag{5.329}$$

$$A_{i,j} = B_{i,j} = C_{i,j} = D_{i,j} = 0,\ i, j = 0 \ldots, J,\ i \neq J, \tag{5.330}$$

means the diagonal entries given as for the scale case in Eq. (5.319) and the outer-diagonal entries are zero.

The explicit form with the time-discretization is given as:

**Algorithm 5.16** (1) Initialization $n = 0$:

$$\begin{pmatrix} N_1^0 \\ N_2^0 \end{pmatrix} = \begin{pmatrix} \tilde{A} & \tilde{B} \\ \tilde{C} & \tilde{D} \end{pmatrix} \begin{pmatrix} -D_- \xi_1^0 \\ -D_- \xi_2^0 \end{pmatrix}, \tag{5.331}$$

where $\xi_1^0 = (\xi_{1,0}^0, \ldots, \xi_{1,J}^0)^T$, $\xi_2^0 = (\xi_{2,0}^0, \ldots, \xi_{2,J}^0)^T$ and $\xi_{1,j}^0 = \xi_1^{in}(j\Delta x)$, $\xi_{2,j}^0 = \xi_2^{in}(j\Delta x)$, $j = 0, \ldots, J$ and given as for the different intializations, we have

(1.1) Uphill example

$$\xi_1^{in}(x) = \begin{cases} 0.8 & \text{if } 0 \leq x < 0.25 \\ 1.6(0.75 - x) & \text{if } 0.25 \leq x < 0.75 \\ 0.0 & \text{if } 0.75 \leq x \leq 1.0 \end{cases}, \tag{5.332}$$

$$\xi_2^{in}(x) = 0.2,\ \text{for all } x \in \Omega = [0, 1], \tag{5.333}$$

(1.2) Diffusion example (Asymptotic behaviour)

$$\xi_1^{in}(x) = \begin{cases} 0.8 \text{ if } 0 \leq x \in 0.5, \\ 0.0 \text{ else}, \end{cases} \tag{5.334}$$

$$\xi_2^{in}(x) = 0.2,\ \text{for all } x \in \Omega = [0, 1], \tag{5.335}$$

The inverse matrices are given as

$$\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D} \in \mathbb{R}^{J+1} \times \mathbb{R}^{J+1}, \tag{5.336}$$

$$\tilde{A}_{j,j} = \gamma_j \left( \frac{1}{D_{23}} + \beta \xi^0_{1,j} \right),\ j = 0 \ldots, J, \tag{5.337}$$

$$B_{j,j} = \gamma_j\ \alpha \xi^0_{1,j},\ j = 0 \ldots, J, \tag{5.338}$$

$$C_{j,j} = \gamma_j\ \beta \xi^0_{2,j},\ j = 0 \ldots, J, \tag{5.339}$$

$$D_{j,j} = \gamma_j \left( \frac{1}{D_{13}} + \alpha \xi^0_{2,j} \right),\ j = 0 \ldots, J, \tag{5.340}$$

$$\gamma_j = \frac{D_{13} D_{23}}{1 + \alpha D_{13} \xi^0_{2,j} + \beta D_{23} \xi^0_{1,j}},\ j = 0 \ldots, J, \tag{5.341}$$

$$\tilde{A}_{i,j} = \tilde{B}_{i,j} = \tilde{C}_{i,j} = \tilde{D}_{i,j} = 0,\ i, j = 0 \ldots, J,\ i \neq J, \tag{5.342}$$

Further the values of the first and the last grid points of $N$ are zero, means $N^0_{1,0} = N^0_{1,J} = N^0_{2,0} = N^0_{2,J} = 0$ (boundary condition).

(2) Next timesteps (till $n = N_{end}$ ):

(2.1) Computation of $\xi^{n+1}_1$ and $\xi^{n+1}_2$

$$\xi^{n+1}_1 = \xi^n_1 - \Delta t\ D_+ N^n_1, \tag{5.343}$$

$$\xi^{n+1}_2 = \xi^n_2 - \Delta t\ D_+ N^n_2, \tag{5.344}$$

(2.2) Computation of $N^{n+1}_1$ and $N^{n+1}_2$

$$\begin{pmatrix} N^{n+1}_1 \\ N^{n+1}_2 \end{pmatrix} = \begin{pmatrix} \tilde{A} & \tilde{B} \\ \tilde{C} & \tilde{D} \end{pmatrix} \begin{pmatrix} -D_- \xi^{n+1}_1 \\ -D_- \xi^{n+1}_2 \end{pmatrix}, \tag{5.345}$$

where $\xi^n_1 = (\xi^n_{1,0}, \ldots, \xi^n_{1,J})^T$, $\xi^n_2 = (\xi^n_{2,0}, \ldots, \xi^n_{2,J})^T$ and the inverse matrices are given as

$$\tilde{A}, \tilde{B}, \tilde{C}, \tilde{D} \in \mathbb{R}^{J+1} \times \mathbb{R}^{J+1}, \tag{5.346}$$

$$\tilde{A}_{j,j} = \gamma_j \left( \frac{1}{D_{23}} + \beta \xi^{n+1}_{1,j} \right),\ j = 0 \ldots, J, \tag{5.347}$$

$$B_{j,j} = \gamma_j\ \alpha \xi^{n+1}_{1,j},\ j = 0 \ldots, J, \tag{5.348}$$

$$C_{j,j} = \gamma_j\ \beta \xi^{n+1}_{2,j},\ j = 0 \ldots, J, \tag{5.349}$$

$$D_{j,j} = \gamma_j \left( \frac{1}{D_{13}} + \alpha \xi^{n+1}_{2,j} \right),\ j = 0 \ldots, J, \tag{5.350}$$

$$\gamma_j = \frac{D_{13}D_{23}}{1 + \alpha D_{13}\xi_{2,j}^{n+1} + \beta D_{23}\xi_{1,j}^{n+1}},\ j = 0\ldots, J, \tag{5.351}$$

$$\tilde{A}_{i,j} = \tilde{B}_{i,j} = \tilde{C}_{i,j} = \tilde{D}_{i,j} = 0,\ i, j = 0\ldots, J,\ i \neq J. \tag{5.352}$$

Further the values of the first and the last grid points of $N$ are zero, means $N_{1,0}^n = N_{1,J}^n = N_{2,0}^n = N_{2,J}^n = 0$ (boundary condition).

(3) Do $n = n + 1$ and goto (2)

We have the following examples:

- Uphill example,
- Diffusion example (Asymptotic behaviour),

and discuss their results in different figures.

The iterative scheme is tested and we obtain convergent results with sufficient small time steps. The results of the concentrations of the three species is shown in Fig. 5.28.

*Remark 5.33* In Fig. 5.28, the concentration of the species 2 shows, that we obtain a so-called reciprocal interaction with the other species. Means the mixture induce a temporary decay of the quantity and after some time, the density tends to the expected asymptotic quantity.

The concentration and their fluxes are given in Fig. 5.29.

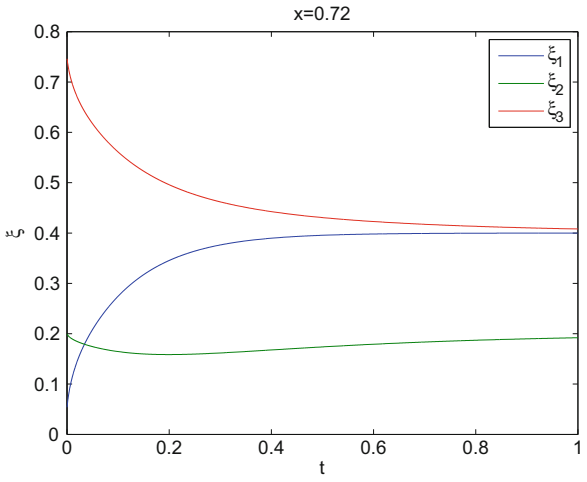



**Fig. 5.28** The figures present the results of the concentration $c_1$, $c_2$ and $c_3$

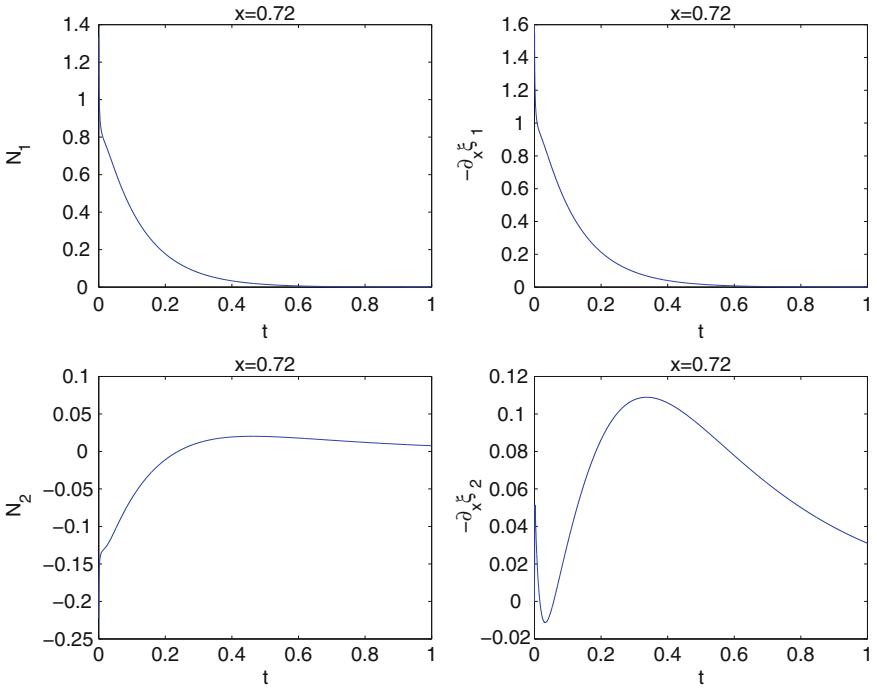



**Fig. 5.29** The *upper figures* present the results of the concentration $c_1$ and $-\partial_x \xi_1$. The *lower figures* presents the results of $c_2$ and $-\partial_x \xi_2$

*Remark 5.34* In Fig. 5.29, we see in detail, the reciprocal interaction between the species in their concentration and their fluxes. We only achieve a stationary behaviour after some time, while in the initialization, we see also fluctuations between the species 1 and 2.

The full plots in time and space of the concentrations and their fluxes are given in Fig. 5.30.

*Remark 5.35* In Fig. 5.30, we see all details in a 3D plot in the time- and spatial-scale. After the initialization, we see a convergent concentration and flux of the species 1 and 2. Such a mixture or reciprocal behaviour can also be resolved by the Maxwell–Stefan's approach, while we allow to interact the gaseous species in the mixture, see [79, 93].

The space-time regions where $(-N_2 \partial_x \xi_2) \geq 0$ for the uphill diffusion and asymptotic diffusion, given in Fig. 5.31.

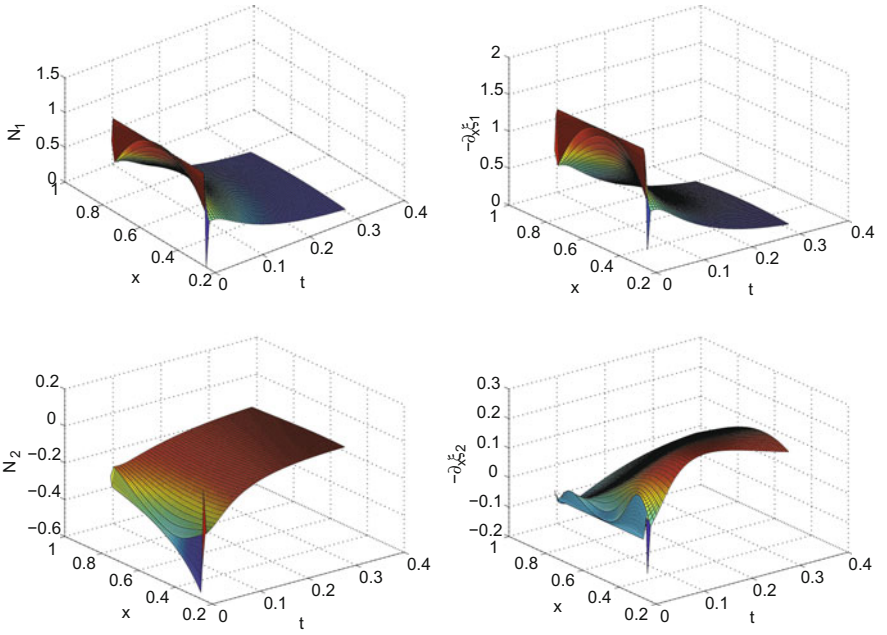



**Fig. 5.30** The figures present the results of the 3D plots in time and space. The *upper figures* present the results of the concentration $c_1$ and $-\partial_x \xi_1$. The *lower figures* presents the results of $c_2$ and $-\partial_x \xi_2$

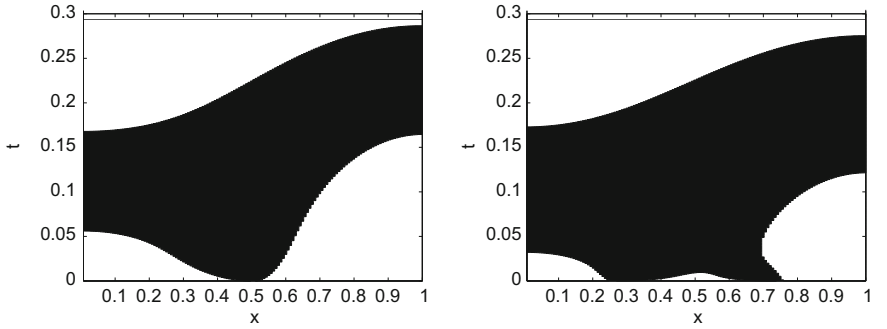



**Fig. 5.31** The figures present the asymptotic diffusion (*left-hand side*) and uphill diffusion (*right-hand side*) in the space-time region

*Remark 5.36* In Fig. 5.31, we present the behaviour of the asymptotic and uphill diffusion in a space-time region with $(-N_2\ \partial_x \xi_2) \geq 0$. We see a different topological behaviour of the two diffusion examples. The influence of the uphill diffusion areas are more active than the asymptotic diffusion behaviour. Such mixture mappings are important to understand the mixture of the three species in different regimes.

*Remark 5.37* For a numerical point of view, the drawback of the explicit schemes are the restriction by the CFL condition. To overcome such a restriction, we have

taken into account implicit methods and the application of linearization methods, e.g. Newton's method or fixpoint schemes, see also some alternative ideas in Appendix B.2.

### *5.5.5 Conclusions and Discussions*

We discussed an extension of the multicomponent models with the Maxwell–Stefan approach. Such novel models allowed a more detailed description of the diffusive processes in gaseous mixtures. We present the coupled and nonlinear model equations, which are numerically more delicate to solve. Based on a novel global iterative scheme with restriction to the time steps, we could solve such multicomponent models. In future, we can overcome such restrictions of the time step with respect to the CFL and the linerization condition with implicit schemes. Such implicit schemes allows much more larger time steps but we have to implement nonlinear solver methods. While in the experiments, we obtain more detailed information at the beginning of the mixture, we also see a stable and stationary result after the initialization. Such that it might be possible to apply such a complex model of the Maxwell–Stefan approach at the beginning of the mixture simultations, while at later timescales, we could deal with the resolved diffusion processes with standard approaches.

## 5.6 Multicomponent Model of a Full-Scale Model of Glycolysis in Saccharomyces cerevisiae: Theory and Splitting Schemes

**Abstract** We present a multicomponent model of a glycolysis pathway in saccharomyes cerevisiae, see the modelling in [94] and the algorithmic exploration in [95]. The model is based on a dynamical system with different behaviours, e.g. fats dissipative actions combined with slow dynamics on the manifolds. We deal with a large-scale model with a reaction network, means we solve a strong coupled nonlinear differential equation with highly nonlinearities and interconnections, see [95]. First we have to analyse the particular importance of the different components with respect to their dynamical behaviour and their oscillation effects, second, we have to take into account an algorithm to solve such a large-scale problem. We deal with splitting methods for the strong coupled ordinary differential equations (ODEs), while separating into different fast and efficient methods for each simpler part is an attractive point of view, see the motivation of splitting schemes in [96]. Splitting method can be applied to solve such a delicate biochemical pathway model. When we decompose to different scale-dependent equation parts, e.g. high- and slow oscillatory biochemical processes, we can decouple into different fast solvable simple and less rough scale models. We discuss different splitting approaches. Further, we present the

oscillatory behaviour of the different equation parts such that a splitting approach can be applied. We present first numerical results of the different eigenvalues of the strong coupled ODEs and their oscillatory behaviour.

### 5.6.1 Introduction

The motivation to analyse the problem arose of the idea to understand the dynamical properties of a delicate biochemistry model dealing with glycolysis. Glycolysis is an important process of classical biochemistry and it has been thoroughly studies, see [94, 97]. Such models are large-scale systems with highly nonlinear and strongly coupled effects. Based on the oscillatory behaviour of the network, see [95], it makes sense to reduce such systems and apply multiscale methods, that apply techniques to skip unnecessary highly oscillating components or average such fine scales to an upscaled system. Such a behaviour can be studied by considering the eigenvalues of the system and obtain information about the roughness of the different components, see introduction to dynamical systems [98].

The ideas to solve the underlying problem are given in the following Fig. 5.32.

We concentrate on the following subsection to the presentation of the splitting approaches.

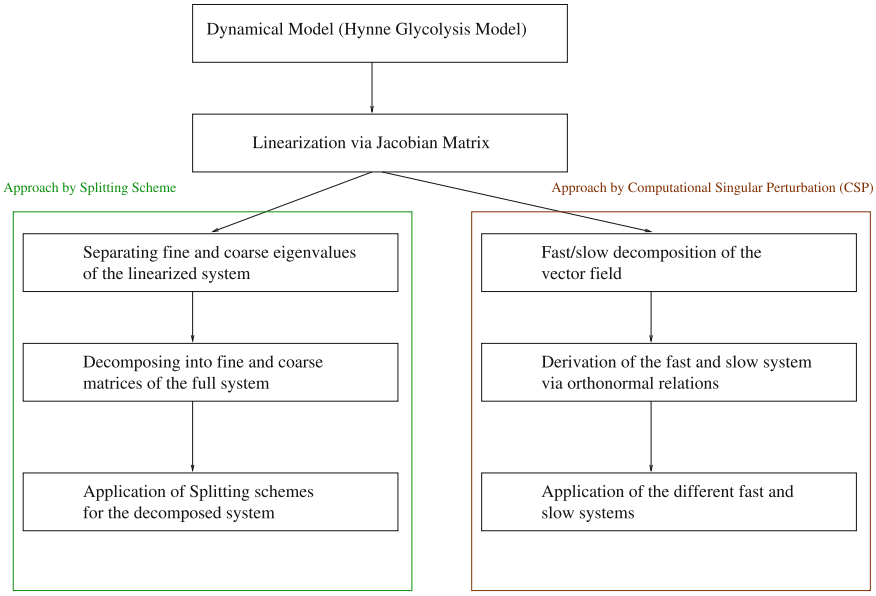



**Fig. 5.32** Multicomponent problem of glycolysis model and application of splitting schemes

### 5.6.2 Introduction to the Pathway Model for the Glycolysis in Saccharomyces cerevisiae

In the following, we have a delicate multiscale pathway model which are strong and weak coupled nonlinear ordinary differential equations.

We deal with the following nonlinear equation system.

$$\partial_t\, c = A_1(c) + \cdots + A_m(c),\ 0 < t \le T < +\infty, \tag{5.353}$$

$$c(0) = c_0, \tag{5.354}$$

where $A_1, A_m$ are given nonlinear functions $A_i : \mathbb{R}^m \to \mathbb{R}^m,\ i = 1, \ldots, m$ and $c_0 \in \mathbb{R}^m$ is a given initial vector and the unknown function is $c : [0, T) \to \mathbb{R}^m$, where $m$ is the number of species in the pathway model.

The coupling between each model.

We assume the linearized model based on the idea of

$$\begin{aligned} A_i(c) &= A_i(c_0) \\ &+ \frac{\partial A_i(c)}{\partial c}|_{c=c_0}(c - c_0) + \frac{1}{2}(c - c_0)^t H(c)|_{c=c_0}(c - c_0), \end{aligned} \tag{5.355}$$

where $H(c)_{ij} = \frac{\partial}{\partial c_i}\frac{\partial A(c)}{\partial c_j}$ are the entries of the Hessian matrix.

For simplicity, we choose $c_0 = 0$, we have

$$\tilde{A}_i c = \frac{\partial A_i(c)}{\partial c}|_{c=c_0}\, c, \tag{5.356}$$

where $\tilde{A}_i$ is the Jacobian matrix of the vectorial function $A_i(c)$ and the approximation error to the nonlinear case is given with the Hessian: $\frac{1}{2}||H(c)|_{c=c_0}||\ ||(c - c_0)^t(c - c_0)||$.

We obtain the full linearized problem as

$$\begin{aligned} \partial_t\, c &= \tilde{A}_1 c + \cdots + \tilde{A}_m c \\ &+ (I - \tilde{A}_1)c_0 + \cdots + (I - \tilde{A}_m)c_0,\ 0 < t \le T < +\infty, \end{aligned} \tag{5.357}$$

$$\partial_t\, c = \tilde{A}_1 c + \cdots + \tilde{A}_m c + f(c_0),\ 0 < t \le T < +\infty, \tag{5.358}$$

$$c(0) = c_0, \tag{5.359}$$

where $f(c_0) = (I - \tilde{A}_1)c_0 + \cdots + (I - \tilde{A}_m)c_0$.

For the numerical analysis of the homogeneous problem, we choose $c_0 = 0$.

### 5.6.3 Model for Hynne Glycolysis

The model is based on the work of [94], where we have the following equations:

$$x' = f(x), \tag{5.360}$$

where $f : \mathbb{R}^n \to \mathbb{R}^m$ is the nonlinear function with the Jacobian matrix

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}. \tag{5.361}$$

We apply the linearization via the Jacobian matrices and can study the dynamical behaviour by the linearized differential equation system:

$$y' = Jy, \tag{5.362}$$

where we apply the splitting into $J = J_1 + J_2$, where $J_1$ is the upper and $J_2$ the lower Jacobian matrix.

The equations of the Hynne glycolysis model, see [94], are given as

$$\begin{aligned} \frac{dy_1}{dt} &= \frac{597.035y_2(t)}{0.0816993y_2(t)y_3(t) + 0.833333y_3(t) + 1.17647y_2(t) + 2} - 2.832y_1(t) \\ &\quad - \frac{597.035y_1(t)}{\frac{(0.0816993y_2(t)y_3(t)+0.833333y_3(t)+0.588235y_2(t)+1)(0.588235y_1(t)+1)}{0.588235y_2(t)+1} + 0.588235y_1(t) + 1} + 52.392, \end{aligned} \tag{5.363}$$

$$\begin{aligned} \frac{dy_2}{dt} &= -\frac{51.7547y_{18}(t)y_2(t)}{y_{18}(t)y_2(t) + 0.1y_2(t) + 0.037} \\ &\quad - \frac{597.035y_2(t)}{0.0816993y_2(t)y_3(t) + 0.833333y_3(t) + 1.17647y_2(t) + 2} \\ &\quad + \frac{597.035y_1(t)}{\frac{(0.0816993y_2(t)y_3(t)+0.833333y_3(t)+0.588235y_2(t)+1)(0.588235y_1(t)+1)}{0.588235y_2(t)+1} + 0.588235y_1(t) + 1}, \end{aligned} \tag{5.364}$$

$$\begin{aligned} \frac{dy_3}{dt} &= \frac{3815.71y_4(t)}{5.33333y_4(t) + y_3(t) + 0.8} - 2.25932y_{18}(t)y_3(t) \\ &\quad - \frac{496.042y_3(t)}{5.33333y_4(t) + y_3(t) + 0.8} + \frac{51.7547y_{18}(t)y_2(t)}{y_{18}(t)y_2(t) + 0.1y_2(t) + 0.037}, \end{aligned} \tag{5.365}$$

$$\begin{aligned} \frac{dy_4}{dt} &= -\frac{45.4327y_4(t)^2}{y_4(t)^2 + 0.021\left(\frac{0.15y_{18}(t)^2}{y_{20}(t)^2} + 1\right)} \\ &\quad - \frac{3815.71y_4(t)}{5.33333y_4(t) + y_3(t) + 0.8} + \frac{496.042y_3(t)}{5.33333y_4(t) + y_3(t) + 0.8}, \end{aligned} \tag{5.366}$$

$$\frac{dy_5}{dt} = \frac{45.4327y_4(t)^2}{y_4(t)^2 + 0.021\left(\frac{0.15y_{18}(t)^2}{y_{20}(t)^2} + 1\right)}$$

$$- \frac{2207.82y_5(t)}{2.46914y_6(t)y_7(t) + 9.87654y_7(t) + y_5(t) + 0.1y_5(t)y_6(t) + 4.93827y_6(t) + 0.3} + \frac{27257.y_7(t)y_6(t)}{2.46914y_6(t)y_7(t) + 9.87654y_7(t) + y_5(t) + 0.1y_5(t)y_6(t) + 4.93827y_6(t) + 0.3}, \quad (5.367)$$

$$\frac{dy_6}{dt} = \frac{116.365y_7(t)}{y_7(t) + 0.968504y_6(t) + 1.23} - \frac{27257.y_6(t)y_7(t)}{2.46914y_6(t)y_7(t) + 9.87654y_7(t) + y_5(t) + 0.1y_5(t)y_6(t) + 4.93827y_6(t) + 0.3} - \frac{2115.73y_6(t)}{y_7(t) + 0.968504y_6(t) + 1.23} + \frac{2207.82y_5(t)}{2.46914y_6(t)y_7(t) + 9.87654y_7(t) + y_5(t) + 0.1y_5(t)y_6(t) + 4.93827y_6(t) + 0.3} - \frac{13897.6y_6(t)y_{22}(t)}{(100.y_8(t) + 1.66667y_6(t) + 1)(10.y_{22}(t) + 16.6667y_{21}(t) + 1)} + \frac{2.52684 \times 10^6 y_8(t)y_{21}(t)}{(100.y_8(t) + 1.66667y_6(t) + 1)(10.y_{22}(t) + 16.6667y_{21}(t) + 1)}, \quad (5.368)$$

$$\frac{dy_7}{dt} = -\frac{116.365y_7(t)}{y_7(t) + 0.968504y_6(t) + 1.23} - \frac{27257.y_6(t)y_7(t)}{2.46914y_6(t)y_7(t) + 9.87654y_7(t) + y_5(t) + 0.1y_5(t)y_6(t) + 4.93827y_6(t) + 0.3} - \frac{81.4797y_7(t)}{25\left(\frac{0.034(7.69231y_{22}(t)+1)}{y_{21}(t)} + 1\right) + y_7(t)\left(\frac{0.13(7.69231y_{22}(t)+1)}{y_{21}(t)} + 1\right)} + \frac{2115.73y_6(t)}{y_7(t) + 0.968504y_6(t) + 1.23} + \frac{2207.82y_5(t)}{2.46914y_6(t)y_7(t) + 9.87654y_7(t) + y_5(t) + 0.1y_5(t)y_6(t) + 4.93827y_6(t) + 0.3}, \quad (5.369)$$

$$\frac{dy_8}{dt} = -443866.y_{19}(t)y_8(t) - \frac{2.52684 \times 10^6 y_{21}(t)y_8(t)}{(100.y_8(t) + 1.66667y_6(t) + 1)(10.y_{22}(t) + 16.6667y_{21}(t) + 1)} + 1528.62y_{18}(t)y_9(t) + \frac{13897.6y_6(t)y_{22}(t)}{(100.y_8(t) + 1.66667y_6(t) + 1)(10.y_{22}(t) + 16.6667y_{21}(t) + 1)}, \quad (5.370)$$

$$\frac{dy_9}{dt} = \left(443866.y_{19}(t)y_8(t) - 1528.62y_{18}(t)y_9(t) - \frac{343.096y_{19}(t)y_9(t)}{(y_{19}(t) + 0.17)(y_9(t) + 0.2)}\right), \quad (5.371)$$

$$\frac{dy_{10}}{dt} = \left(\frac{343.096y_{19}(t)y_9(t)}{(y_{19}(t) + 0.17)(y_9(t) + 0.2)} - \frac{53.1328y_{10}(t)}{y_{10}(t) + 0.3}\right), \quad (5.372)$$

$$\frac{dy_{11}}{dt} = \left(-\frac{89.8023y_{21}(t)y_{11}(t)}{(y_{11}(t) + 0.71)(y_{21}(t) + 0.1)} - 24.7y_{11}(t) + 24.7y_{16}(t) + \frac{53.1328y_{10}(t)}{y_{10}(t) + 0.3}\right), \quad (5.373)$$

$$\frac{dy_{12}}{dt} = \left(-16.72y_{12}(t) + 16.72y_{13}(t) + \frac{89.8023y_{11}(t)y_{21}(t)}{(y_{11}(t) + 0.71)(y_{21}(t) + 0.1)}\right), \quad (5.374)$$

$$\frac{dy_{13}}{dt} = (16.72y_{12}(t) - 19.552y_{13}(t)), \quad (5.375)$$

$$\frac{dy_{14}}{dt} = \frac{81.4797y_7(t)}{25\left(\frac{0.034(7.69231y_{22}(t)+1)}{y_{21}(t)} + 1\right) + y_7(t)\left(\frac{0.13(7.69231y_{22}(t)+1)}{y_{21}(t)} + 1\right)}$$

$$- 1.9y_{14}(t) + 1.9y_{15}(t), \tag{5.376}$$

$$\frac{dy_{15}}{dt} = (1.9y_{14}(t) - 4.732y_{15}(t)), \tag{5.377}$$

$$\frac{dy_{16}}{dt} = (24.7y_{11}(t) - 27.532y_{16}(t) - 0.167459y_{16}(t)y_{17}(t)), \tag{5.378}$$

$$\frac{dy_{17}}{dt} = (-0.167459y_{16}(t)y_{17}(t) - 2.832y_{17}(t) + 15.8592), \tag{5.379}$$

$$\begin{aligned}\frac{dy_{18}}{dt} &= 133.333y_{19}(t)^2 + 443866.y_8(t)y_{19}(t) + \frac{343.096y_9(t)y_{19}(t)}{(y_{19}(t)+0.17)(y_9(t)+0.2)} \\ &\quad - 432.9y_{20}(t)y_{18}(t) - 3.2076y_{18}(t) - 2.25932y_{18}(t)y_3(t) \\ &\quad - 1528.62y_{18}(t)y_9(t) - \frac{45.4327y_4(t)^2}{y_4(t)^2 + 0.021\left(\frac{0.15y_{18}(t)^2}{y_{20}(t)^2}+1\right)} \\ &\quad - \frac{51.7547y_{18}(t)y_2(t)}{y_{18}(t)y_2(t) + 0.1y_2(t) + 0.037},\end{aligned} \tag{5.380}$$

$$\begin{aligned}\frac{dy_{19}}{dt} &= -266.666y_{19}(t)^2 - 443866.y_8(t)y_{19}(t) - \frac{343.096y_9(t)y_{19}(t)}{(y_{19}(t)+0.17)(y_9(t)+0.2)} \\ &\quad + 865.8y_{20}(t)y_{18}(t) + 3.2076y_{18}(t) + 2.25932y_{18}(t)y_3(t) \\ &\quad + 1528.62y_{18}(t)y_9(t) + \frac{45.4327y_4(t)^2}{y_4(t)^2 + 0.021\left(\frac{0.15y_{18}(t)^2}{y_{20}(t)^2}+1\right)} \\ &\quad + \frac{51.7547y_{18}(t)y_2(t)}{y_{18}(t)y_2(t) + 0.1y_2(t) + 0.037},\end{aligned} \tag{5.381}$$

$$\frac{dy_{20}}{dt} = \left(133.333y_{19}(t)^2 - 432.9y_{20}(t)y_{18}(t)\right), \tag{5.382}$$

$$\begin{aligned}\frac{dy_{21}}{dt} &= -\frac{81.4797y_7(t)}{25\left(\frac{0.034(7.69231y_{22}(t)+1)}{y_{21}(t)}+1\right) + y_7(t)\left(\frac{0.13(7.69231y_{22}(t)+1)}{y_{21}(t)}+1\right)} \\ &\quad - \frac{89.8023y_{11}(t)y_{21}(t)}{(y_{11}(t)+0.71)(y_{21}(t)+0.1)} \\ &\quad + \frac{13897.6y_6(t)y_{22}(t)}{(100.y_8(t)+1.66667y_6(t)+1)(10.y_{22}(t)+16.6667y_{21}(t)+1)} \\ &\quad - \frac{2.52684\times 10^6 y_8(t)y_{21}(t)}{(100.y_8(t)+1.66667y_6(t)+1)(10.y_{22}(t)+16.6667y_{21}(t)+1)},\end{aligned} \tag{5.383}$$

$$\begin{aligned}\frac{dy_{22}}{dt} &= \frac{81.4797y_7(t)}{25\left(\frac{0.034(7.69231y_{22}(t)+1)}{y_{21}(t)}+1\right) + y_7(t)\left(\frac{0.13(7.69231y_{22}(t)+1)}{y_{21}(t)}+1\right)} \\ &\quad + \frac{89.8023y_{11}(t)y_{21}(t)}{(y_{11}(t)+0.71)(y_{21}(t)+0.1)} \\ &\quad - \frac{13897.6y_6(t)y_{22}(t)}{(100.y_8(t)+1.66667y_6(t)+1)(10.y_{22}(t)+16.6667y_{21}(t)+1)} \\ &\quad + \frac{2.52684\times 10^6 y_8(t)y_{21}(t)}{(100.y_8(t)+1.66667y_6(t)+1)(10.y_{22}(t)+16.6667y_{21}(t)+1)},\end{aligned} \tag{5.384}$$

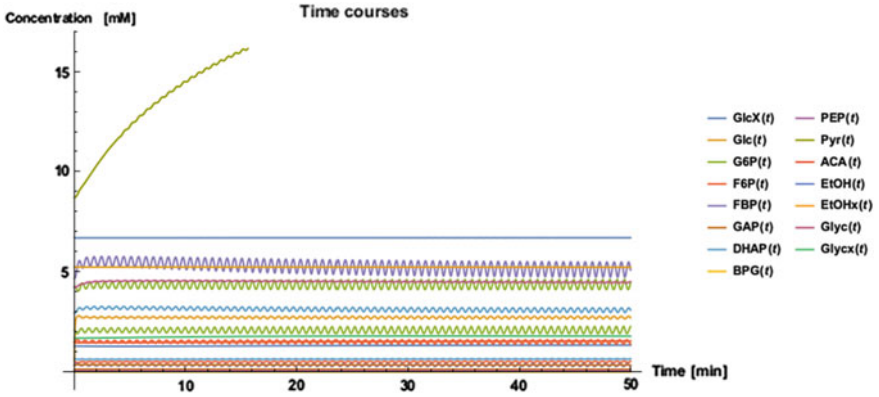



**Fig. 5.33** Time course of the different concentrations of the full system

$$\begin{aligned}\frac{dy_{23}}{dt} &= 4.51864y_3(t)y_{18}(t) + 3.2076y_{18}(t)\\ &+ \frac{81.4797y_7(t)}{25\left(\frac{0.034(7.69231y_{22}(t)+1)}{y_{21}(t)}+1\right)+y_7(t)\left(\frac{0.13(7.69231y_{22}(t)+1)}{y_{21}(t)}+1\right)}\\ &- \frac{13897.6y_6(t)y_{22}(t)}{(100.y_8(t)+1.66667y_6(t)+1)(10.y_{22}(t)+16.6667y_{21}(t)+1)}\\ &+ \frac{2.52684\times 10^6 y_8(t)y_{21}(t)}{(100.y_8(t)+1.66667y_6(t)+1)(10.y_{22}(t)+16.6667y_{21}(t)+1)}.\end{aligned} \tag{5.385}$$

We have the following notations:

$$y_1(t) = GlcX(t),\ y_2(t) = Glc(t),\ y_3(t) = G6P(t), \tag{5.386}$$
$$y_4(t) = F6P(t),\ y_5(t) = FBP(t),\ y_6(t) = GAP(t), \tag{5.387}$$
$$y_7(t) = DHAP(t),\ y_8(t) = BPG(t),\ y_9(t) = PEP(t), \tag{5.388}$$
$$y_{10}(t) = Pyr(t),\ y_{11}(t) = ACA(t),\ y_{12}(t) = EtOH(t), \tag{5.389}$$
$$y_{13}(t) = EtOHx(t),\ y_{14}(t) = Glyc(t),\ y_{15}(t) = Glycx(t). \tag{5.390}$$

The computation of the equation by a higher order ODE solver, e.g. Runge–Kutta method is given in Fig. 5.33.

Here we see the different scales of the concentrations. We have high oscillatory behaviour of the concentration: $y_1, y_2, y_3, y_4, y_{10}, y_{14}$, while the other concentrations are less oscillatory.

*Remark 5.38* Based on the different scale dependencies of the equation, we have to decompose with respect to the eigenvalues of the Jacobian matrix. We reconstruct a new equation system based on decomposing into $A = A_{low} + A_{higher}$, high oscillating equations and low oscillating equations. The solver methods, which can deal with such decomposed systems are splitting schemes.

In the following, we discuss the splitting methods, which can be applied for the decomposed operators based on the decomposition of the eigenvalues of the Jacobian matrix.

### 5.6.4 Splitting Schemes for Partitioned Multicomponent Equations

In the following, we discuss splitting approaches in a additive or multiplicative scheme.

While additive splitting schemes are interested for parallel implementation, they have their drawbacks, while they deal with lower order, see [96]. Instead multiplicative splitting schemes are interested for higher order approaches, see [44, 99, 100], but they have their drawbacks to obtain a parallel version of such a scheme, see [96].

We deal with the linearized equation system given in Eq. (5.357) and apply the following splitting schemes:

- Parallel Splitting (Additive Splitting)

$$\frac{\partial c_1^n(t)}{\partial t} = \tilde{A}_1 c_1^n(t), \quad \text{with } (n-1)\tau < t \leq n\tau,$$
$$c_1^n((n-1)\tau) = c_{sp}^N((n-1)\tau), \tag{5.391}$$
$$\vdots \tag{5.392}$$
$$\frac{\partial c_m^n(t)}{\partial t} = \tilde{A}_m c_m^n(t), \quad \text{with } (n-1)\tau < t \leq n\tau,$$
$$c_m^n((n-1)\tau) = c_{sp}^N((n-1)\tau), \tag{5.393}$$

  and the additive step :

$$c_{sp}^n(n\tau) = c_{sp}^N((n-1)\tau) + \sum_{i=1}^{m} (c_i^n(n\tau) - c_{sp}^N((n-1)\tau)),$$
$$n = 1, 2, \ldots, N, \text{ where } c_{sp}^N(0) = c_0.$$

- Multiplicative or A–B Splitting Scheme

$$\frac{\partial c_1(t)}{\partial t} = \tilde{A}_1 c_1(t), \quad \text{with } c_1(t^n) = c^n, \tag{5.394}$$
$$\frac{\partial c_2(t)}{\partial t} = \tilde{A}_2 c_2(t), \quad \text{with } c_2(t^n) = c_1(t^{n+1}), \tag{5.395}$$
$$\vdots \tag{5.396}$$
$$\frac{\partial c_m(t)}{\partial t} = \tilde{A}_m c_m(t), \quad \text{with } c_m(t^n) = c_{m-1}(t^{n+1}), \tag{5.397}$$

where the time step is $\tau^n = t^{n+1} - t^n$. The solution of equations are $c^{n+1} = c_m(t^{n+1})$.

- Symmetrically Splitting Scheme (Parallel A–B Splitting Scheme) (e.g. with two operators)

$$\frac{\partial c_1(t)}{\partial t} = \tilde{A}_1 c_1(t), \quad \text{with } c_1(t^n) = c_{sp}^n(t^n), \tag{5.398}$$

$$\frac{\partial c_2(t)}{\partial t} = \tilde{A}_2 c_2(t), \quad \text{with } c_2(t^n) = c_1(t^{n+1}), \tag{5.399}$$

where the time step is $\tau^n = t^{n+1} - t^n$.

$$\frac{\partial v_1(t)}{\partial t} = \tilde{A}_2 v_1(t), \quad \text{with } v_1(t^n) = c_{sp}^n(t^n), \tag{5.400}$$

$$\frac{\partial v_2(t)}{\partial t} = \tilde{A}_1 v_2(t), \quad \text{with } v_2(t^n) = v_1(t^{n+1}), \tag{5.401}$$

where the time step is $\tau^n = t^{n+1} - t^n$. The summation step is given as

$$c_{sp}^n(t^{n+1}) = \frac{c_2(t^{n+1}) + v_2(t^{n+1})}{2}. \tag{5.402}$$

*Remark 5.39* The implementation of fast splitting schemes for the large equation systems need to apply parallel schemes, therefore we propose additive splitting approaches. Such schemes can be applied as pure additive splitting (lower order). We could improve the order of the scheme by combining multiplicative splitting scheme, while parts of the scheme are computed in parallel.

### 5.6.5 Splitting Errors and Time Step Control

For the accuracy of the computations, it is important to estimate the errors of the underlying splitting methods. On the one side, we know the error of the scheme, on the other side, we could control the time step sizes of the underlying schemes.

The splitting errors are given in the following:

- Parallel Splitting (Additive Splitting)

$$\begin{aligned} err_{sp,parallel} &= \exp\left(\tau\left(\sum_{i=1}^{m}\tilde{A}_i\right)\right) - \left(I + \sum_{i=1}^{m}(\exp(\tau\tilde{A}_i) - I)\right), \\ &= \sum_{i=1}^{m}\sum_{j=1,j\neq i}^{m}(\tilde{A}_i\tilde{A}_j + \tilde{A}_j\tilde{A}_i)\tau^2, \end{aligned} \tag{5.403}$$

and the time step control is given as

$$\tau < \frac{1}{||\sum_{i=1}^{m}\sum_{j=1,j\neq i}^{m}(\tilde{A}_i\tilde{A}_j + \tilde{A}_j\tilde{A}_i)||}. \tag{5.404}$$

- A–B Splitting (Multiplicative Splitting)

$$\begin{aligned} err_{sp,parallel} &= \exp\left(\tau\left(\sum_{i=1}^{m}\tilde{A}_i\right)\right) - \Pi_{i=1}^{m}\exp(\tau\tilde{A}_i), \\ &= \sum_{i=1}^{m}\sum_{j=1,j\neq i}^{m}[\tilde{A}_i,\tilde{A}_j]\tau^2, \\ &= \sum_{i=1}^{m}\sum_{j=1,j\neq i}^{m}(\tilde{A}_i\tilde{A}_j - \tilde{A}_j\tilde{A}_i)\tau^2, \end{aligned} \tag{5.405}$$

and the time step control is given as

$$\tau < \frac{1}{||\sum_{i=1}^{m}\sum_{j=1,j\neq i}^{m}[\tilde{A}_i,\tilde{A}_j]||}. \tag{5.406}$$

- Symmetric A–B Splitting (parallel Splitting, with 2 Operators)

$$\begin{aligned} err_{sp,A-Bparallel} &= \exp\left(\tau\left(\sum_{i=1}^{2}\tilde{A}_i\right)\right) \\ &\quad - \frac{1}{2}(\exp(\tau\tilde{A}_2)\exp(\tau\tilde{A}_1) + \exp(\tau\tilde{A}_1)\exp(\tau\tilde{A}_2)) \\ &= \sum_{i=1}^{m}\sum_{j=1,j\neq i}^{m}[\tilde{A}_i,\tilde{A}_j]\tau^2||, \\ &= \left(\left(\frac{1}{12}[\tilde{A}_1,[\tilde{A}_1,\tilde{A}_2]] + \frac{1}{12}[\tilde{A}_2,[\tilde{A}_2,\tilde{A}]]\right)\right)\tau^2, \end{aligned} \tag{5.407}$$

and the time step control is given as

$$\tau < \frac{1}{\frac{1}{12}||[\tilde{A}_1,[\tilde{A}_1,\tilde{A}_2]] + \frac{1}{12}||[\tilde{A}_2,[\tilde{A}_2,\tilde{A}]]||}. \tag{5.408}$$

In the following, we discuss the splitting ideas based on the separation of the eigenvectors.

### 5.6.6 Splitting Based on Separation of Eigenvectors (Assumption: Linearized Jacobian Matrix)

In the following, the idea is based on decomposing to fast and slow reaction equations, see the literature [101–103].

We separate into a fast subspace ($m, n = 1, \ldots, M$), and a slow subspace ($I, J = M + 1, \ldots, N$).

If we rewrite to a fast and slow equations separately, we have

$$\frac{du_m}{dt} = \sum_{n=1}^{M} a_{n,m}u_n + \sum_{J=M+1}^{N} a_{J,m}u_J, m = 1, 2, \ldots, M, \tag{5.409}$$

$$\frac{du_I}{dt} = \sum_{n=1}^{M} a_{n,I}u_n + \sum_{J=M+1}^{N} a_{J,I}u_J, I = M + 1, 2, \ldots, N, \tag{5.410}$$

or writing into the splitting matrices:

$$\frac{d\mathbf{u}}{dt} = A_1\mathbf{u} + A_2\mathbf{u}, \tag{5.411}$$

$\mathbf{u}=(\mathbf{u_F}, \mathbf{u_S})^t, A_1=\begin{pmatrix} A_{1,1} & A_{1,2} \\ 0 & 0 \end{pmatrix}, A_2 = \begin{pmatrix} 0 & 0 \\ A_{2,1} & A_{2,2} \end{pmatrix}$, where $\mathbf{u_F} = (u_1, \ldots, u_M)^t$ and $\mathbf{u_F} = (u_{M+1}, \ldots, u_N)^t$.

For the assumption of a constant Jacobian $A$, we have the following decomposition:

$$Am_i = \lambda_i m_i, \; i = 1, \ldots, N, \tag{5.412}$$

where we have the eigenvectors $m_i, \; i = 1, \ldots, N$ with the eigenmatrix $M = (m_1, \ldots, m_N)$ and eigenvalues $\lambda_i, \; i = 1, \ldots, N$.

We have the following decomposition criterion:

$$\lambda_j \geq \lambda_{bound}, j = 1, \ldots, N_{fast}, \tag{5.413}$$

$$\lambda_j < \lambda_{bound}, j = 1, \ldots, N_{slow}, \tag{5.414}$$

and for example $\lambda_{bound} = 10^2$ and we order into the fast and slow eigenmatrices: $\Lambda_{fast}, \Lambda_{slow}$.

Then we decouple into the splitting matrices:

$$A_1 = M\lambda_{fast}M^{-1}, \; A_2 = M\lambda_{slow}M^{-1}. \tag{5.415}$$

### 5.6.7 Splitting Based on Fast and Slow Dynamics Based on the Idea of the CSP (Computational Singular Perturbation) (Assumption: Linear Jacobian)

The idea is based on decomposing with respect to the dynamics (Jacobian matrix) of the underlying nonlinear problem.

Based on the orthogonalization of the eigenvalue problem, we decompose into a fast and slow dynamics.

In the following, the idea is based on decomposing to a orthogonal system of basis vectors $\{a_1, \ldots, a_n\}$, which are decomposed as a fast vector field $\{a_1, \ldots, a_m\}$ and slow vector field $\{a_{m+1,\ldots,a_n}\}$.

We deal with the $N$ dimensional stiff autonomous ODEs:

$$\frac{d\mathbf{y}}{dt} = g(\mathbf{u}), \tag{5.416}$$

further the Jacobian is given as

$$Dg = \frac{dg(\mathbf{y})}{d\mathbf{y}}. \tag{5.417}$$

We apply the decomposition via the eigenvalue problem:

$$\Lambda = B(Dg)A + \frac{dB}{dt}A, \tag{5.418}$$

where $\Lambda$ is a linear operator, if we assume the constant case, we have the eigenvalue problem:

$$\Lambda = B(Dg)A, \tag{5.419}$$

where $BA = I$ and $A = [a_1, \ldots, a_n]$ are the eigenspace with the eigenvalues.

We have to do an orthogonalization of the eigenspace where $b^j a_i = \delta_i^j$.

Further we assume a dissipative nature of the scheme means $\lambda_{m,r} < 0$ and $|\lambda_{m,r}| \gg |\lambda_{m,i}|$ (where $r$ is the real and $i$ is the imaginary part), we reorder the eigenvalues in the following case:

$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|. \tag{5.420}$$

We assume to that $\lambda_1, \ldots, \lambda_m$ is the fast regime and $\lambda_{m+1}, \ldots, \lambda_n$ is the slow regime and we have the underlying eigenmatrices $A_{fast} = [a_1, \ldots, a_m]$ and $A_{slow} = [a_{m+1}, \ldots, a_n]$.

Then we decouple the system into:

$$\frac{d\mathbf{u}}{dt} = g_{fast}(\mathbf{u}) + g_{slow}(\mathbf{u}), \tag{5.421}$$

$$= B^{fast} A_{fast} g(\mathbf{u}) + B^{slow} A_{slow} g(\mathbf{u}), \tag{5.422}$$

then we apply the splitting scheme:

$$\frac{d\mathbf{u}_1}{dt} = B^{fast} A_{fast} g(\mathbf{u}_1), \ \mathbf{u}_1(t^n) = \mathbf{u}(t^n), \tag{5.423}$$

$$\frac{d\mathbf{u}_2}{dt} = B^{slow} A_{slow} g(\mathbf{u}_2), \quad \mathbf{u}_1(t^n) = \mathbf{u}_1(t^{n+1}). \tag{5.424}$$

*Remark 5.40* The algorithm is based on the Gram–Schmidt orthogonalization, see [104]. The matrices are reordered into lower and higher eigenvalues, which are separated in different matrices, means a matrix with higher eigenvalues and a matrix with lower eigenvalues. At least, one can consider, what means lower or higher eigenvalue, e.g. $\lambda_{\min} \leq \lambda_{low} << \lambda_{high} \leq \lambda_{\max}$, where one define $\lambda_{\min}$ or $\lambda_{\max}$, and choose the separation into such reordered matrices. We could also skip the high oscillating matrix with the higher eigenvalues and consider only the slow scales of the dynamical system, see [102].

### 5.6.8 Strategies for the Decomposition

In the following, we discuss our strategies based on reducing the dimensionality of the system of chemical kinetics equations.

- Model reduction: skip fast scales (slow manifolds only apply $\mathcal{O}(1)$),
- Decomposition into a stiff and non-stiff part (perturbation with $\mathcal{O}(\varepsilon)$),
- CSP (Computational singular Perturbations), here we apply an algorithm to separate the fast and slow manifolds.

The strategies are discussed in the following subsections.

#### 5.6.8.1 Slow Manifolds (Perturbation of the Fast Scales)

The idea is based on the following problem, also called method of multiple timescales [105].

Some previous definitions:

**Definition 5.1** We assume $\rho A = \max\{|\lambda| : \lambda \in \sigma(A)\}$ is the spectral radius of $A$ (means the maximal eigenvalue), $\sigma(A)$ is the spectrum of $A$ means all eigenvalues of $A$.

We assume to have an operator $A$ being the operator for the slow manifolds and further an operator $B$ is the operator for fast manifolds.

$$\frac{\partial U(t)}{\partial t} = AU(t) + \tilde{B}U(t), \text{ with } U(t^n) = U^n, \tag{5.425}$$

where we assume
$\lambda_{\max,A} = \rho(A)$ and $\lambda_{\max,\tilde{B}} = \rho(\tilde{B}) = \frac{1}{\varepsilon}\lambda_{\max,A}$, so if we apply $\varepsilon\tilde{B} = B$ and we obtain $\lambda_{\max,B} = \rho(B) = \lambda_{\max,A}$.

So that means

$$\varepsilon = \frac{\lambda_{A,\max}}{\lambda_{\tilde{B},\max}}. \tag{5.426}$$

So we have the same timescale of the operator $A$ and $B$, means

$$\Delta t \leq \frac{1}{\lambda_{A,\max}}. \tag{5.427}$$

We derive a solutions $U(t, \varepsilon)$ and apply:

$$U(t, \varepsilon) = U_0(t) + \frac{1}{\varepsilon}U_1(t) + \frac{1}{\varepsilon^2}U_2(t) + \cdots + \frac{1}{\varepsilon^J}U_J(t), \tag{5.428}$$

with the initial conditions $U(0, \varepsilon) = U(0)$ and $J \in \mathbb{N}^+$ is a fixed iteration number.

Then the hierarchical equations are given as

$$\frac{\partial U_0(t)}{\partial t} = AU_0(t), \text{ with } \mathscr{O}(1), \tag{5.429}$$

$$\frac{\partial U_1(t)}{\partial t} = AU_1(t) + BU_0(t), \text{ with } \mathscr{O}\left(\frac{1}{\varepsilon}\right), \tag{5.430}$$

$$\vdots$$

$$\frac{\partial U_I(t)}{\partial t} = AU_J(t) + BU_{J-1}, \text{ with } \mathscr{O}\left(\frac{1}{\varepsilon^I}\right), \tag{5.431}$$

and we have also to expand the initial conditions to $U_0(0) = U(0)$ and $U_j(0) = 0, \ \forall j = 1, \ldots, J$.

*Remark 5.41* In our application, we compute the maximal eigenvalues of our operators (Jacobian), we separate a slow and fast operator based on the range of the eigenvalues, means

$$A = \Lambda_{|\lambda| \leq \lambda_{decomp}}, \tag{5.432}$$

and

$$\tilde{B} = \Lambda_{|\lambda| > \lambda_{decomp}}, \tag{5.433}$$

where for example $\lambda_{decomp} = 10^3$.
Then we apply $\varepsilon = \frac{\lambda_{decomp}}{\lambda_{\tilde{B},\max}}$ and start the algorithm.

#### 5.6.8.2 Simple Decomposition Algorithm of $\mathcal{O}(1)$

In the following, we deal with a separation of so-called stiff and non-stiff parts.

We decompose by the following criterion of the Jacobian matrix $J$. We assume that a Jacobian can be written in the following portions:

$$J = J_1 + J_2 = \begin{pmatrix} J_{1,11} + J_{2,11} & J_{1,12} + J_{2,11} \\ J_{1,21} + J_{2,11} & J_{1,22} + J_{2,11} \end{pmatrix}, \tag{5.434}$$

where each submatrix is given as $||J_{1,ij} \gg J_{1,ij}||$ with $i, j = 1, 2$. Means the matrix norm of the submatrices are for one part (the $J_1$ matrix) much more larger than for the second part (the $J_2$ matrix).

Therefore, we decompose into:

$$J_1 = \begin{pmatrix} J_{1,11} & J_{1,12} \\ J_{1,21} & J_{1,22} \end{pmatrix}, \tag{5.435}$$

and

$$J_2 = \begin{pmatrix} J_{2,11} & J_{2,12} \\ J_{2,21} & J_{2,22} \end{pmatrix}, \tag{5.436}$$

where we have assumed $J_1$ is the stiff and $J_2$ the non-stiff part, while we apply a much more smaller time step to resolve the stiff part as the non-stiff part, see separation into stiff- and non-stiff parts of differential equations in [106].

Further we have tested the modification into:

$$||J_{1,11}|| \gg \max\{(||J_{1,12}||, ||J_{1,21}||), ||J_{1,22}||\},$$
$$||J_{2,11}|| \gg \max\{(||J_{2,12}||, ||J_{2,21}||), ||J_{2,22}||\},$$

with $||J_{1,11}|| \gg ||J_{2,11}||$. Such that we could apply much more smaller time steps for the separated parts of the stiff equation $J_{1,11}$ and larger time steps for the non-stiff parts. Therefore, we could accelerate the computation and reduce larger matrices into much more smaller matrices, which are faster to solve.

#### 5.6.8.3 CSP (Computational Singular Perturbation) Method

The idea is based on a decomposition to a fast and slow vector field, means

$$\frac{d\mathbf{y}}{dt} = g(\mathbf{y}), \tag{5.437}$$

is decoupled into $g(\mathbf{y}) = g_{fast}(\mathbf{y}) + g_{slow}(\mathbf{y})$

$$g = Af, \tag{5.438}$$
$$f = Bg, \tag{5.439}$$

where we have

$$\frac{df}{dt} = \Lambda f, \tag{5.440}$$

where $\Lambda = B(Dg)A + \frac{dB}{dt}A = B[A, g]$ and $[\cdot, \cdot]$ is the Lie-bracket ($[a, g] = (Dg)a - (Da)g$).

Such a linear transformation gives the dynamics of the vector $f$.

The algorithm is based on a Gram–Schmidt orthogonalization, see the idea in [107], and we obtain the following reduced equation system:

$$\frac{df}{dt} = \Lambda_{new} f, \tag{5.441}$$

where $\Lambda_{new} = \begin{pmatrix} B^{s\perp}[A_f, g] & 0 \\ 0 & B^{f\perp}[A_s, g] \end{pmatrix}$, and $B^{s\perp}$ is a basis of the slow manifold, while $B^{f\perp}$ is a basis of the fast manifold.

### 5.6.9 Numerical Examples

In the following, we deal with the different numerical examples and the norms of the errors of the additive and multiplicative splitting approach

- Additive Splitting:

$$err_{Multipl} = ||\sum_{i=1}^{m} \sum_{j=1, j\neq i}^{m} (\tilde{A}_i' \tilde{A}_j + \tilde{A}_j' \tilde{A}_i)||, \tag{5.442}$$

- Multiplicative Splitting

$$err_{Multipl} = \frac{1}{12}||[\tilde{A_1}', [\tilde{A}_1', \tilde{A}_2]] + \frac{1}{12}||[\tilde{A_2}', [\tilde{A}_2', \tilde{A}]]||, \tag{5.443}$$

where $\tilde{A}_i'$ are the derivation, means Jacobian of the operator $\tilde{A}_i$. For the norms of the matrices, we apply the maximum- and $L_2$-norm.

We apply the following decomposition techniques:

- Decomposition without eigenvalue consideration (we apply all eigenvalues).
- Decomposition with eigenvalue consideration (we skip the highest eigenvalues).

The decomposition techniques are presented with the following numerical computational results and we consider the following eigenvalues:

1. Decomposition without eigenvalue consideration
   In the following, we only decompose to an upper and lower matrices
   We have the following time step.
   The parallel splitting has a time step of

$$\tau < \frac{1}{||J_1J_2 + J_2J_1)||} = 7.9452\ 10^{-12},$$

   The commutator $||(J_1J_2 - J_2J_1)|| = 3.76685 * 10^{10}$,

   and the anti-commutator $||(J_1J_2 + J_2J_1)|| = 7.52185 * 10^{10}$.

2. Decomposition with eigenvalue consideration
   In the following, we only decompose to an upper and lower matrices.
   We have the following time step.

$$||J_1J_2 + J_2J_1||_{\max} = \max_{1\leq j\leq n} \sum_{i=i}^{m} |a_{i,j}| = 5.85252 \times 10^6, \tag{5.444}$$

$$||J_1J_2 - J_2J_1||_{\max} = \max_{1\leq j\leq n} \sum_{i=i}^{m} |a_{i,j}| = 5.85252 \times 10^6, \tag{5.445}$$

   and we apply the time step $\tau = 1.26438 \times 10^{-7}$.

*Remark 5.42* For the numerical computations, it makes sense to embed the informations about the eigenvalues of the dynamical system. We obtain a reduction of the time steps, if we reordered the matrices and skip to high oscillating eigenvalues. The time step enlarged by a factor of $10^5$. Means we could accelerate the computation of the dynamical system, without reducing too much information about the dynamics in the slower regions.

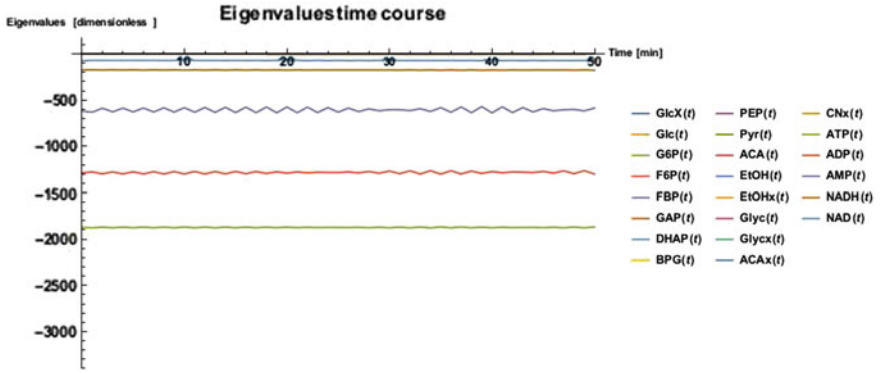



**Fig. 5.34** Eigenvalues of the Jacobian's of the differential equations

The graphical presentation of the eigenvalues of the Jacobian's of the dynamical system is shown in Fig. 5.34.

*Remark 5.43* The detailed explanation of the separation by the different eigenvalues and the separation into a lower and upper part of a Jacobian matrix is discussed in the Appendix B.3.

The detailed timecourse of concentrations, which are applied with the linearized dynamical system (5.455) based on the Jacobian matrix, are shown in Figs. 5.35 and 5.36.

*Remark 5.44* The variance of the concentrations for the different species allows a separation into different submatrices to reduce the amount of computations. We could apply the different decomposition strategy to split into different simpler and monoscaled problems. We also see a possibility to smoothen the perturbations of the highly oscillating concentration by an averaging over a longer distance.

### *5.6.10 Conclusion*

We have discussed a delicate dynamical system based on a full model of glycolysis in Saccharomyces cerevisiae. While the problem is related to highly oscillating parts of the equation system, we present decomposition ideas to separate the slow and fast oscillating parts of the differential equations. We concentrate on the eigenvalues of the underlying Jacobian matrix, which can be seen as an indicator to the dynamical behaviour of the systems. Different splitting ideas are presented and we could discuss the benefits and drawbacks of such schemes. Based on the study of the individual oscillatory behaviour of the concentrations, we could apply model reduction ideas to

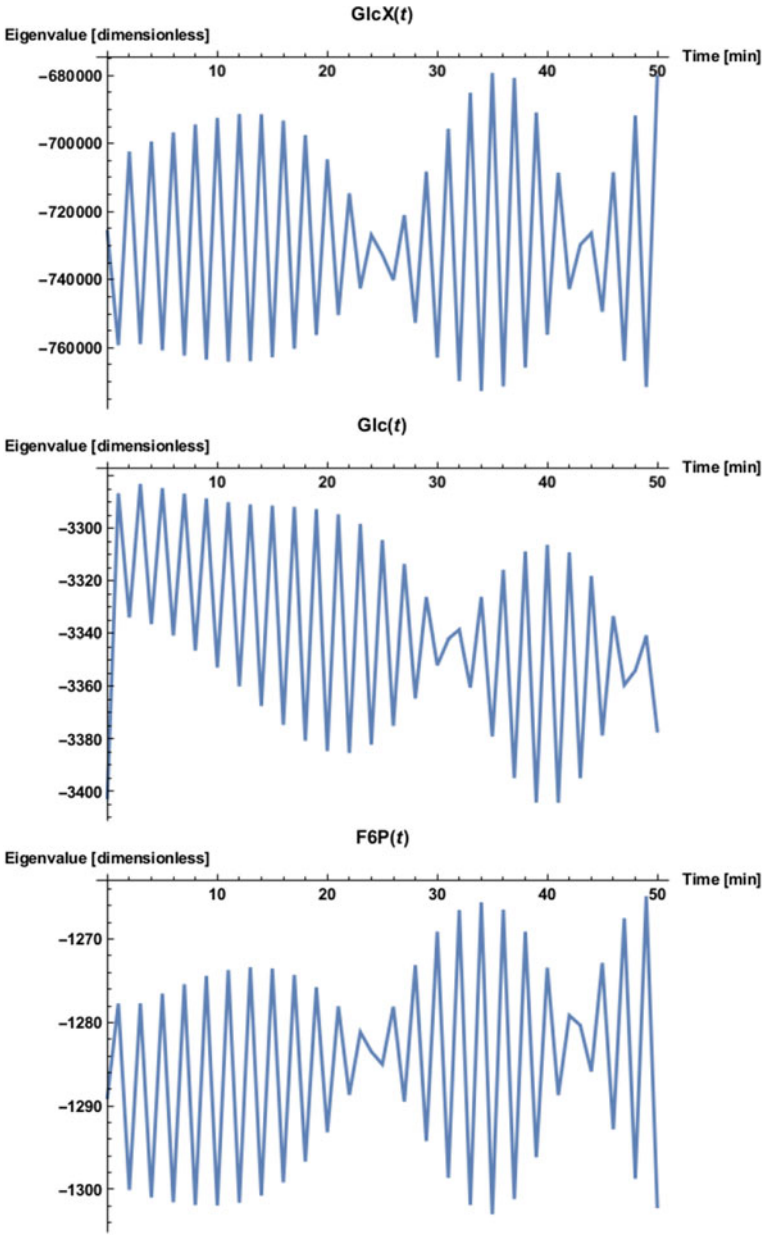



**Fig. 5.35** Timecourse of the different eigenvalues of the Jacobian matrix (multiple timescales: $10^{5}$–$10^{-29}$)

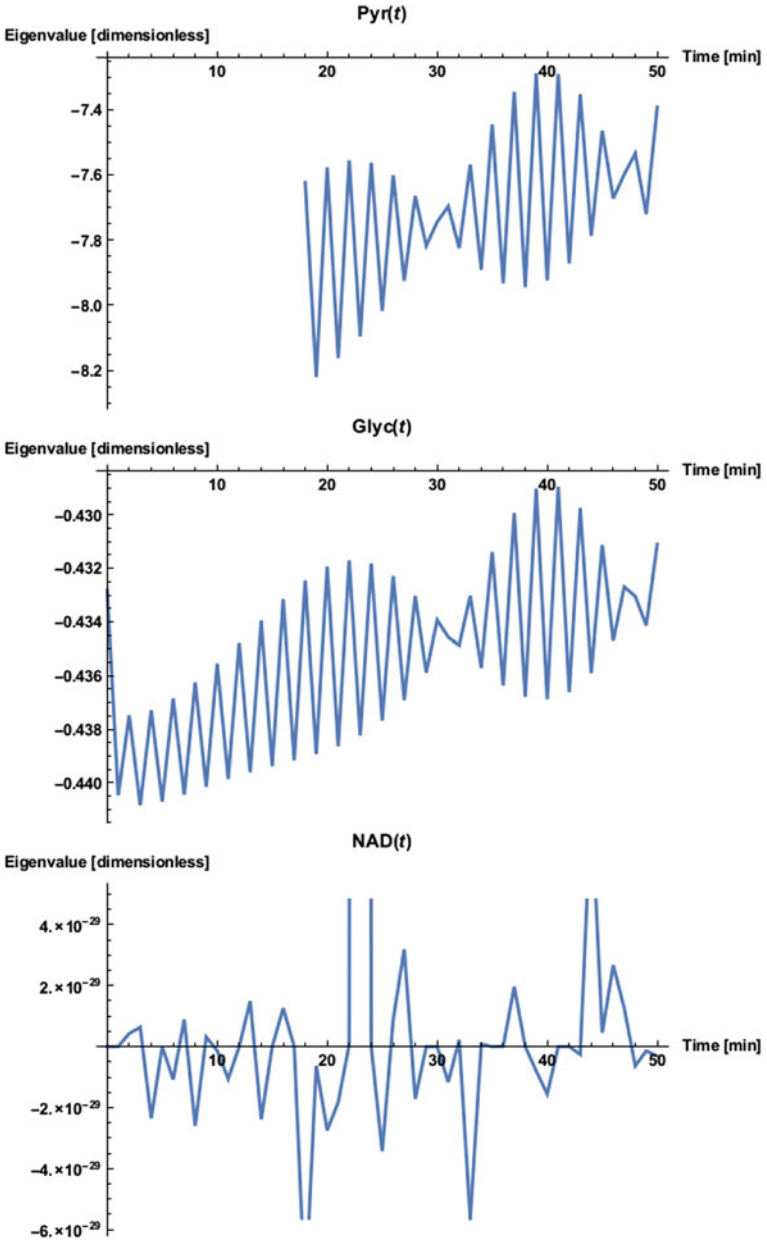



**Fig. 5.36** Timecourse of the different eigenvalues of the Jacobian matrix (multiple timescales: $10^{5}$–$10^{-29}$)

such systems. The understanding of the dynamical behaviour based on the Jacobian is therefore important for a model reduction of such delicate dynamical systems, which are multicomponent models.

## 5.7 Splitting Approach for a Plasma Resonance Spectroscopy

**Abstract** The model is motivated to a real-world application in a plasma diagnostic method. The idea is to absorb a probe into a plasma and apply a radio frequency signal to the probe. The diagnostic is based on the resonance near the plasma frequency. The model is formulated based on the plasma/probe idea and the underlying equation is based on a Boltzmann's equation coupled by a Poisson's equation. One of the main problems are an application to a general geometry, such that the model can be applied to a realistic apparatus. Here, one obtain a multiple operator equation with different scale behaviours. We discuss a splitting approach for solving a delicate model in active plasma resonance spectroscopy.

### *5.7.1 Introduction*

Since recent years, active plasma resonance spectroscopy is a widely used plasma diagnostic method. The motivation arose to the idea that a plasma resonates near the plasma frequency, see Tonks and Langmuir [108]. We are motivated to simulate an active plasma resonance spectroscopy, which is a well established plasma diagnostic method. The underlying idea is to immerse a probe into a plasma and applying an rf signal to the probe tip. Such a response allows to see the natural ability of plasmas, which is the resonance on or near the plasma frequency.

The model equation of such a process is a Boltzmann equation, we deal with a linearization and obtain a linear kinetic equation, see [109], which can be applied with linear splitting methods.

We split the original discretized operator into a sum of two operators, one of which corresponds with a transport equation, which can be solved fast with characteristics methods, the other one is the collision operator, which can be solved with fast integral solvers, see [110].

We propose a new sequential and iterative splitting method, that taken into account the different underlying spatial discretized operators. While the non-stiff transport operator is solved with standard characteristics methods, the stiff linearized collision operator is solved with implicit ODE solvers.

After a number of approximations we consider the error of the method and proposed a choice of the operators.

### 5.7.2 Modelling

We deal with the following model, which is a linearized and normalized kinetic equation, based on the Boltzmann equation [109], and is given as

$$\frac{\partial g}{\partial t} + \boldsymbol{v} \cdot \nabla (g - \phi\{g\}) + \nabla \Phi \cdot \nabla_v g = \frac{\nu}{4\pi} \int_\Omega g(|\boldsymbol{v}|\boldsymbol{e})\, d\Omega - \nu g + \boldsymbol{v} \cdot \nabla \phi^{vac}. \tag{5.446}$$

The linearized Boltzmann equation is coupled to Poisson's equation with homogeneous boundary conditions

$$\nabla\,(\varepsilon \nabla \phi) = \int_{\mathbf{R}^3} wg\; d^3v. \tag{5.447}$$

It is a challenging task to solve the Boltzmann–Poisson system in a general geometry. Therefore, we focus on a planar geometry in $z$-direction with $z \in [0, L]$. In Cartesian coordinates Poisson' equation can be solved and introduced as integral operator in (5.446). In velocity space a transformed spherical coordinate system is applied caused by the isotropic collision operator.

We have the transformed Boltzmann equation related to the realistic coordinates of the real-world problem:

$$\begin{aligned}
&\frac{\partial g}{\partial t} + \frac{\partial \Phi}{\partial z} \left( \frac{(1-\xi^2)}{\sqrt{2\varpi}} \frac{\partial g}{\partial \xi} + \xi \sqrt{2\varpi} \frac{\partial g}{\partial \varpi} \right) + \xi \sqrt{2\varpi} \frac{\partial g}{\partial z} \\
&\quad - \frac{\xi \sqrt{2\varpi}}{\sqrt{\pi}} \int_0^z \int_{-1}^1 \int_0^\infty \sqrt{\varpi} e^{-\varpi + \Phi(z')} g\, d\varpi\, d\xi\, dz' \\
&\quad + \frac{\xi \sqrt{2\varpi}}{\sqrt{\pi} L} \int_0^L \int_0^{z'} \int_{-1}^1 \int_0^\infty \sqrt{\varpi} e^{-\varpi + \Phi(z'')} g\, d\varpi\, d\xi\, dz''\, dz' \\
&= \xi \sqrt{2\varpi} \frac{\partial \phi^{vac}}{\partial z} + \nu \left( \frac{1}{2} \int_{-1}^1 g(z, \varpi, \xi)\, d\xi - g \right).
\end{aligned} \tag{5.448}$$

We decompose the transformed Boltzmann equation into the following operator equation:

$$\frac{\partial g}{\partial t} = Ag + Bg + Cg + f, \tag{5.449}$$

where the operators are defined as

$$A = -\frac{\partial \Phi}{\partial z} \left( \frac{(1-\xi^2)}{\sqrt{2\varpi}} \frac{\partial}{\partial \xi} + \xi \sqrt{2\varpi} \frac{\partial}{\partial \varpi} \right), \tag{5.450}$$

$$B(g) = \frac{\xi \sqrt{2\varpi}}{\sqrt{\pi}} \left( \int_0^z \int_{-1}^1 \int_0^\infty \sqrt{\varpi} e^{-\varpi + \Phi(z')} g\, d\varpi\, d\xi\, dz' \right.$$

$$-\frac{1}{L}\int_0^L\int_0^{z'}\int_{-1}^{1}\int_0^{\infty}\sqrt{\varpi}e^{-\varpi+\Phi(z'')}g\,d\varpi\,d\xi\,dz''\,dz' - \sqrt{\pi}\frac{\partial g}{\partial z}\Big), \quad (5.451)$$

$$C(g) = \nu\left(\frac{1}{2}\int_{-1}^{1} g(z,\varpi,\xi)\,d\xi - g\right), \quad (5.452)$$

$$f = \xi\sqrt{2\varpi}\frac{\partial\phi^{vac}}{\partial z}. \quad (5.453)$$

We obtain one linear and two nonlinear operators, while we have different timescales based on the linear and nonlinear operators.

Such a coupling of multi-operators with linear and nonlinear parts can be solved with splitting methods, which are at least also multiscale methods to separate between different spatial and timescales.

### *5.7.3 Splitting Schemes*

The operator splitting methods are used to solve complex models in the geophysical and environmental physics, they are developed and applied in [14, 29, 111].

The idea is based on solving simpler equations with respect to receive higher order discretization methods for the remain equations.

For our coupled linearized Boltzmann and Poisson's equations, we have derived an embedded approach of the Poisson's equation to the boundary integral, see Eq. (5.449).

Therefore, we deal with the following splitting approach for the transformed Boltzmann Eq. (5.449):

- Sequential splitting methods,
- Iterative splitting approach (fixpoint schemes),

based on the operators $A$, $B$, $C$.

#### 5.7.3.1 Sequential Splitting: First Method for Linearized Equations

First we describe the simplest operator splitting , which is called sequential operator splitting for the following system of ordinary linear differential equations:

$$\partial_t c(t) = A\,c(t) \;+\; B\,c(t) + Cc(t), \quad (5.454)$$

whereby the initial conditions are $c^n = c(t^n)$. The operators $A$ and $B$ are spatially discretized operators, e.g. they correspond to the discretized in space convection and diffusion operators (matrices). Hence, they can be considered as bounded operators.

The sequential operator splitting method is introduced as a method which solve the two sub-problems sequentially, see [112], where the different sub-problems are connected via the initial conditions. This means that we replace the original problem (5.454) with the sub-problems

$$\begin{aligned}
&\frac{\partial c^*(t)}{\partial t} = Ac^*(t) + f_1(t), \quad \text{with } c^*(t^n) = c^n,\\
&\frac{\partial c^{**}(t)}{\partial t} = Bc^{**}(t) + f_2(t), \quad \text{with } c^{**}(t^n) = c^*(t^{n+1}),\\
&\frac{\partial c^{***}(t)}{\partial t} = Cc^{***}(t) + f_3(t)\ , \quad \text{with } c^{***}(t^n) = c^{**}(t^{n+1}),
\end{aligned} \tag{5.455}$$

whereby the splitting time step is defined as $\tau_n = t^{n+1} - t^n$. The approximated split solution is defined as $c^{n+1} = c^{***}(t^{n+1})$ and the inhomogeneous parts are given as $c_1 + c_2 + c_3 = 1, f_1(t) = c_1 f(t), f_2(t) = c_2 f(t), f_3(t) = c_3 f(t)$.

Clearly, the change of the original problems with the sub-problems usually results some error, called *splitting error*. Obviously, the splitting error of the operator splitting method can be derived as follows (cf. e.g. [113]):

$$\begin{aligned}
\rho_n &= ||\frac{1}{\tau}(\exp(\tau_n(A+B+C)) - \exp(\tau_n C)\exp(\tau_n B)\exp(\tau_n A))\ c(t^n)||\\
&\quad + ||\mathscr{F}(t_n+\tau_n, t_n) - S_2(\tau_n)\mathscr{F}_1(t_n+\tau_n, t_n) - \mathscr{F}_2(t_n+\tau_n, t_n)||\\
&= \frac{1}{2}\tau_n([A,B] + [A,C] + [B,C])\ c(t^n) + O(\tau^2).
\end{aligned} \tag{5.456}$$

whereby for example $[A, B] := AB - BA$ is the commutator of $A$ and $B$. Consequently, the splitting error is $O(\tau_n)$ when the operators $A, B$ and $C$ do not commute, otherwise the method is exact. Hence, by definition, the operator splitting method is called *first order splitting method*.

The inhomogeneous part can be estimated as

$$\begin{aligned}
&||\mathscr{F}(t_n+\tau_n, t_n) - S_2(\tau_n)\mathscr{F}_1(t_n+\tau_n, t_n) - \mathscr{F}_2(t_n+\tau_n, t_n)||\\
&\quad = \tau_n^2 ||A+B+C||\left(C(T)||f_1(t)|| + C'(T)||f_2(t)||\right) + O(\tau^3),
\end{aligned} \tag{5.457}$$

where

$$\begin{aligned}
\mathscr{F}(t_n+\tau_n, t_n) &= \int_{t_n}^{t} \exp((A+B)(t-s))f(s)ds,\\
\mathscr{F}_1(t_n+\tau_n, t_n) &= \int_{t_n}^{t} \exp(A(t-s))f_1(s)ds,\\
\mathscr{F}_2(t_n+\tau_n, t_n) &= \int_{t_n}^{t} \exp(B(t-s))f_2(s)ds,\\
S_2(\tau_n) &= \exp(B\tau_n),
\end{aligned}$$

and $t = t_n + \tau_n$.

*Remark 5.45* The optimal choice of the operators are given with the maximal eigenvalues or norm of the operators. We start with the operator of the largest eigenvalue and end with such operators with the lowest eigenvalue. Means the highest influence to the solutions is done in previous.

In the next subsection we present the iterative splitting method.

### 5.7.3.2 Iterative Splitting Method

The following algorithm is based on the iteration with fixed splitting discretization step-size $\tau$, namely, on the time interval $[t^n, t^{n+1}]$ we solve the following subproblems consecutively for $i = 0, 3, \ldots 3m$, (confer [14, 34]).

$$\frac{\partial c_i(t)}{\partial t} = Ac_i(t) + Bc_{i-1}(t) + Cc_{i-1}(t) + f, \text{ with } c_i(t^n) = c^n$$
$$\text{and } c_0(t^n) = c^n, \; c_{-1} = 0.0, \tag{5.458}$$
$$\frac{\partial c_{i+1}(t)}{\partial t} = Ac_i(t) + Bc_{i+1}(t) + Cc_i(t) + f,$$
$$\text{with } c_{i+1}(t^n) = c^n, \tag{5.459}$$
$$\frac{\partial c_{i+2}(t)}{\partial t} = Ac_{i+1}(t) + Bc_{i+1}(t) + Cc_{i+2}(t) + f,$$
$$\text{with } c_{i+2}(t^n) = c^n, \tag{5.460}$$

where $c^n$ is the known split approximation at the time level $t = t^n$. The split approximation at the time level $t = t^{n+1}$ is defined as $c^{n+1} = c_{3m+1}(t^{n+1})$. (Clearly, the function $c_{i+1}(t)$ depends on the interval $[t^n, t^{n+1}]$, too, but, for the sake of simplicity, in our notation we omit the dependence on $n$.)

In the following we will analyse the convergence and the rate of the convergence of the method (5.458) and (5.459) for $m$ tends to infinity for the linear operators $A, B, C : \mathbf{X} \to \mathbf{X}$ where we assume that these operators and their sum are generators of the $C_0$ semigroups. We emphasize that these operators are not necessarily bounded, so, the convergence is examined in general Banach space setting.

**Theorem 5.17** *Let us consider the abstract Cauchy problem in a Banach space* $\mathbf{X}$

$$\begin{gathered} \partial_t c(t) = Ac(t) + Bc(t) + Cc(t) + f, \quad 0 < t \le T \\ c(0) = c_0 \end{gathered} \tag{5.461}$$

*where* $A, B, C, A + B, A + C, B + C : \mathbf{X} \to \mathbf{X}$ *are given linear operators being generators of the* $C_0$*-semigroup and* $c_0 \in \mathbf{X}$ *is a given element. Then the iteration process (5.458)–(5.460) is convergent and for* $i = 0, 3, 6, \ldots (m-1)3$ *we have the order of* $3m$.

The proof based on Waveform-relaxation schemes is given in [114], in the following, we apply the proof idea in [115].

*Proof* Let us consider the iteration (5.458)–(5.460) on the sub-interval $[t^n, t^{n+1}]$. For the local error function $e_i(t) = c(t) - c_i(t)$ we have the relations:

$$\begin{aligned} \partial_t e_i(t) &= Ae_i(t) + (B + C)e_{i-1}(t), \quad t \in (t^n, t^{n+1}], \\ e_i(t^n) &= 0, \end{aligned} \tag{5.462}$$

and

$$\begin{aligned} \partial_t e_{i+1}(t) &= (A + C)e_i(t) + Be_{i+1}(t), \quad t \in (t^n, t^{n+1}], \\ e_{i+1}(t^n) &= 0, \end{aligned} \tag{5.463}$$

and

$$\begin{aligned} \partial_t e_{i+2}(t) &= (A + B)e_{i+1}(t) + Ce_{i+2}(t), \quad t \in (t^n, t^{n+1}], \\ e_{i+2}(t^n) &= 0, \end{aligned} \tag{5.464}$$

for $m = 0, 3, 6, \ldots$, with $e_0(0) = 0$ and $e_{-1}(t) = c(t)$.

The errors are given as

$$\begin{aligned} &||e_i(t)|| \leq ||A_2 + A_3|| \int_{t^n}^{t} ||\exp(||A_1||(t-s))\, ds\, ||e_{i-1}(t)||, \quad t \in [t^n, t^{n+1}]. \\ &||e_i(t)|| \leq ||A_2 + A_3|| \frac{\exp(t||A_1||)}{||A_1||} ||e_{i-1}(t)||, \\ &||e_i(t)|| \leq ||A_2 + A_3|| M_1(t) ||e_{i-1}(t)||, \end{aligned} \tag{5.465}$$

where we assume $M_1(t)$ is of order $\mathcal{O}(t)$.

The same are done with the other errors

$$\begin{aligned} ||e_{i+1}(t)|| &\leq ||A_1 + A_3|| M_2(t) ||e_i(t)||, \\ ||e_{i+2}(t)|| &\leq ||A_1 + A_2|| M_3(t) ||e_{i+1}(t)||, \end{aligned} \tag{5.466}$$

where we assume $M_2(t), M_3(t)$ is of order $\mathcal{O}(t)$.

We obtain:

$$||e_{i+2}(t)|| \leq \Pi_{l=1}^{3} ||A - A_l|| M_l(t) ||e_{i-1}(t)||, \tag{5.467}$$

and recursively to $e_0$ we have

$$||e_{3(m-1)}(t)|| \leq (\Pi_{l=1}^{3} ||A - A_l|| M_l(t))^m ||e_0(t)||. \tag{5.468}$$

We obtain the given order of the scheme, see also [115].

*Remark 5.46* The optimal choice of the operators are given with the maximal eigenvalues or norm of the operators. We iterate over the operator of the largest eigenvalue, while the operators with the lower eigenvalue are only used as perturbation operators. Means the highest influence to the solutions is done in each iterative step.

### 5.7.4 Ideas of Numerical Examples of the Splitting Approaches

We discuss a simplified the kinetic equation based on the approach of a constant velocity field, given as

$$\frac{\partial g}{\partial t} = Ag + Bg + Cg, \tag{5.469}$$

where $g_0$ is the initial value of the equation.

Where we deal with three operators, which can be solved by Laplacian transformation and we obtain solver operators in matrix form for each operator means

$$g(t) = S_A(t)g_{A,0}, \tag{5.470}$$

where $S_A(t) = InvInverseLaplaceTransform[InvOpAz, t]$, solves the first equation with operator $A$,

$$g(t) = S_A(t)g_{B,0}, \tag{5.471}$$

where $S_B(t) = InvInverseLaplaceTransform[InvOpBz, t]$, solves the second equation with operator $B$,

$$g(t) = S_C(t)g_{C,0}, \tag{5.472}$$

where $S_C(t) = InvInverseLaplaceTransform[InvOpCz, t]$, solves the third equation with operator $C$.

The A–B splitting is coupled as

$$g(t) = S_{AB}(t)g_0 = S_C(t)S_B(t)S_A(t)g_0, \tag{5.473}$$

where $g_0$ is the initial value of the full equation and $t$ is the time step.

The Strang Splitting is given as

$$g(t) = S_{Strang}(t)g_0 = S_A(t/2)S_B(t/2)S_C(t)S_B(t/2)S_A(t/2)g_0, \tag{5.474}$$

where $g_0$ is the initial value of the full equation and $t$ is the time step.

For a sequence of computations, we separate the large time interval $[0, T]$ into smaller time intervals $\Delta t$.

We start with $t = 0$ and $c(0) = c_0$, we have $N$-timesteps, means $T/N = \Delta t$, with time points $t_0 = 0, t_1 = \Delta t, \ldots, t_N = N\Delta t$

$$g(t_{i+1}) = S_{Splitting}(\Delta t)g(t_i),\ i = 0, 1, 2, \ldots, N, \tag{5.475}$$

where $t_0 = 0$ and $g(0) = g_0$, further $Splitting = \{AB, Strang\}$.

*Remark 5.47* The numerical implementation was done in MATHEMATIKA, see [116], while we apply the Laplace transformation with respect to the spatial variables. The solution operator was derived by the semi-analytical solution of each operator equation with the inverse Laplace transformation. Such a decomposition is a multiscale method and allows an individual treatment of each operator part and an optimal resolution of the spatial scales, while the time scales are solved analytically by the exp-functions.

### *5.7.5 Conclusions and Discussions*

We present sequential and iterative operator splitting method to solve a linearized Boltzmann equation. The splitting approach is based on decomposing the delicate linear Boltzmann equation, while each splitted part can be solved with semi-analytical or analytical method. Based on the real-life application of the resonance spectroscopy, the transformation to a realistic, here spherical coordinate system was important. Both non-iterative and iterative splitting approaches are presented and an engineering toolbox based on MATHEMATIKA to solve each equation part independently.

## 5.8 Multiscale Approach with Adaptive and Equation-Free Methods for Transport Problems with Electric Fields

**Abstract** The model problem is related to a transport problem of particles in a electrical field. Based on the different scales of each model equation, means the macroscopic equation of the particles and the underlying microscopic equation of the electric field, we deal with a multiscale problem. We discuss a multiscale apporach based on the equation-free method to overcome the limitation of the time step of the microscopic scales. We apply a implicit time-discretization of the macroscopic equation and embed the microscopic equation via EFM into the large scales.

### *5.8.1 Introduction*

We are motivated to accelerate plasma models, which are related to transport and electric field equations, see the modelling idea in [117]. While, we have different scales belonging to the transport model of the particles and their underlying electromagnetic field equation, we have to deal with a multiscale model. Here the data transfer between the different scales are important to circumvent simulations with respect to the finest time and spatial scales.

In the following, we discuss the multiscale mode of macroscopic transport equations and microscopic electromagnetic field equations. Here, we apply the word microscopic and macroscopic, with respect to the different scales, finer scale equation is named as microscopic equation and the coarser scale equation is named as macroscopic equation.

In the following, we discuss the model equations:

- Unscaled model equations:

$$\frac{\partial}{\partial t} n_{\mathrm{i}}(x,t) + \frac{\partial}{\partial x}\left(\Gamma_{\mathrm{i}}(x,t)\right) = \alpha\left(E(x,t)\right)\left|\Gamma_{\mathrm{e}}(x,t)\right|, \tag{5.476}$$

$$\varepsilon_0 \frac{\partial}{\partial t} E(x,t) = \frac{1}{A_E} I_D(t) - e\left(\Gamma_{\mathrm{i}} - \Gamma_{\mathrm{e}}\right), \tag{5.477}$$

further we have the following additional constraints:

$$\frac{\mathrm{d}}{\mathrm{d}t} U_{\mathrm{C}_1}(t) = \frac{1}{C_1}\left(I_D + q_{\mathrm{e}} A_E\left(\Gamma_i\Big|_{x=0} - \Gamma_e\Big|_{x=0}\right)\right), \tag{5.478}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} U_{\mathrm{C}_2}(t) = \frac{1}{C_2}\left(I_D + q_{\mathrm{e}} A_E\left(\Gamma_i\Big|_{x=L} - \Gamma_e\Big|_{x=L}\right)\right). \tag{5.479}$$

The additional assumptions are given as

$$I_D = \frac{1}{R_{\mathrm{S}}}\left(V_{\mathrm{S}} - \int_0^L E(x,t)\,\mathrm{d}x - V_{\mathrm{C1}} - V_{\mathrm{C2}}\right), \tag{5.480}$$

$$\frac{\partial E}{\partial x} = \frac{q_{\mathrm{e}}}{\varepsilon_0}\Big(n_{\mathrm{i}} - n_{\mathrm{e}}\Big), \tag{5.481}$$

and the parameters of the equations are given as

$$\alpha\left(E(x,t)\right) = Ap\,\mathrm{e}^{-\frac{Bp}{|E(x,t)|}} \tag{5.482}$$

$$\Gamma_{\mathrm{i}}(x,t) = n_{\mathrm{i}}(x,t)\,\mu_{\mathrm{i}} E(x,t) \tag{5.483}$$

$$\Gamma_{\mathrm{e}}(x,t) = n_{\mathrm{e}}(x,t)\,\mu_{\mathrm{e}} E(x,t) \tag{5.484}$$

with the TOWNSEND-coefficents $A$ and $B$, see [118], and the neutral gas pressure $p$.

- Scaled model equations: For an engineering application, we retransfer to a partial differential equation system. We deal with the following scaled constants:

$$t_0 = \frac{1}{\alpha_0 \mu_{\mathrm{e}} E_0},\ q_0 = \varepsilon_0 \alpha_0 E_0,\ x_0 = \frac{1}{\alpha_0}, \tag{5.485}$$

$$j_0 = q_0 \mu_{\mathrm{e}} E_0,\ R_0 = \frac{E_0 \alpha_0}{j_0},\ C_0 = \frac{\varepsilon_0}{\alpha_0}, \tag{5.486}$$

$$U_0 = j_0 R_0 A_0,\ A_0 = \frac{1}{\alpha_0^2}. \tag{5.487}$$

The equation system (5.476) and (5.477) is reduced to the following equation system:

$$\frac{\partial n_\mathrm{i}}{\partial \tau} = -\frac{\partial}{\partial x}\left(\mu E n_\mathrm{i}\right) + |E n_\mathrm{e}|\, \mathrm{e}^{-\frac{1}{|E|}}, \tag{5.488}$$

$$\frac{\partial E}{\partial \tau} = j - \left(\mu n_\mathrm{i} - n_\mathrm{e}\right) E, \tag{5.489}$$

the additional constraint equations are given as

$$\frac{\mathrm{d}U_{\mathrm{C}_1}}{\mathrm{d}\tau} = \frac{A_\mathrm{E}}{C_1}\left(j + \left(\Gamma_i\Big|_{x=0} - \Gamma_e\Big|_{x=0}\right)\right), \tag{5.490}$$

$$\frac{\mathrm{d}U_{\mathrm{C}_2}}{\mathrm{d}\tau} = \frac{A_\mathrm{E}}{C_2}\left(j + \left(\Gamma_i\Big|_{x=L} - \Gamma_e\Big|_{x=L}\right)\right), \tag{5.491}$$

$$n_\mathrm{e} = n_\mathrm{i} - \frac{\partial E}{\partial x}, \tag{5.492}$$

$$V_\mathrm{D} = -\int_0^L E(x,t)\,\mathrm{d}x. \tag{5.493}$$

*Remark 5.48* Based on the transport equation of the particles (5.488), the electric field $E$ given in Eq. (5.489) is related as a convection parameter and therefore has a strong influence to the transport equation. Here, we have to resolve the electric field in order to achieve a correct convection operator. For such a relation the sensitive variable is given as $E$ and fine scale differences are important, while on the other hand the variable $n_i$ is more sensitive for coarser scales and we have to take into account such a relation of a multiscale dependency.

### 5.8.2 Numerical Methods

In the following, we discuss the numerical methods that are applied as multiscale schemes to coupled the microscopic scale of the electromagnetic field and the macroscopic scales of the transport field.

Here, we apply explicit or implicit integrators for the time variable and upwind finite difference methods for the space variable on a staggered grid. The benefit of the EFM is given with respect to extrapolate between the fine and coarse spatial- and timescale to accelerate the computations.

In the following we present the different schemes:

- Full explicit scheme (unsplitted version, we apply only one timescale $\Delta t$),
- Adaptive explicit scheme: we split the different equations into a slow scale (density equation) and fast scale (E-field equation),
- EFM explicit scheme: we extrapolate the fast scale to the slow scale and apply explicit time-integrators.

### 5.8.3 Full Explicit Scheme: With One Timescale $\Delta t$

The macroscopic equation is given as

$$\frac{{n_{\mathrm{i}}}_j^{n+1}}{\Delta t} = -\frac{{\Gamma_{\mathrm{i}}}_{j+\frac{1}{2}}^{n} - {\Gamma_{\mathrm{i}}}_{j-\frac{1}{2}}^{n}}{\Delta x} + {S_{\mathrm{e}}}_j^{n}, \tag{5.494}$$

where the CFL condition is given as $CFL : \Delta t \leq \frac{\Delta x}{\max(E_0^n, \ldots, E_{J+1}^n)}$.

For the one-scale method, the equations are discretized as

$$\frac{E_{j+\frac{1}{2}}^{n+1}}{\Delta t} = {j_{\mathrm{D}}}^{n} + {\Gamma_{\mathrm{i}}}_{j+\frac{1}{2}}^{n} + {\Gamma_{\mathrm{e}}}_{j+\frac{1}{2}}^{n}, \tag{5.495}$$

$$\frac{{U_{\mathrm{C}_1}}^{n+1}}{\Delta t} = \frac{A_{\mathrm{E}}}{C_1}\left({j_{\mathrm{D}}}^{n} + {\Gamma_{\mathrm{i}}}_0^n + {\Gamma_{\mathrm{e}}}_0^n\right), \tag{5.496}$$

$$\frac{{U_{\mathrm{C}_2}}^{n+1}}{\Delta t} = \frac{A_{\mathrm{E}}}{C_2}\left({j_{\mathrm{D}}}^{n} + {\Gamma_{\mathrm{i}}}_{J+1}^n + {\Gamma_{\mathrm{e}}}_{J+1}^n\right), \tag{5.497}$$

where we have a time step $\Delta t$-limit, which is given by the CFL condition.

Further the electron density $n_{\mathrm{e}}$, source term $S_{\mathrm{e}}$ and discharge current density $j_{\mathrm{D}}$ is given by the discrete approach

$${n_{\mathrm{e}}}_j^{n} = {n_{\mathrm{i}}}_j^{n} - \frac{E_{j+\frac{1}{2}}^{n} - E_{j-\frac{1}{2}}^{n}}{\Delta x}, \tag{5.498}$$

$${S_{\mathrm{e}}}_j^{n} = \frac{{n_{\mathrm{e}}}_j^{n}}{2}\left(\left|E_{j-\frac{1}{2}}^{n}\right| \exp\left(-\frac{1}{\left|E_{j-\frac{1}{2}}^{n}\right|}\right) + \left|E_{j+\frac{1}{2}}^{n}\right| \exp\left(-\frac{1}{\left|E_{j+\frac{1}{2}}^{n}\right|}\right)\right), \tag{5.499}$$

$${j_{\mathrm{D}}}^{n} = \frac{1}{A_{\mathrm{E}} R_{\mathrm{S}}}\left({U_{\mathrm{S}}}^{n} - {U_{\mathrm{C}_1}}^{n} - {U_{\mathrm{C}_2}}^{n} - \Delta x\left(\frac{1}{2}E_0^n + \sum_{j=1}^{J} E_j^n + \frac{1}{2}E_{J+1}^n\right)\right). \tag{5.500}$$

Particle fluxes are given by the upwind scheme, accounting for flux direction. In the volume they are given as

$${\Gamma_{\mathrm{i}}}_{j+\frac{1}{2}}^{n} = \mu E_{j+\frac{1}{2}}^{n} \begin{cases} {n_{\mathrm{i}}}_j^{n} & \forall\, E_{j+\frac{1}{2}}^{n} \geq 0 \\ {n_{\mathrm{i}}}_{j+1}^{n} & \forall\, E_{j+\frac{1}{2}}^{n} < 0, \end{cases} \tag{5.501}$$

$${\Gamma_{\mathrm{e}}}_{j+\frac{1}{2}}^{n} = E_{j+\frac{1}{2}}^{n} \begin{cases} {n_{\mathrm{e}}}_{j+1}^{n} & \forall\, E_{j+\frac{1}{2}}^{n} \geq 0 \\ {n_{\mathrm{e}}}_j^{n} & \forall\, E_{j+\frac{1}{2}}^{n} < 0. \end{cases} \tag{5.502}$$

The boundary conditions for the particle fluxes are defined as

- For the ion flux we have

$$\Gamma_{\mathrm{i}}{}^{n}_{-\frac{1}{2}} = \mu E^{n}_{-\frac{1}{2}} \begin{cases} 0 & \forall\, E^{n}_{-\frac{1}{2}} \geq 0 \\ n_{\mathrm{i}0}^{n} & \forall\, E^{n}_{-\frac{1}{2}} < 0 \end{cases}, \tag{5.503}$$

$$\Gamma_{\mathrm{i}}{}^{n}_{J+\frac{1}{2}} = \mu E^{n}_{J+\frac{1}{2}} \begin{cases} n_{\mathrm{i}J}^{n} & \forall\, E^{n}_{J+\frac{1}{2}} \geq 0 \\ 0 & \forall\, E^{n}_{J+\frac{1}{2}} < 0 \end{cases}, \tag{5.504}$$

- For the electron flux, we have

$$\Gamma_{\mathrm{e}}{}^{n}_{-\frac{1}{2}} = E^{n}_{-\frac{1}{2}} \begin{cases} n_{\mathrm{e}0}^{n} & \forall\, E^{n}_{-\frac{1}{2}} \geq 0 \\ \gamma_{\mathrm{se}} \mu n_{\mathrm{i}0}^{n} + \Gamma_{\mathrm{ph}} & \forall\, E^{n}_{-\frac{1}{2}} < 0 \end{cases}, \tag{5.505}$$

$$\Gamma_{\mathrm{e}}{}^{n}_{J+\frac{1}{2}} = E^{n}_{J+\frac{1}{2}} \begin{cases} \gamma_{\mathrm{se}} \mu n_{\mathrm{i}J}^{n} + \Gamma_{\mathrm{ph}} & \forall\, E^{n}_{J+\frac{1}{2}} \geq 0 \\ n_{\mathrm{e}J}^{n} & \forall\, E^{n}_{J+\frac{1}{2}} < 0 \end{cases}. \tag{5.506}$$

*Remark 5.49* Here, the benefit are the monoscales for all the model equations, such that apply fast solver schemes and parallelize the explicit solvers. The main drawback are the very fine time steps that do not allow realistic approaches.

*Remark 5.50* The experiments are also enlarged in the case of an electric field pointing to an electrode ion impact secondary electron emission as well as photon induced secondary electron emission. Here, we apply time-dependent function for such emission problems.

### *5.8.4 Adaptive Explicit Scheme: With Two Timescales $\delta t$, $\Delta t$*

In the following, we discuss a more accelerated method, that taken into account the different macro- and microscales. Based on the different schemes, we apply adaptive time step schemes to the different models and accelerate the macroscopic model with larger time steps.

Here, we deal with a multiscale method, that embed the adaptivity based on different time steps, which are related to the different parts, e.g. macroscopic time step and microscopic time step. We do not consider a reconstruction or embedding of the microscopic scales into the macroscopic scales, while we have to apply the full time intervals for all of the micro- and macroscopic equation.

The macroscopic equation (slow equation) is given as

$$\frac{\partial n_{\mathrm{i}}}{\partial t} = -\tfrac{\partial}{\partial x}\left(\mu E n_{\mathrm{i}}\right) + |E n_{\mathrm{e}}|\, \mathrm{e}^{-\frac{1}{|E|}}, \tag{5.507}$$

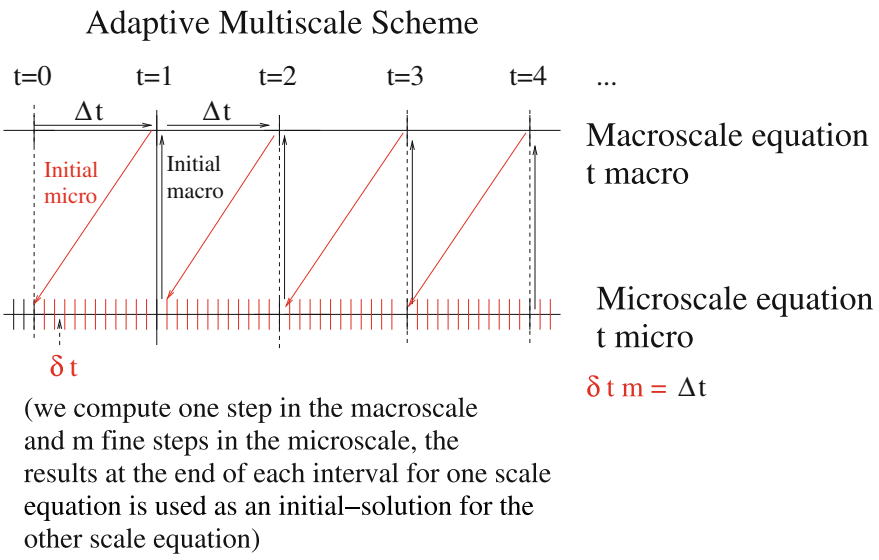



**Fig. 5.37** Two scales are coupled, fine scale (microscale) and slow scale (macroscale) via the end points in each corresponding scale

to the explicit discretization we apply a time step with $\Delta t$, which is limited via the CFL condition.

The fast equation (microscopic equation) is given as

$$\frac{\partial E}{\partial t} = j - (\mu n_{\mathrm{i}} - n_{\mathrm{e}})\, E, \tag{5.508}$$

based on the explicit discretization we have a time step $\delta t << \Delta t$-limit given by an *CFL* condition.

The idea of the adaptive method is given in Fig. 5.37.

The microscopic equation can also be integrated by a higher order scheme to gain a larger time step $\delta t$, see the method of Shu–Osher [119], were $L(y, t)$ is the right hand side of differential equation $\partial_t y = L(y, t)$

$$y_0 = y(t), \tag{5.509}$$

$$y_1 = y_0 + \Delta t\ L(y_0, t), \tag{5.510}$$

$$y_2 = \frac{3}{4} y_0 + \frac{1}{4} \left(y_1 + \Delta t\ L(y_1, t)\right), \tag{5.511}$$

$$y_3 = \frac{1}{3} y_0 + \frac{2}{3} \left(y_2 + \Delta t\ L(y_2, t)\right), \tag{5.512}$$

$$y(t + \Delta t) = y_3. \tag{5.513}$$

*Remark 5.51* The explicit time step control are given by the CFL condition of each scale equation, e.g. $CFL_{micro}$ and $CFL_{macro}$. By decoupling the multiscale equations, we can accelerate each single-scale equation with a parallel implementation of the solver schemes.

*Remark 5.52* We apply an implicit Scheme for the macroscopic equation, which is given as

$$\begin{aligned}\frac{n_{\mathrm{i}}(x_j, t_{n+1}) - n_{\mathrm{i}}(x_j, t_n)}{\Delta t} =& -\mu E(x_j, t_n)\frac{n_{\mathrm{i}}(x_{j+1}, t_{n+1}) - n_{\mathrm{i}}(x_j, t_{n+1})}{\Delta x} \\ &+ \left|E n_{\mathrm{e}}(x_j, t_n)\right| \mathrm{e}^{-\frac{1}{\left|E(x_j, t_n)\right|}},\end{aligned} \tag{5.514}$$

where we are unconditional stable and independent of the CFL condition and $n = 1, \ldots, N$.

The same idea can be applied to the microscopic equation, which is given as

$$\begin{aligned}\frac{E(x_j, t_{\tilde{n}+1}) - E(x_j, t_{\tilde{n}})}{\Delta t} =& \, j(x_j, t_{\tilde{n}n+1}) \\ &- \left(\mu n_{\mathrm{i}}(x_j, t_{\tilde{n}+1}) - n_{\mathrm{e}}(x_j, t_{\tilde{n}+1})\right) E(x_j, t_{\tilde{n}+1}),\end{aligned} \tag{5.515}$$

where we are unconditional stable and independent of the CFL condition and $\tilde{n} = 1, \ldots, N \cdot m$.

We result to a seamless time step, e.g. $\Delta t = \Delta t_{seam}$ with $\hat{n} = 1, \ldots, N \cdot \tilde{m}$ and $\tilde{m} > m$. Such that we can solve the implicit scheme as

$$\mathscr{N}(t^{\hat{n}+1}) = (I - \Delta t_{seam} B)^{-1} \mathscr{N}(t^{\hat{n}}), \tag{5.516}$$

where $\mathscr{N}(t^{\hat{n}+1}) = (\mathscr{N}_i(t^{\hat{n}+1}), \mathscr{N}_e(t^{\hat{n}+1}), \mathscr{E}(t^{\hat{n}+1}), \mathscr{J}(t^{\hat{n}+1}))^t$,
with $\mathscr{N}_i(t^{\hat{n}+1}) = (n_{\mathrm{i}}(x_1, t_{\hat{n}+1}), \ldots, n_{\mathrm{i}}(x_J, t_{n+1}))^t$,
$\mathscr{N}_i(t^{\hat{n}+1}) = (n_{\mathrm{e}}(x_j, t_{\tilde{n}+1}), \ldots, n_{\mathrm{e}}(x_J, t_{n+1}))^t$,
$\mathscr{E}(t^{\hat{n}+1}) = (E(x_1, t_{\hat{n}+1}), \ldots, E(x_J, t_{\hat{n}+1}))^t$,
$\mathscr{J}(t^{\hat{n}+1}) = (j(x_1, t_{\hat{n}+1}), \ldots, j(x_J, t_{n+1}))^t$ and $B \in \mathbb{R}^{4J \times 4J}$ is the resulting matrix of the semi-discretization and $J$ is the number of spatial grid points. $\Delta t_{seam}$ is the intermediate time step between the macroscopic and microscopic time step. The same notation is also done for $\mathscr{N}(t^{\hat{n}})$.

In the next section, we apply a EFM, which is a multiscale method and the microscopic equation, which is upscaled to the macroscopic equation.

### 5.8.5 *Equation-Free Explicit Scheme: With Two Timescale $\delta t$, $\Delta t$*

In the following scheme, we decompose the macro- and microscopic equation with a EFM (equation-free method) and the macroscopic solver is reconstructed via the extrapolation of the microscopic solver.

The macroscopic equation (slow equation) is given as

$$\frac{\partial n_\mathrm{i}}{\partial t} = -\frac{\partial}{\partial x}\left(\mu E n_\mathrm{i}\right) + |E n_\mathrm{e}|\, \mathrm{e}^{-\frac{1}{|E|}}, \tag{5.517}$$

to the explicit discretization we apply a time step with $\Delta t$, which is limited via the CFL condition.

The fast equation (microscopic equation) is given as

$$\frac{\partial E}{\partial t} = j - \left(\mu n_\mathrm{i} - n_\mathrm{e}\right) E, \tag{5.518}$$

based on the explicit discretization we have a time step $\delta t << \Delta t$-limit given by a CFL condition.

The idea of the EFM method is given in Fig. 5.38.

The EFM scheme is given as We have done the semi-discretization with the spatial operator $L = \frac{\partial}{\partial x}$, e.g. via upwinding, we apply the following parts to our EFM:

- Initialization of the microscopic equation (Lifting):

$$E^{n,0} = E^n, \tag{5.519}$$

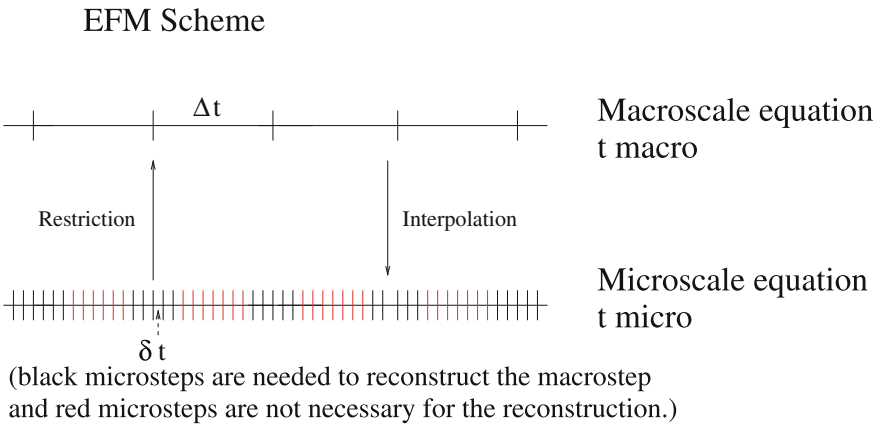



**Fig. 5.38** Two scales are coupled, fine scale (microscacale) and slow scale (macroscale) via the extrapolation steps in the microscopic scale

- Calculation of the microscopic equation (Evolving):

$$E^{n,m+1} = E^{n,m} + \delta t\, f(j^n, {n_\mathrm{i}}^n, {n_\mathrm{e}}^n, E^{n,m}), \tag{5.520}$$

where $f(j, n_\mathrm{i}, n_\mathrm{e}, E) = j - (\mu\, n_\mathrm{i} - n_\mathrm{e})\, E$ and $\Delta t = M\ \delta t$ is the macroscopic time step,
we apply $m = 0, \ldots, M-1$.

- Extrapolation into the macroscopic equation (Restriction):

$$E^n = E^n + (\Delta t - M\delta t)\frac{E^{n,m} - E^{n,m-1}}{\delta t}, \tag{5.521}$$

- The next macroscopic time step:

$${n_\mathrm{i}}^{n+1} = {n_\mathrm{i}}^n - L\left(\mu E^n {n_\mathrm{i}}^n\right) + \left|E^n n_\mathrm{e}\right| \mathrm{e}^{-\frac{1}{|E^n|}}. \tag{5.522}$$

*Remark 5.53* We control the numerical error of the multiscale scheme, while we compared different time step results. We start with different small timescales:

$$\delta t_1 < \delta t_2 \ll \Delta t, \tag{5.523}$$

and if the result:

$$||E_{\delta t_1} - E_{\delta t_2}|| \leq err, \tag{5.524}$$

we can apply the finer scale, while we are bounded below the error estimates.

### 5.8.6 Conclusions and Discussions

We present a multiscale approach based on a electronic model with macroscopic transport and microscopic electromagnetic equations. We discuss different approaches to overcome the disparate scales. While simple approaches are time-consuming, we discuss adaptive approaches and the equation-free method, which overcome the scale and embeded the microscopic model into the macroscopic model. Here, we accelerate the computations of the schemes, while applying larger time steps to the multiscale approaches. Extrapolation and seamless ideas are embedded with implicit schemes that overcome time restriction to the finer scales. Such strategies allow to accelerate the computations and speed up with parallel implementation the simulations.

## 5.9 Multiscale Approach for Complex Fluids: Applications in Non-Newtonian Fluids

**Abstract** The model problem is related to complex fluids. Such problems occurr in Non-Newtonian flow problems, while we have an underlying microstructure, e.g. polymeric fluids or complex interactions at the boundaries. Here, we discuss the modelling problems and the multiscale methods to overcome the multiscale problem. In a example, we present the interaction of the macroscopic and microscopic model.

### *5.9.1 Introduction*

Complex fluids are such problems with Non-Newtonian flows, means we have a complex nature in the constitutive molecules, e.g. polymeric fluids, geometric size of the problem as in microfluids, complex interactions at the boundary, chemical reactions at the fluid–fluid or fluid–solid interface, see [120, 121]. To describe such a problem a very detailed model such as a molecular dynamics (MD) model is needed. By the way, it is impossible to perform such a model and hybrid numerical methods as the multiscale methods, are important to take the efficiency of the continuum model and the accuracy of the molecular model into account.

### *5.9.2 Non-Newtonian Fluid: Influence of the Microscopic Model*

In the following example, which is discussed in [120], presents the influence of the microscopic model based on Non-Newtonian flow.

We have a macroscopic (conservation equation of incompressible flow) and a microscopic (dynamics of the fluid, Newton's law) equation.

The model equations are given in the following:

- Macroscopic equation (conservation equation) of an incompressible fluid:

$$\rho \partial_t v + \nabla \cdot \tau = f, \text{in } \Omega \times (0, T), \tag{5.525}$$

$$\nabla \cdot u = 0, \text{in } \Omega \times (0, T), \tag{5.526}$$

$$u(0) = u_0, \text{on } \Omega,$$

$$u = 0, \text{on } \partial\Omega \times (0, T).$$

where $v$ is the velocity field of the fluid, the momentum flux is given as $\tau = \rho v \otimes v + pI - \tau_{stress}$, $\rho$ is the density of the fluid, $p$ is the pressure, $\mu$ is the viscosity and $\tau_{stress}$ is the stress tensor.

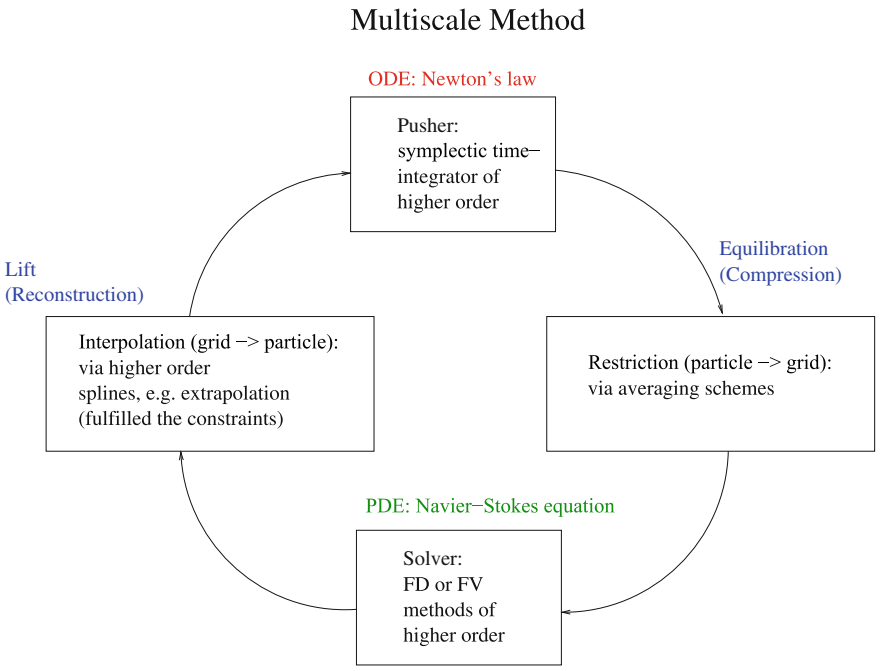



**Fig. 5.39** Multiscale solver based on the top-down principle, e.g. HMM

- Microscopic equation (Newton's equation of motion):

$$m_i \partial_{tt} x_i = F_i, i = 1, \ldots, N, \tag{5.527}$$

where $v_i = \partial_t x_i$ is the velocity of the particle $i$, $F_i$ is the force acting on particle $i$ via the interaction to the neighbour particles.

We apply the following coupling structure between the grid- and grid-free-solvers, see Fig. 5.39.

The coupling of the between the atomar and continuum model is done by the averaging microscopic stress tensors or reconstruction of the macroscopic velocity field to the microstructure.

- The averaging (atomar to continuum) we apply the Irving–Kirkwood formula, see [122], which is implemented as

$$\tilde{\tau}(\xi, t, x) = \sum_i \left( f_{free\ Energy}(v_i, \xi) + \sum_{j \neq i} f_{collision}(x_i, x_j, \xi) \right), \tag{5.528}$$

$$\tilde{\tau}(\xi, t, x) = -\sum_i m_i v_i v_i \delta(r_i - \xi)$$

$$-\frac{1}{2}\sum_{j\neq i}((r_j - r_i)f_{ij})\int_0^1 \delta(\lambda r_i + (1-\lambda)r_i - \xi)d\lambda \tag{5.529}$$

$$\tau(t,x) = \frac{1}{V}\int_V \tilde{\tau}(\xi,t,x)d\xi, \tag{5.530}$$

where the microscopic stress tensor is $\tilde{\tau}$ and the particles $i = 1,\ldots,I$, $x_i$ are the space coordinates of particle $i$, $v_i$ are the velocity coordinates of particle $i$ and the macroscopic stress tensor is given as $\tau$.

- The reconstruction (continuum to atomar) we apply the shape functions and interpolate the atomar velocities with the continuum velocities in each finite volume cell

$$v_i = v(S(x - x_i)),\ x_i, v_i \in V, \tag{5.531}$$

where $v$ is the velocity of the finite volume box (with the volume $V$). The interpolation function is given with $S$, e.g. spline function.

We deal with the following Algorithm 5.18.

**Algorithm 5.18** The algorithm is a top–down algorithm based on the HMM, see [120].

1. We solve the microscopic equation:

$$x_{i,j}^{(n,m+1)} = x_{i,j}^{(n,m)} + \delta t v_{i,j}^{(n,m)} - \frac{\delta t^2}{2m_{i,j}}\nabla U_{i,j}^{(n,m)},$$

$$v_{i,j}^{(n,m+1)} = v_{i,j}^{(n,m)} + \frac{\delta t}{2m_{i,j}}\nabla U_{i,j}^{(n,m)},$$

$$\tilde{\tau}(x_{i,j}^{n,m+1}, (m+1)\delta t, x_j^n) = g(x_{i,j}^m, v_{i,j}^m, \delta t),$$

where $i = 1,\ldots,I$, $m = 0,1,\ldots,M-1$, e.g. $\delta t \leq \Delta t/M$.

2. Equilibration of the micro-operators (compression):

$$\tau_j^n(\Delta t, x_j^n) = \frac{1}{V_j}\sum_i V_{i,j}\ \tilde{\tau}(x_{i,j}^{n,M}, \Delta t, x_j^n).$$

3. Solving the macroscopic equation on the grid:

$$v_j^{n+1} = v_j^n + \Delta t A v_j^n + \Delta t \tau_j^n, \tag{5.532}$$

where $A$ is the stiffness matrix based on the finite volume discretization with the finite volume boxes $V_j$ and $j = \{1,\ldots,J\}$.

4. Reconstruction of the microscopic velocities in each finite volume cell:

$$v_{i,j} = v_j(S(x_j - x_{i,j})),\ x_{i,j}, v_{i,j} \in V_j, \text{with } j = \{1,\ldots,J\}, \tag{5.533}$$

where $J$ is the number of finite volume boxes.
Then we go to the step (1) with $n = n + 1$, till the time frame is done.

*Remark 5.54* We apply the discrete macroscopic equation with the large macroscopic time steps and embed the microscopic equation with respect to the stress tensor to the macroscopic equation. Therefore, we save computational time, while we only resolve partially the microscopic scale, e.g. in the finite volume boxes.

### 5.9.3 Non-Newtonian Fluid: Influence of the at the Boundary Flow at the Channel

In the following example, which is discussed in [123], presents the influence of the boundary flow at the channel lower and upper walls. Here, we have a Non-Newtonian flow at the region of the walls.

We have a macroscopic (Navier–Stokes equation) and a microscopic (dynamics of the fluid, Newton's law) equation. Both equations are coupled near the walls, while in the core of the channel, we apply the pure Navier–Stokes equation.

- The macroscale equation is given by the Navier–Stokes equation for incompressible continuum flow:

$$\rho \partial_t u + \rho(u \cdot \nabla)u - \mu \Delta u + \nabla p = \rho f, \text{ in } \Omega \times (0, T), \tag{5.534}$$
$$\nabla \cdot u = 0, \text{ in } \Omega \times (0, T), \tag{5.535}$$
$$u(0) = u_0, \text{ on } \Omega,$$
$$u = 0, \text{ on } \partial\Omega \times (0, T),$$

  The unknown flow vector $u = u(x, t)$ is considered in $\Omega \times (0, T)$. In the above equations, $\rho$ and $p$ represent the fluid density and pressure, respectively. Here, $\mu$ represents the dynamic viscosity of the fluid.
- The microscopic equation is given by Newton's equation of motion for each individual molecule $i$ for a sample of $N$ molecules,

$$m_i \partial_{tt} x_i = F_i, i = 1, \ldots, N, \tag{5.536}$$

  here, $x_i$ is the position vector of atom $i$, and the force $F_i$ acting on each molecule is the result of the intermolecular interaction of a molecule $i$ with the neighbouring molecular within a finite interaction range.

Further we assume to have a Lennard–Jones interaction potential, see [124] and that we can couple the two models by the viscous stress conditions, see [123].

The viscous stress contribution $\mu \Delta v$ in Eq. (5.353) can be modified for a non-Newtonian flow as $\partial \sigma_{ij} / \partial x_j$, and we have

$$\partial\sigma_{ij}/\partial x_j = \mu_{apparent}\partial^2 v_i/\partial x_j^2 \tag{5.537}$$

where, we assume the we deal with an apparent viscosity, which can be implemented and computed by non-Newtonian flow conditions, e.g. moleculcar dynamics computations.

So, finally, we obtain the coupled multiscale equations:

$$\begin{aligned} \rho\partial_t u + \rho(u\cdot\nabla)u - \mu_{approx}\Delta u + \nabla p = f, \text{in } \Omega\times(0,T), \\ f_i = \partial\sigma_{ij}/\partial x_j|_{molecular} - \mu_{approx}\Delta v_i \end{aligned} \tag{5.538}$$

where Einstein's summation is used for the volumetric source term $f$, which accounts for the deviation of the viscous stresses evaluated at molecular-level from the approximate Newtonian relation $\mu_{approx}\Delta v$.

We apply the following multiscale method as an adaptive fixed-point approach. Further the Algorithm 5.19 is given in the following, see also [123].

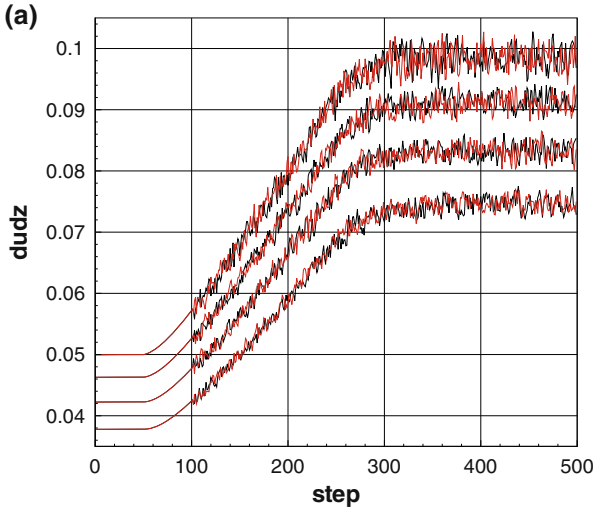
**Algorithm 5.19** On a uniform time grid with $t^n = t_0 + n\Delta t, n = 0,\ldots,N$, (where $N$ is given), the discretized coupled macro- and microscale equations are integrated in time from time level $n$ to $n+1$ using the following adaptive scheme:

(1) For the cell faces with microscale fluxes, compute the velocity gradients

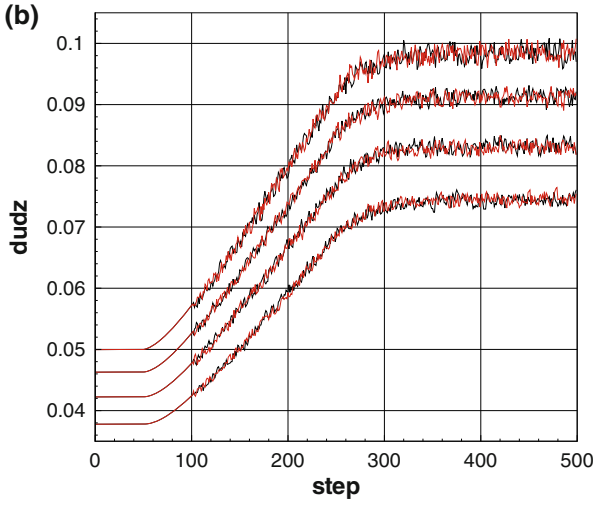
(2) Velocity gradient criterion: $\max\left(\left|\left(\frac{\partial u_i}{\partial x_j}\right)^n_{micro} - \left(\frac{\partial u_i}{\partial x_j}\right)^{last}_{micro}\right|\right) > crit_{grad}, crit_{grad} \in \mathbb{R}^+$. If criterion is satisfied goto (3) else (4)

(3) Dual-time step with fixed-point iteration. Perform the following steps:

  (i) Using the last estimate of the microscale apparent viscosity, compute an estimate of the updated velocity field at the present time step using dual-time step update.
  (ii) Compute the velocity normalization factor as $u_{norm} = \max(\tilde{u}_i(t^{n+1} - u_i(t^n)$, where $\tilde{u}$ denotes the estimated velocity from step (i)
  (iii) Compute viscous flux correction normalization as $f_{norm} = \max(\tilde{f}_i(t^{n+1})$, with $\tilde{f}$ the viscous flux correction based on the last apparent viscosity computation
  (iv) Store microscale velocity gradients: $\left(\frac{\partial u_i}{\partial x_j}\right)^{last}_{micro} = \left(\frac{\partial u_i}{\partial x_j}\right)^n_{micro}$
  (v) Initialize the Molecular Dynamics microscale problems with the imposed velocity gradients from the finite volume cell faces and integrate these through the initial equilibration phase (e.g. $t_{equi} = 50\tau$)
  (vi) Fixed-point iteration:
    (a) integrate microscale problems in time through an additional $10\tau$ and sample apparent viscosity through total microscale sampling time and ensemble average over $n_{ensemble}$ independent realizations
    (b) construct the viscous flux corrections $f_i$ using updated apparent viscosity
    (c) perform dual-time step update using $n_{Newton}$ relaxation steps




Velocity gradients near walls - 10 fixed-point inner iterations




Velocity gradients near walls - 15 fixed-point inner iterations

**Fig. 5.40** Channel flow with time-dependent pressure gradient. Dual-time stepping method with fixed-point iteration in MD sampling of apparent viscosity. Finite-volume discretization method with approximated statistical scatter of molecular dynamics viscous fluxes as function of sampling duration on first 4 cells near lower and upper walls

(d) check convergence of fixed-point iteration using the stop criteria: $\left|u_i(t^{n+1}) - u_{i-1}(t^{n+1}\right|/u_{norm} \leq err_u$, $\text{err}_u \in \mathbb{R}^+$ and/or $\left|f_i(t^{n+1}) - f_{i-1}(t^{n+1}\right|/f_{norm} \leq err_f$, $\text{err}_f \in \mathbb{R}^+$
(e) if criterion is satisfied time step is completed, else go to step (a)

(4) Dual-time step without fixed-point iteration.

(i) Using the last estimate of the microscale apparent viscosity, compute flux corrections $f_i$
(ii) Perform dual-time step update using $n_{Newton}$ relaxation steps

if $n < N$ go to (1)

In the following experiment, we deal with the dual-time step formulation as multiscale approach, each fixed-point iteration corresponds to the solution of a 'pseudo-steady' problem using a Newton relaxation method.

For each increment of the fixed-point iteration counter, the microscale problem is March ed forward by a pre-defined time-increment (in the present section, 10 Lennard–Jones time units (macroscopic scale), corresponding to 10,000 time steps in the Molecular Dynamics method).

So the coupled idea is based by a relaxation between the macro- and microscopic model.

The results with different iterative steps is given in Fig. 5.40. We simulate the wall influence based on the microscopic fluid. We see an relaxation with larger iterative steps at the macro–micro interface.

*Remark 5.55* The benefit of the modelling approach was the decomposition of the core and wall computations. While the core of the channel is assumed to be only a macroscopic model, the complex fluid is only assumed near the wall interface. Based on the iterative approach to couple micro- and macroscopic model at the interface, we only have to update the macroscopic volume cells at the wall. Therefore, we could accelerate the computations. Hiher coupling approaches, means more iterative steps, smoothen the influence of the microscopic approach, see [123].

## References

1. L. Rosso, A.F de Baas., Review of materials modelling: what makes a material function? Let me compute the ways... European Commission, General for Research and Innovation Directorate, Industrial Technologies, Unit G3 Materials (2014). http://ec.europa.eu/research/industrial_technologies/modelling-materials_en.html
2. H. Risken, *The Fokker-Planck Equation Methods of Solution and Applications*, Series in Synergetics, vol. 18, 3rd edn. (Springer, Berlin, 1996)
3. J. Geiser, Recent advances in splitting methods for multiphysics and multiscale: theory and applications. J. Algorithms Comput. Technol., Multi-Sci., Brentwood, Essex, UK, accepted August 2014 (to be published second issue 2015)

4. J. Geiser, Additive via iterative splitting schemes: algorithms and applications in heat-transfer problems, in *Proceedings of the Ninth International Conference on Engineering Computational Technology*, ed. by P. Ivanyi, B.H.V. Topping (Civil-Comp Press, Stirlingshire, 2014), Paper 51. doi:10.4203/ccp.105.51
5. B.I. Cohen, A.M. Dimits, A. Friedman, R.E. Caflisch, Time-step considerations in particle simulation algorithms for Coulomb collisions in plasmas. IEEE Trans. Plasma Sci. **38**(9), 2394–2406 (2010)
6. K. Nanbu, Theory of cumulative small-angle collisions in plasmas. Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top. **55**(4), 4642–4652 (1997)
7. T. Takizuka, H. Abe, A binary collision model for plasma simulation with a particle code. J. Comput. Phys. **25**, 205–219 (1977)
8. B.I. Cohen, L. Divol, A.B. Langdon, E.A. Williams, Effects of ion-ion collisions and inhomogeneity in two-dimensional kinetic ion simulations of stimulated Brillouin backscattering. Phys. Plasmas **13**(2), 022705 (2006)
9. M.E. Jones, D.S. Lemons, R.J. Mason, V.A. Thomas, D. Winske, A grid-based Coulomb collision model for PIC codes. J. Comput. Phys. **123**(1), 169–181 (1996)
10. D.S. Lemons, D. Winske, W. Daughton, B. Albright, Small-angle Coulomb collision model for particle-in-cell simulations. J. Comput. Phys. **228**(5), 1391–1403 (2009)
11. W.M. Manheimer, M. Lampe, G. Joyce, Newblock Langevin representation of Coulomb collisions in PIC simulations. J. Comput. Phys. **138**(2), 563–584 (1997)
12. M. Sherlock, A Monte-Carlo method for Coulomb collisions in hybrid plasma models. J. Comput. Phys. **227**(4), 2286–2292 (2008)
13. A.M. Dimits, B.I. Cohen, R.E. Caflisch, M.S. Rosin, L.F. Ricketson, Higher-order time integration of Coulomb collisions in a plasma using Langevin equations. J. Comput. Phys. **242**, 561–580 (2013)
14. J. Geiser, in *Iterative Splitting Methods for Differential Equations*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.-H. Lai (Chapman & Hall/CRC, 2011)
15. J. Geiser, Multiscale splitting for stochastic differential equations: applications in particle collisions. J. Coupled Syst. Multiscale Dyn. **1**(2), 241–250(10) (2013)
16. E. Hairer, C. Lubich, G. Wanner, Geometric numerical integration illustrated by the Störmer Verlet method. Acta Numer. **12**, 399–450 (2003)
17. E. Weinan, *Principle of Multiscale Modelling* (Cambridge University Press, Cambridge, 2010)
18. J. Geiser, in *Coupled Systems: Theory, Models and Applications in Engineering*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.-H. Lai (CRC Press, Chapman & Hall/CRC, Boca Raton, 2014)
19. R. Glowinski, Numerical methods for fluids, in *Handbook of Numerical Analysis*, vol. IX, ed. by P.G. Ciarlet, J. Lions (North-Holland Elsevier, Amsterdam, 2003)
20. R. Glowinski, P.G. Ciarlet, J.L. Lions (eds.), Numerical Methods for Non-Newtonian Fluids: Special Volume, *Handbook of Numerical Analysis*, vol. XVI (North-Holland Elsevier, Amsterdam, 2010)
21. J. Geiser, St. Guettel, Coupling methods for heat-transfer and heat-flow: operator splitting and the parareal algorithm. J. Math. Anal. Appl. **388**(2), 873–887 (2012) (Elsevier, North Holland)
22. M.J. Gander, S. Vanderwalle, Analysis of the parareal time-parallel time-integration method. SIAM J. Sci. Comput. **29**(2), 556–578 (2007)
23. OpenFOAM, OpenFOAM Softwarepackage (2004). http://www.openfoam.com/, 2004-2013 OpenCFD Ltd (ESI Group), Bracknell, UK, 2014
24. Ricardo Software, VECTIS, three-dimensional fluid dynamics program (2010). http://www.ricardo.com/What-we-do/Software/Products/VECTIS/
25. O. Arici, S. Yang, D. Huang, E. Oker, Computer model for automobile climate control system simulation and application. Int. J. Appl. Thermodyn. **2**(2), 59–68 (1999)
26. J. Geiser, Iterative operator-splitting methods for nonlinear differential equations and applications. Numer. Methods Partial Differ. Equ. **27**(5), 1026–1054 (2011)

27. S. Descombes, Convergence of a splitting method of high order for reaction-diffusion systems. Math. Comput. **70**, 1481–1501 (2001)
28. J. Salcedo Rulz, F.J. Sanchez Bernabe, A numerical study of stiffness effects on some higher order splitting methods. Revista Mexicana de Fisica **52**(2), 129–134 (2006)
29. G. Strang, On the construction and comparison of difference schemes. SIAM J. Numer. Anal. **5**, 506–517 (1968)
30. I. Farago, J. Geiser, Iterative operator-splitting methods for linear problems. Int. J. Comput. Sci. Eng. **3**(4), 255–263 (2007)
31. J. Geiser, Iterative operator-splitting methods with higher order time-integration methods and applications for parabolic partial differential equations. J. Comput. Appl. Math. **217**, 227–242 (2008). (Elsevier, Amsterdam)
32. S.A. Chin, The complete characterization of fourth-order symplectic integrators with extended-linear coefficients. Phys. Rev. E **73**, 026705 (2006)
33. C.T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*. SIAM Frontiers in Applied Mathematics, vol. 16 (SIAM, Philadelphia, 1995)
34. J. Kanney, C. Miller, C.T. Kelley, Convergence of iterative split-operator approaches for approximating nonlinear reactive transport problems. Adv. Water Res. **26**, 247–261 (2003)
35. E. Zeidler, *Nonlinear Functional Analysis and its Applications. II/B Nonlinear Montone Operators* (Springer, Berlin, 1990)
36. K.H. Karlsen, N. Risebro, An operator splitting method for nonlinear convection-diffusion equation. Numer. Math. **77**(3), 365–382 (1997)
37. M.V. Berry, The Levitron: an adiabatic trap for spins. Proc. R. Soc. Lond. A **452**, 1207–1220 (1996)
38. J. Geiser, K.F. Lüskow, R. Schneider, Levitron: multi-scale analysis of stability. Dyn. Syst. **29**(2), 208–224 (2014)
39. H.R. Dullin, Poisson integrator for symmetric rigid bodies. Regul. Chaotic Dyn. **9**, 255–264 (2004)
40. J. Geiser, Multiscale methods for Levitron problems: theory and applications. Comput. Struct. **122**, 27–32 (2013). (Elsevier, North Holland)
41. J. Geiser, Nonlinear extension of multiproduct expansion schemes and applications to rigid bodies. Int. J. Differ. Equ. (Hindawi Publishing Corporation, New York, USA, accepted, August 2013)
42. R.I. McLachlan, P. Atela, The accuracy of symplectic integrators. Nonlinearity **5**, 541–562 (1992)
43. H. Yoshida, Recent process in the theory and application of symplectic integrators. Celest. Mech. Dyn. Astron. **56**, 27–43 (1993)
44. E. Hairer, C. Lubich, G. Wanner, *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. SCM, vol. 31 (Springer, Berlin, 2002)
45. S. Earnshaw, On the nature of the molecular forces which regulate the constitution of the luminiferous ether. Trans. Camb. Philos. Soc. **7**, 97–112 (1842)
46. H.R. Dullin, R. Easton, Stability of Levitron. Phys. D: Nonlinear Phenom. **126**(1–2), 1–17 (1999)
47. R.F. Gans, T.B. Jones, M. Washizu, Dynamics of the Levitron. J. Phys. D **31**, 671–679 (1998)
48. J. Geiser, K.F. Lueskow, R. Schneider, Iterative implicit methods for solving nonlinear dynamical systems: application in Levitron problems, in *Proceeding of the 6th Conference on FDM*. Lecture Notes in Computer Science (LNCS) (Springer, accepted May 2014)
49. S. Vandewalle, *Parallel Multigrid Waveform Relaxation for Parabolic Problems*. Teubner Skripten zur Numerik (B.G. Teubner, Stuttgart, 1993)
50. Y.-L. Jiang, O. Wing, A note on convergence conditions of waveform relaxation algorithms for nonlinear differential algebraic equations. Appl. Numer. Math. **36**(2–3), 281–297 (2001)
51. P. Console, E. Hairer, Ch. Lubich, Symmetric multistep methods for constrained Hamiltonian systems. Numerische Mathematik **124**, 517–539 (2013)
52. J. Geiser, Model order reduction for numerical simulation of particle transport based on numerical integration approaches (UK, accepted, Mathematical and Computer Modelling of Dynamical Systems, Taylor and Francis, Abingdon, October 2013)

53. M.E. Innocenti, G. Lapenta, S. Markidis, A. Beck, A. Vapirev, A multi level multi domain method for particle in cell plasma simulations. J. Comput. Phys. **238**, 115–140 (2013)
54. K. Lueskow, J. Duras, O. Kalentev, K. Matyash, J. Geiser J, R. Schneider, D. Tskhakaya. Non-equidistant particle-in-cell for ion thruster plumes, in *Proceedings of the 33rd IEPC*, Washington, DC, IEPC-2013-067, October 2013
55. R. Hockney, J. Eastwood, *Computer Simulation Using Particles* (CRC Press, Boca Raton, 1985)
56. G. Lapenta, DEMOCRITUS: an adaptive particle in cell (PIC) code for object-plasma interactions. J. Comput. Phys. **230**(12), 4679–4695 (2011)
57. D. Tskhakaya, K. Matyash, R. Schneider, F. Taccogna, The particle-in-cell method. Contrib. Plasma Phys. **47**(8–9), 563–594 (2007)
58. J. Duras, K. Matyash, D. Tskhakaya, O. Kalentev, R. Schneider, Self-force in 1D electrostatic particle-in-cell codes for non-equidistant grids. Contrib. Plasma Phys. **54**(8), 697–711 (2014)
59. P. Colella, P.C. Norgaard, Controlling self-force errors at refinement boundaries for AMR-PIC. J. Comput. Phys. **229**, 947–957 (2010)
60. M.E. Innocenti, G. Lapenta, S. Markidis, A. Beck, A. Vapirev, A multi level multi domain method for particle in cell plasma simulations (2012). arXiv:1201.6208v1 [physics.plasm-ph]
61. W. Hackbusch, *Elliptic Differential Equations. Theory and Numerical Treatment*. Springer Series in Computational Mathematics, vol. 18 (Springer, Berlin, 1992)
62. St. McDonald, Finite difference approximation for linear stochastic partial differential equations with method of lines. MPRA paper no. 3983 (2007). http://mpra.ub.uni-muenchen.de/3983
63. D.W. Kelly, R.J. Millis, J.A. Reizes, A posteriori error estimates in finite difference techniques. J. Comput. Phys. **74**, 214–232 (1998)
64. W. Bangerth, R. Rannacher, *Adaptive Finite Element Methods for Differential Equations* (Birkhäuser, Boston, 2003)
65. J.T. Oden, J.N. Reddy, *Variational Methods in Theoretical Mechanics* (Springer, Berlin, 1976)
66. P. Ciarlet, *The Finite Element Method for Elliptic Problems* (North Holland, Amsterdam, 1975)
67. B.D. Vujanovic, T.M. Atanackovic, *An Introduction to Modern Variational Techniques in Mechanics and Engineering* (Birkhauser, Boston, 2004)
68. A. Jentzen, P.E. Kloeden, The numerical approximation of stochastic partial differential equations. Milan J. Math. **77**(1), 205–244 (2009)
69. P.E. Kloeden, E. Platen, *The Numerical Solution of Stochastic Differential Equations* (Springer, Berlin, 1992)
70. M.A. Lieberman, A.J. Lichtenberg, *Principle of Plasma Discharges and Materials Processing*, 2nd edn. (Wiley-Interscience, New York, 2005)
71. V.J. Ervin, W.W. Miles, Approximation of time-dependent, multi-component, viscoelastic fluid flow. Comput. Methods Appl. Mech. Eng. **194**(18–20), 2229–2255 (2005)
72. A. Fick, On liquid diffusion. Philos. Mag. **10**, 30–39 (1855)
73. A. Fick, On liquid diffusion. J. Membr. Sci. **100**, 33–38 (1995)
74. J.C. Maxwell, On the dynamical theory of gases. Philos. Trans. R. Soc. **157**, 49–88 (1866)
75. J. Stefan, Ueber das Gleichgewicht und die Bewegung insbesondere die Diffusion von Gasgemengen. Akad. Wiss. Wien **63**, 63–124 (1871)
76. R. Balescu, *Transport Processes in Plasma: Classical Transport*, vol. 1 (North Holland Publ., Amsterdam, 1988)
77. C. Le Bris, T. Lelievre, *Multiscale Modeling and Simulation in Science*. Lecture Notes in Computational Science and Engineering, vol. 66 (2009), pp. 49–137
78. R. Krishna, R. Taylor, Multicomponent mass transfer theory and applications, in *Handbook for Heat and Mass Transfer*, vol. 2, Chapter 7, ed. by N. Cheremisinoff (Gulf, Houston, 1986)
79. R. Krishna, J. Wesselingh, The Maxwell-Stefan approach to mass transfer. Chem. Eng. Sci. **52**, 861–911 (1997)
80. D. Bothe, On the Maxwell-Stefan approach to multicomponent diffusion, *Parabolic Problems, Progress in Nonlinear Differential Equations and Their Applications*, vol. 80 (Springer, Basel, 2011), pp. 81–93

81. K. Böttcher, Numerical solution of a multi-component species transport problem combining diffusion and fluid flow as engineering benchmark. Int. J. Heat Mass Transf. **53**, 231–240 (2010)
82. T.K. Senega, R.P. Brinkmann, A multi-component transport model for non-equilibrium low-temperature low-pressure plasmas. J. Phys. D: Appl. Phys. **39**, 1606–1618 (2006)
83. M.K. Gobbert, C.A. Ringhofer, An asymptotic analysis for a model of chemical vapor deposition on a microstructured surface. SIAM J. Appl. Math. **58**, 737–752 (1998)
84. S. Chapman, Th.G. Cowling, *The Mathematical Theory of Non-uniform Gases: An Account of the Kinetic Theory of Viscosity, Thermal Conduction, and Diffusion in Gases* (Cambridge University Press, Cambridge, 1990)
85. L. Boudin, B. Grec, F. Salvarani, A mathematical and numerical analysis of the Maxwell-Stefan diffusion equation. Discrete Contin. Dyn. Syst. Ser. B **17**(5), 1427–1440 (2012)
86. L. Boudin, B. Grec, F. Salvarani, The Maxwell-Stefan diffusion limit for a kinetic model of mixtures. Unpublished paper, INRIA - Laboratoire Jacques-Louis Lions - Universite Pierre et Marie Curie, Paris (2013). http://hal.archives-ouvertes.fr/hal-00554744
87. P.J. Antsaklis, A.N. Michel, *Linear Systems* (Birkhäuser, Boston, 2005). (Corrected edition)
88. S. Blanes, F. Casas, J.A. Oteo, The Magnus expansion and some of its applications. Phys. Rep. **470**(5–6), 151–238 (2009)
89. S. Blanes, P.C. Moan, Fourth- and sixth-order commutator free Magnus integrators for, linear and nonlinear dynamical systems. Appl. Numer. Math. **56**, 1519–1537 (2006)
90. F. Casas, A. Iserles, Explicit Magnus expansions for nonlinear equations. J. Phys. A: Math. Gen. **39**, 5445–5461 (2006)
91. D. Braess, *Finite Elemente* (Springer, Berlin, 1992)
92. R. Eymard, T.R. Gallouet, R. Herbin, The finite volume method, in *Handbook of Numerical Analysis*, vol. VII, ed. by P.G. Ciarlet, J.L. Lions (2000), pp. 713–1020
93. V. Giovangigli, *Multicomponent Flow Modeling* (Birkhäuser, Boston, 1999)
94. F. Hynne, S. Dano, P.G. Sorensen, Full-scale model of glycolysis in Saccharomyces cerevisiae. Biophys. Chem. **94**(1–2), 121–163 (2001)
95. P.D. Kourdis, D.A. Goussis, Glycolysis in saccharomyces cerevisiae: algorithmic exploration of robustness and origin of oscillations. Math. Biosci. **243**, 190–214 (2013)
96. H. Holden, K.H. Karlsen, K.-A. Lie, N.H. Risebro, *Splitting Methods for Partial Differential Equations with Rough Solutions*. EMS Series of Lectures in Mathematics (2010)
97. B. Teusink, J. Passarge, C.A. Reijenga et al., Can yeast glycolysis be understood in terms of in vitro kinetics of the constituent enzymes? Testing biochemistry. Eur. J. Biochem. **267**(17), 5313–5329 (2000)
98. D.K. Arrowsmith, C.M. Place, *An Introduction to Dynamical Systems* (Cambridge University Press, Cambridge, 1990)
99. E. Hansen, A. Ostermann, Exponential splitting for unbounded operators. Math. Comput. **78**, 1485–1496 (2009)
100. E. Hansen, A. Ostermann, High order splitting methods for analytic semigroups exist. BIT **49**, 527–542 (2009)
101. F.A. Williams, *Combustion Theory, The Fundamental Theory of Chemically Reacting Systems*, 2nd edn. (Benjamin and Cummings Pub. Co., Menlo Park, 1985)
102. S.H. Lam, D.A. Goussis, The CSP method for simplifying kinetics. Int. J. Chem. Kinet. **26**, 461–486 (1994)
103. S.H. Lam, in *Singular Perturbation for Stiff Equations Using Numerical Methods*, Recent Advances in the Aerospace Sciences, ed. by C. Casi (Plenum Press, New York, 1985), pp. 3–20
104. G.H. Golub, Ch.F. Van Loan, *Matrix Computations*, 3rd edn. (Johns Hopkins University, Baltimore, 1996)
105. J. Kevorkian, J.D. Cole. *Multiple Scale and Singular Perturbation Methods* (Springer, Berlin 1996)
106. W. Hundsdorfer, J.G. Verwer, *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations* (Springer, Berlin, 2003)

107. A. Zagaris, H.G. Kaper, T.J. Kaper, Fast and slow dynamics for the computational singular perturbation method. Multiscale Model. Simul. **2**(4), 613–638 (2004)
108. L. Tonks, I. Langmuir, Oscillations in Ionized gases. Phys. Rev. **33**, 195 (1929)
109. J. Oberrath, T. Mussenbrock, R.P. Brinkmann, Active plasma resonance spectroscopy: a kinetically functional analytic description. Preprint, TET, Ruhr University of Bochum (2013)
110. A. Narayan, A. Klöckner. *Deterministic Methods for the Boltzmann Equation.* Lecture Notes (2013). http://mathema.tician.de/dl/academic/talks/boltzmann-notes.pdf
111. G.I. Marchuk, Some applications of splitting-up methods to the solution of problems in mathematical physics. Aplikace Matematiky **1**, 103–132 (1968)
112. M. Bjorhus, Operator splitting for abstract Cauchy problems. IMA J. Numer. Anal. **18**(3), 419–443 (1998)
113. J. Geiser, in *Decomposition Methods for Partial Differential Equations: Theory and Applications in Multiphysics Problems*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, F. Lai (CRC Press, Chapman & Hall/CRC, Boca Raton, 2009)
114. J. Geiser, An iterative splitting method via waveform relaxation. Int. J. Comput. Math. **88**(17), 3646–3665 (2011). (Taylor and Francis, New York)
115. T. Ladics, I. Farago, Generalizations and error analysis of the iterative operator splitting method. Cent. Eur. J. Math. **11**(8), 1416–1428 (2013)
116. Mathematika, *Software-Package: Mathematika*. Wolfram Mathematica (2015). http://www.wolfram.com/mathematica/
117. A. Wollny, R.-P. Brinkmann, Plasma-plasma interaction-simulations of ionization wave propagation on micro cavity plasma arrays, in *Proceeding of the Conference WELTPP-17*, Kerkrade, The Netherlands, 20–21 November 2014
118. L. Friedland, Ju.M. Kagan, Generalized theory of first Townsend ionization coefficient in strong electric fields. J. Appl. Phys. **54**, 4947 (1983)
119. C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes. J. Comput. Phys. **77**, 439–471 (1988)
120. W. Ren, E. Weinan, Heterogeneous multiscale method for the modeling of complex fluids and micro-fluidics. J. Comput. Phys. **204**(1):1–26 (2005)
121. C. Le Bris, T. Lelievre, Multiscale modelling of complex fluids: a mathematical initiation. Research report, RR-6275 (2007). https://hal.inria.fr/inria-00165171
122. J. Irving, J. Kirkwood, The statistical mechanical theory of transport processes IV. J. Chem. Phys. **18**, 817–829 (1950)
123. J. Geiser, Coupled Navier Stokes-molecular dynamics simulation using iterative operator-splitting methods. Comput. Fluids. **77**(1), 97–111 (2013). (Elsevier, North Holland)
124. J.E. Lennard-Jones, On the determination of molecular fields. Proc. R. Soc. Lond., A **106**(738), 463–477 (1924)

# Conclusions

## Conclusions and Perspectives

In this monograph, we discuss the numerical methods, models and applications to multicomponent and multiscale systems. We cover many fields of engineering problems, ranging from reactive flow simulations to electronic applications. The model equations are based on coupled deterministic and stochastic differential equations with multiple time- and spatial-scales. The differential equations are given with multiple components, which can have different time and space scales, hence we deal with large coupled system partial differential equations in our underlying engineering problems.

Such multicomponent and multiscale systems are delicate to simulate, while the main problem is disparate time- and spatial-scales. Such multiscale problems need multiscale treatment with underlying multiscale solvers to resolved each sub-scale problem with adequate time- and spatial-schemes.

We present multiscale methods and their numerical implementation, which can be applied to solve such delicate systems and close the gap between numerical methods and their practical application in engineering problems.

In the theoretical part, we present different numerical approaches based on multiscale methods and multicomponent methods, which partitioned the full system into smaller and simpler solvable partial systems. We discuss the numerical errors of such approximation methods, while we separate into simpler regimes or upscale multiscale regimes and neglect some information, see [1]. Based on the new formulation of problems, by application of numerical schemes, we discuss the different behaviour of such mono-component or single-scale approximations and overcome such errors with higher accurate schemes, see [2].

The following problems are discussed and solved in the monograph:

- The extension of standard numerical scheme with respect to multicomponent and multiscale applications.
- General principles, which can be applied for multicomponent and multiscale methods to understand their behaviour, are used to develop new methods.



J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5

- Different iterative and additive methods are discussed with respect to their flexibility to solve multicomponent and multiscale systems.
- We discuss the modifications of the methods, their errors and how to overcome such errors and method drawbacks.

In various applications, we present solutions with different numerical schemes that are developed to multiscale and multicomponent method using the following ideas:

- Separating of the different components;
- Separating of the different scales;
- Averaging of scales to upscale fast scales and dealing with an averaged scale;
- Reduction of models by compressing and skipping unnecessary components in the models.

Further, we apply preferred and embed implicit schemes to have an averaging behaviour and incorporate the finer scales of the model problems. Therefore, we could apply larger time- and spatial-scales and reduce the amount of computations.

In the future, the methods for multicomponent and multiscale systems can be seen as a combination of different sub-methods, for example:

- Decomposition methods (time and space);
- Adaptive methods and analytical methods;
- Multiple scale methods,
- Implicit and explicit discretization methods;
- Model-reduction methods.

It is important to obtain an overview of the complex models and those problems in the model that have to be studied more carefully.

Further, the mathematical correctness of the methods is important, which errors will be embedded (e.g. splitting errors, reduction errors of no applied components, etc.), which errors can be skipped or are not important to the modified system. It is important to find a balance between a simplification, e.g. smaller equation system with lesser components and scales or decomposing into simpler parts of the equation with numerical errors of the decomposition method, and the necessity for resolving the physical behaviour of the model, which needs a nearly full model description, see delicate plasma models [3, 4].

We present different applications which allow to apply such modifications in simpler equation systems, which can be solved much faster and allow to have high accuracy in each scale and their components.

In the book, we could apply and extend the ideas of splitting methods and multiple scale methods into a general spectrum of multicomponent and multiscale systems and derive applicable schemes which are used in practical experience and engineering software.

Such a scientific tool allows engineers to deal more with simpler and understandable problems, which are testable, without losing the context to the full multicomponent and multiscale system.

In the near future, when we deal with engineering complexities as in the framework of Horizon 2020, see [5], combinations of multiscale and multicomponent systems in engineering will be an important part of studying complex fluid or solid problems. To understand the modelling equation is delicate and to apply accurate solver methods to perform simulations is important. Here, we gave contributions to methods which are related in this context and could present a method-toolbox to apply to complex processes. We gave a theoretical overview of the relevant multiscale and multicomponent methods and also their applications to engineering problems, so that scientists and practitioners can use our underlying ideas.

## References

1. E. Weinan, *Principle of Multiscale Modelling* (Cambridge University Press, Cambridge, 2010)
2. J. Geiser, in *Iterative Splitting Methods for Differential Equations*. Numerical Analysis and Scientific Computing Series, ed. by F. Magoules, C.-H. Lai (Chapman & Hall/CRC, Boca Raton, 2011)
3. R. Balescu, *Transport Processes in Plasma: Classical Transport*, vol. 1 (North Holland Publications, Amsterdam, 1988)
4. R. Balescu, *Transport Processes in Plasmas: Neoclassical Transport Theory*, vol. 2 (North Holland Publications, Amsterdam, 1988)
5. L. Rosso, A.F. de Baas, Review of Materials Modelling: What Makes a Material Function? Let Me Compute the Ways... European Commision, General for Research and Innovation Directorate, Industrial Technologies, Unit G3Materials (2014). http://ec.europa.eu/research/industrial_technologies/modelling-materials_en.html

# Appendix

In the following, we have some notes to the additional mathematical methods used in the previous sections.

## A.1 Computation of Multiple Stochastic Integrals

In Sect. 5.1, we have applied stochastic integrals in multiple form. Here are some ideas to solve them with numerical approximations.

We assume to compute

$$\int_0^1 W_j(s)dW_i(s) = \sum_{k=1}^{N} W_j\left(\frac{t_j + t_{j+1}}{2}\right)\Delta W_i, \tag{A.1}$$

$$\delta t = 1/N, t_{j+1} = \delta t + t_j, t_1 = 0, \tag{A.2}$$

where the intermediate values of $W_j$ are given with respect to the Brownian bridge:

$$W(t) = (1+t)B\left(\frac{t}{1+t}\right). \tag{A.3}$$

The Brownian bridge is presented as a Fourier series with stochastic coefficients, as

$$B_t = \sum_{k=1}^{\infty} Z_k \frac{\sqrt{2}\sin(k\pi t)}{k\pi}, \tag{A.4}$$

where $Z_1, Z_2, \ldots$ are independent identically distributed standard normal random variables (means $N(0, 1)$) (see the Karhunen Loeve theorem).



J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5

## A.2 Alternative Ideas for Solving the Nonlinear Multicomponent Equation

In the following, we present some alternative ideas to solve the nonlinear behaviour in the multicomponent equations:

- Iterative scheme in time with a finite volume scheme for the spatial operators: The benefit of the scheme is straightforward implementation while dealing with forward steps and linearization via time steps. A drawback is restriction with respect to the CFL condition.
- Closed form of the Stefan–Maxwell equation: One of the benefits is a closed form of the equations, i.e. we deal with only one equation which can be done explicitly or implicitly. The drawbacks are the mixed derivations in the equations and the delicate implicit scheme, which needs an additional nonlinear solver.

### *A.2.1 Iterative Scheme in Time (Finite Volume Scheme, Global Linearisation, Successive Method), Given in [1]*

We solve the iterative scheme

$$\left(\frac{1}{D_{13}} + \alpha\xi^n_{2,j-1/2}\right) N^n_{1,j} - \alpha\xi^n_{1,j-1/2} N^n_{2,j} = \frac{\xi^n_{1,j-1} - \xi^n_{1,j}}{\Delta x}, \tag{A.5}$$

$$-\beta\xi^n_{2,j-1/2} N^n_{1,j} + \left(\frac{1}{D_{23}} + \beta\xi^n_{1,j-1/2}\right) N^n_{2,j} = \frac{\xi^n_{2,j-1} - \xi^n_{2,j}}{\Delta x}, \tag{A.6}$$

for $n \in \mathbb{N}$ (time-index), $j$, $1 \le j \le J-1$.

$$\xi^{n+1}_{i,j} = \xi^n_{i,j} - \frac{\Delta t}{\Delta x^2}(N^n_{i,j+1}\,\Delta x - N^n_{i,j}\,\Delta x), \tag{A.7}$$

with $i = 1, 2$ and $n \in \mathbb{N}$ (time-index), $j$, $0 \le j \le J-1$.

We have further

$$\xi^n_{i,j-1/2} = \frac{1}{2}(\xi^n_{i,j} + \xi^n_{i,j-1}),\ 0 \le j \le J-1, \tag{A.8}$$

$$\xi^0_{i,j} = \xi^{\text{in}}_i(x_{j+1/2}), \tag{A.9}$$

$$N^n_{i,0} = 0,\ n \in \mathbb{N}, \tag{A.10}$$

$$N^n_{i,J} = 0,\ n \in \mathbb{N}. \tag{A.11}$$

Further, $D_{12} = D_{13} = 0.833$, $D_{23} = 0.168$ $J = 140$ with $\Delta x = 1/J$ and $\Delta t \le \frac{1}{2}\frac{\Delta x^2}{\max\{D_{12}, D_{13}, D_{23}\}}$.

**Algorithm A.1** (1) Initialization $n = 0$:

We compute for $j = 1, \ldots, J-1$:

$$\begin{pmatrix} N^0_{1,j} \\ N^0_{2,j} \end{pmatrix} = \begin{pmatrix} A^0_{11,j} & A^0_{12,j} \\ A^0_{21,j} & A^0_{22,j} \end{pmatrix}^{-1} \begin{pmatrix} \frac{\xi^0_{1,j-1} - \xi^0_{1,j}}{\Delta x} \\ \frac{\xi^0_{2,j-1} - \xi^0_{2,j}}{\Delta x} \end{pmatrix}, \tag{A.12}$$

where $\xi^0_{1,j} = \xi^{\text{in}}_1(j\Delta x)$, $\xi^0_{2,j} = \xi^{\text{in}}_2(j\Delta x)$, $j = 0, \ldots, J$ and given as for the different initializations, we have

1. Uphill example

$$\xi^{\text{in}}_1(x) = \begin{cases} 0.8 & \text{if } 0 \le x < 0.25 \\ 1.6(0.75 - x) & \text{if } 0.25 \le x < 0.75 \\ 0.0 & \text{if } 0.75 \le x \le 1.0 \end{cases}, \tag{A.13}$$

$$\xi^{\text{in}}_2(x) = 0.2, \text{ for all } x \in \Omega = [0, 1], \tag{A.14}$$

2. Diffusion example (asymptotic behaviour)

$$\xi^{\text{in}}_1(x) = \begin{cases} 0.8 \text{ if } 0 \le x \in 0.5, \\ 0.0 \text{ else}, \end{cases} \tag{A.15}$$

$$\xi^{\text{in}}_2(x) = 0.2, \text{ for all } x \in \Omega = [0, 1], \tag{A.16}$$

The matrix entries are given as

$$A^0_{11,j} = \frac{1}{D_{13}} + \alpha\xi^0_{2,j-1/2}, A^0_{12,j} = -\alpha\xi^0_{1,j-1/2}, \tag{A.17}$$

$$A^0_{21,j} = -\beta\xi^0_{2,j-1/2}, A^0_{22,j} = \frac{1}{D_{23}} + \beta\xi^0_{1,j-1/2}, \tag{A.18}$$

$$\xi^0_{1,j-1/2} = \frac{1}{2}(\xi^0_{1,j} + \xi^0_{1,j-1}), \xi^0_{2,j-1/2} = \frac{1}{2}(\xi^0_{2,j} + \xi^0_{2,j-1}). \tag{A.19}$$

Further, the values of the first and the last grid points of $N$ are zero, i.e. $N^0_{1,0} = N^0_{1,J} = N^0_{2,0} = N^0_{2,J} = 0$ (boundary condition).

(2) Next time steps (till $n = N_{\text{end}}$):

(2.1) Computation of $\xi^{n+1}_1$ and $\xi^{n+1}_2$ :

We compute for $j = 0, \ldots, J-1$:

$$\xi_{1,j}^{n+1} = \xi_{1,j}^{n} - \frac{\Delta t}{\Delta x^2}\left(N_{1,j+1}^{n}\Delta x - N_{1,j}^{n}\Delta x\right), \tag{A.20}$$

$$\xi_{2,j}^{n+1} = \xi_{2,j}^{n} - \frac{\Delta t}{\Delta x^2}\left(N_{2,j+1}^{n}\Delta x - N_{2,j}^{n}\Delta x\right), \tag{A.21}$$

where $N_{1,0}^{n} = N_{1,J}^{n} = N_{2,0}^{n} = N_{2,J}^{n} = 0$ (boundary condition).

(2.2) Computation of $N_1^{n+1}$ and $N_2^{n+1}$

We compute for $j = 1, \ldots, J-1$:

$$\begin{pmatrix} N_{1,j}^{n+1} \\ N_{2,j}^{n+1} \end{pmatrix} = \begin{pmatrix} A_{11,j}^{n+1} & A_{12,j}^{n+1} \\ A_{21,j}^{n+1} & A_{22,j}^{n+1} \end{pmatrix}^{-1} \begin{pmatrix} \frac{\xi_{1,j-1}^{n+1} - \xi_{1,j}^{n+1}}{\Delta x} \\ \frac{\xi_{2,j-1}^{n+1} - \xi_{2,j}^{n+1}}{\Delta x} \end{pmatrix}, \tag{A.22}$$

where the matrix entries are given as

$$A_{11,j}^{n+1} = \frac{1}{D_{13}} + \alpha\xi_{2,j-1/2}^{n+1},\ A_{12,j}^{n+1} = -\alpha\xi_{1,j-1/2}^{n+1}, \tag{A.23}$$

$$A_{21,j}^{n+1} = -\beta\xi_{2,j-1/2}^{n+1},\ A_{22,j}^{n+1} = \frac{1}{D_{23}} + \beta\xi_{1,j-1/2}^{n+1}, \tag{A.24}$$

$$\xi_{1,j-1/2}^{n+1} = \frac{1}{2}(\xi_{1,j}^{n+1} + \xi_{1,j-1}^{n+1}),\ \xi_{2,j-1/2}^{n+1} = \frac{1}{2}(\xi_{2,j}^{n+1} + \xi_{2,j-1}^{n+1}). \tag{A.25}$$

Further, the values of the first and the last grid points of $N$ are zero, i.e. $N_{1,0}^{n+1} = N_{1,J}^{n+1} = N_{2,0}^{n+1} = N_{2,J}^{n+1} = 0$ (boundary condition).

(3) Do $n = n + 1$ and goto (2)

### *A.2.2 Closed Form with Chain-Rule of the SM Equation*

We deal with the multicomponent problem given in Sect. 5.5 as

$$\partial_t \xi_1 + \partial_x \cdot N_1 = 0, \tag{A.26}$$

$$\partial_t \xi_2 + \partial_x \cdot N_2 = 0, \tag{A.27}$$

$$\begin{pmatrix} N_1 \\ N_2 \end{pmatrix} = \frac{D_{13}D_{23}}{1 + \alpha D_{13}\xi_2 + \beta D_{23}\xi_1} \begin{pmatrix} \frac{1}{D_{23}} + \beta\xi_1 & \alpha\xi_1 \\ \beta\xi_2 & \frac{1}{D_{13}} + \alpha\xi_2 \end{pmatrix} \begin{pmatrix} -\partial_x\xi_1 \\ -\partial_x\xi_2 \end{pmatrix}. \tag{A.28}$$

It is also possible to apply Eq. (A.28) directly into Eqs. (A.26) and (A.27). We apply the partial derivations with respect to the chain-rule and have the following equations:

$$\begin{pmatrix} \partial_x N_1 \\ \partial_x N_2 \end{pmatrix} = \partial_x \left( \frac{D_{13} D_{23}}{1 + \alpha D_{13}\xi_2 + \beta D_{23}\xi_1} \times \begin{pmatrix} \frac{1}{D_{23}} + \beta\xi_1 & \alpha\xi_1 \\ \beta\xi_2 & \frac{1}{D_{13}} + \alpha\xi_2 \end{pmatrix} \begin{pmatrix} -\partial_x \xi_1 \\ -\partial_x \xi_2 \end{pmatrix} \right). \tag{A.29}$$

Further, we have

$$N_1 = \frac{D_{13} D_{23}}{1 + \alpha D_{13}\xi_2 + \beta D_{23}\xi_1} \left( -\frac{1}{D_{23}} \partial_x \xi_1 - \beta\xi_1 \partial_x \xi_1 - \alpha\xi_1 \partial_x \xi_2 \right), \tag{A.30}$$

$$N_2 = \frac{D_{13} D_{23}}{1 + \alpha D_{13}\xi_2 + \beta D_{23}\xi_1} \left( \beta\xi_2 \partial_x \xi_1 - \frac{1}{D_{13}} \partial_x \xi_2 - \alpha\xi_2 \partial_x \xi_2 \right), \tag{A.31}$$

and the derivations with respect to $\partial_x$ and $\partial_y$ are given as

$$\begin{aligned} \partial_x N_1 = & \left( \frac{D_{13} D_{23}}{(1 + \alpha D_{13}\xi_2 + \beta D_{23}\xi_1)^2} \left( \frac{\alpha D_{13}}{D_{23}} \partial_x \xi_2 \partial_x \xi_1 \right.\right. \\ & \left.\left. + \beta \partial_x \xi_1 \partial_x \xi_1 + \alpha\beta D_{13} \partial_x \xi_2 \xi_1 \partial_x \xi_1 + \beta^2 D_{23} \partial_x \xi_1 \xi_1 \partial_x \xi_2 \right) \right) \\ & + \frac{D_{13} D_{23}}{1 + \alpha D_{13}\xi_2 + \beta D_{23}\xi_1} \left( -\frac{1}{D_{23}} \partial_x^2 \xi_1 - \beta \partial_x \xi_1 \partial_x \xi_1 - \beta\xi_1 \partial_x^2 \xi_1 \right. \\ & \left. - \alpha \partial_x \xi_1 \partial_x \xi_2 - \alpha\xi_1 \partial_x^2 \xi_2 \right), \end{aligned} \tag{A.32}$$

$$\begin{aligned} \partial_x N_2 = & \left( \frac{D_{13} D_{23}}{(1 + \alpha D_{13}\xi_2 + \beta D_{23}\xi_1)^2} \left( \frac{\beta D_{23}}{D_{13}} \partial_x \xi_1 \partial_x \xi_2 \right.\right. \\ & \left.\left. + \alpha \partial_x \xi_2 \partial_x \xi_2 + \alpha\beta D_{23} \partial_x \xi_1 \xi_2 \partial_x \xi_2 + \alpha^2 D_{13} \partial_x \xi_2 \xi_2 \partial_x \xi_2 \right) \right) \\ & + \frac{D_{13} D_{23}}{1 + \alpha D_{13}\xi_2 + \beta D_{23}\xi_1} \left( -\frac{1}{D_{13}} \partial_x^2 \xi_2 - \alpha \partial_x \xi_2 \partial_x \xi_2 - \alpha\xi_2 \partial_x^2 \xi_2 \right. \\ & \left. - \beta \partial_x \xi_2 \partial_x \xi_1 - \beta\xi_2 \partial_x^2 \xi_1 \right). \end{aligned} \tag{A.33}$$

Based on the full equation, we could apply the different time- and spatial-discretization schemes. We apply the finite-difference discretization in space, with the operators $D_+$ (forward difference operator) and $D_-$ (backward difference operator), see also Sect. 5.5.

For the time-discretization, we apply the explicit or implicit Euler-schemes, which are discussed in the following.

(1) The explicit form with the time-discretization is given as

$$\xi_1^{n+1} = \xi_1^n - \Delta t\ \mathscr{N}_1^n, \tag{A.34}$$

$$\xi_2^{n+1} = \xi_2^n - \Delta t\ \mathscr{N}_2^n, \tag{A.35}$$

$$\begin{aligned}\mathscr{N}_1^n = \Gamma_1^n \Big(&\frac{\alpha D_{13}}{D_{23}} D_+\xi_2^n D_-\xi_1^n + \beta D_+\xi_1^n D_+\xi_1^n \\ &+\alpha\beta D_{13} D_+\xi_2^n \Xi_1^n D_-\xi_1^n + \beta^2 D_{23} D_+\xi_1^n \Xi_1^n D_-\xi_2^n\Big) \\ +\Gamma_2^n \Big(&-\frac{1}{D_{23}} D_+ D_-\xi_1^n - \beta D_+\xi_1^n D_-\xi_1^n \\ &-\beta \Xi_1^n D_+ D_-\xi_1^n - \alpha D_+\xi_1^n D_-\xi_2^n - \alpha \Xi_1^n D_+ D_-\xi_2^n\Big),\end{aligned} \tag{A.36}$$

$$\begin{aligned}\mathscr{N}_2^n = \Gamma_1^n \Big(&\frac{\beta D_{23}}{D_{13}} D_+\xi_1^n D_-\xi_2^n + \alpha D_+\xi_2^n D_-\xi_2^n \\ &+\alpha\beta D_{23} D_+\xi_1^n \Xi_2^n D_-\xi_2^n + \alpha^2 D_{13} D_+\xi_2^n \Xi_2^n D_-\xi_1^n\Big) \\ +\Gamma_2^n \Big(&-\frac{1}{D_{13}} D_+ D_-\xi_2^n - \alpha D_+\xi_2^n D_-\xi_2^n \\ &-\alpha \Xi_2^n D_+ D_-\xi_2^n - \beta D_+\xi_2^n D_-\xi_1^n - \beta \Xi_2^n D_+ D_-\xi_1^n\Big),\end{aligned} \tag{A.37}$$

for $j = 1, \ldots, J$, where $\xi_1^n = (\xi_{1,1}^n, \ldots, \xi_{1,J}^n)^T$ and $\xi_2^n = (\xi_{2,1}^n, \ldots, \xi_{2,J}^n)^T$ and $I_J \in \mathbb{R}^J \times \mathbb{R}^J$ and the matrices are given as

$$\Xi_1^n, \Xi_2^n, \Gamma_1^n, \Gamma_2^n \in \mathbb{R}^J \times \mathbb{R}^J, \tag{A.38}$$

$$\Xi_{1,j,j}^n = \xi_{1,j}^n,\ \ j = 1 \ldots, J, \tag{A.39}$$

$$\Xi_{2,j,j}^n = \xi_{2,j}^n,\ \ j = 1 \ldots, J, \tag{A.40}$$

$$\Gamma_{1,j,j}^n = \frac{D_{13} D_{23}}{(1 + \alpha D_{13}\xi_{2,j}^n + \beta D_{23}\xi_{1,j}^n)^2},\ \ j = 1 \ldots, J, \tag{A.41}$$

$$\Gamma_{2,j,j}^n = \frac{D_{13} D_{23}}{1 + \alpha D_{13}\xi_{2,j}^n + \beta D_{23}\xi_{1,j}^n},\ \ j = 1 \ldots, J, \tag{A.42}$$

$$\Gamma_{1,i,j}^n = \Gamma_{2,i,j} = \Xi_{1,i,j}^n = \Xi_{2,i,j}^n = 0,\ \ i, j = 1 \ldots, J,\ \ i \neq J. \tag{A.43}$$

Further, $D_+$ and $D_-$ are the finite difference matrices, given in Sect. 5.5.

(2) The implicit form with the time-discretization is given as

$$\xi_1^{n+1} = \xi_1^n - \Delta t\, \mathcal{N}_1^{n+1}, \tag{A.44}$$

$$\xi_2^{n+1} = \xi_2^n - \Delta t\, \mathcal{N}_2^{n+1}, \tag{A.45}$$

$$\begin{aligned}\mathcal{N}_1^{n+1} = \Gamma_1^{n+1}\Big(&\frac{\alpha D_{13}}{D_{23}} D_+\xi_2^{n+1} D_-\xi_1^{n+1} + \beta D_+\xi_1^{n+1} D_+\xi_1^{n+1} \\ &+\alpha\beta D_{13} D_+\xi_2^{n+1}\Xi_1^{n+1} D_-\xi_1^{n+1} + \beta^2 D_{23} D_+\xi_1^{n+1}\Xi_1^{n+1} D_-\xi_2^{n+1}\Big) \\ +\Gamma_2^{n+1}\Big(&-\frac{1}{D_{23}} D_+D_-\xi_1^{n+1} - \beta D_+\xi_1^{n+1} D_-\xi_1^{n+1} - \beta\Xi_1^{n+1} D_+D_-\xi_1^{n+1} \\ &-\alpha D_+\xi_1^{n+1} D_-\xi_2^{n+1} - \alpha\Xi_1^{n+1} D_+D_-\xi_2^{n+1}\Big),\end{aligned} \tag{A.46}$$

$$\begin{aligned}\mathcal{N}_2^{n+1} = \Gamma_1^{n+1}\Big(&\frac{\beta D_{23}}{D_{13}} D_+\xi_1^{n+1} D_-\xi_2^{n+1} + \alpha D_+\xi_2^{n+1} D_-\xi_2^{n+1} \\ &+\alpha\beta D_{23} D_+\xi_1^{n+1}\Xi_2^{n+1} D_-\xi_2^{n+1} + \alpha^2 D_{13} D_+\xi_2^{n+1}\Xi_2^{n+1} D_-\xi_1^{n+1}\Big) \\ +\Gamma_2^{n+1}\Big(&-\frac{1}{D_{13}} D_+D_-\xi_2^{n+1} - \alpha D_+\xi_2^{n+1} D_-\xi_2^{n+1} \\ &-\alpha\Xi_2^{n+1} D_+D_-\xi_2^{n+1} - \beta D_+\xi_2^{n+1} D_-\xi_1^{n+1} - \beta\Xi_2^{n+1} D_+D_-\xi_1^{n+1}\Big),\end{aligned} \tag{A.47}$$

for $j = 1, \ldots, J$, where $\xi_1^{n+1} = (\xi_{1,1}^{n+1}, \ldots, \xi_{1,J}^{n+1})^T$ and $\xi_2^{n+1} = (\xi_{2,1}^{n+1}, \ldots, \xi_{2,J}^{n+1})^T$ and $I_J \in \mathbb{R}^J \times \mathbb{R}^J$ and the matrices are given as

$$\Xi_1^{n+1}, \Xi_2^{n+1}, \Gamma_1^{n+1}, \Gamma_2^{n+1} \in \mathbb{R}^J \times \mathbb{R}^J, \tag{A.48}$$

$$\Xi_{1,j,j}^{n+1} = \xi_{1,j}^{n+1},\ j = 1 \ldots, J, \tag{A.49}$$

$$\Xi_{2,j,j}^{n+1} = \xi_{2,j}^{n+1},\ j = 1 \ldots, J, \tag{A.50}$$

$$\Gamma_{1,j,j}^{n+1} = \frac{D_{13} D_{23}}{(1 + \alpha D_{13}\xi_{2,j}^{n+1} + \beta D_{23}\xi_{1,j}^{n+1})^2},\ j = 1 \ldots, J, \tag{A.51}$$

$$\Gamma_{2,j,j}^{n+1} = \frac{D_{13} D_{23}}{1 + \alpha D_{13}\xi_{2,j}^{n+1} + \beta D_{23}\xi_{1,j}^{n+1}},\ j = 1 \ldots, J, \tag{A.52}$$

$$\Gamma_{1,i,j}^{n+1} = \Gamma_{2,i,j} = \Xi_{1,i,j}^{n+1} = \Xi_{2,i,j}^{n+1} = 0,\ i, j = 1 \ldots, J,\ i \neq J. \tag{A.53}$$

Further, $D_+$ and $D_-$ are the finite difference matrices, given in Sect. 5.5.

*Remark A.1* The explicit scheme is simpler to implement but has the drawback of the CFL condition, i.e. we are restricted in the time step. The implicit scheme is more

delicate, while we have to deal with a coupled nonlinear equation system which can be solved by Newton's method and fixpoint schemes, see [1]. We have the benefit of a more flexible time step, while we do not have a time restriction.

## A.3 Detail Separation of the Underlying Jacobian Matrix in the Mathematical Notation

In the following, we discuss the separation of the matrix in the mathematical notation.
We have the following idea:

- Decomposition in upper and lower matrix part without eigenvalue separation;
- Decomposition in upper and lower matrix part with eigenvalue separation.

### *A.3.1 Decomposition in Upper and Lower Matrix Part (Without Eigenvalue Separation)*

**Jacobi Matrix upper part of glycolysis.**
Only present in this module
$$\begin{aligned}
&(\text{Glycx1}'(t):=\text{v161}-\text{v162}-\text{v17})\\
&(\text{Glc1}'(t):=\text{v021}-\text{v022}-\text{v03})\\
&(\text{G6P1}'(t):=\text{v03}-\text{v041}+\text{v042}-\text{v22})\\
&(\text{F6P1}'(t):=\text{v041}-\text{v042}-\text{v05})\\
&(\text{FBP1}'(t):=\text{v05}-\text{v061}+\text{v062})\\
&(\text{GAP1}'(t):=\text{v061}-\text{v062}+\text{v071}-\text{v072}-\text{v081}+\text{v082})\\
&(\text{DHAP1}'(t):=\text{v061}-\text{v062}-\text{v071}+\text{v072}-\text{v15})\\
&(\text{Glyc1}'(t):=\text{v15}-\text{v161}+\text{v162})\\
&(\text{Glcx1}'(t):=\text{v011}-\text{v012}-\text{v021}+\text{v022})
\end{aligned}$$
Common species, present in both modules
$$\begin{aligned}
&(\text{BPG1}'(t):=\text{v081}-\text{v082})\\
&(\text{ATP1}'(t):=-\text{v03}-\text{v05}-\text{v22}-\text{v241}+\text{v242})\\
&(\text{ADP1}'(t):=\text{v03}+\text{v05}+\text{v22}+\text{v23}+2\text{v241}-2\text{v242})\\
&(\text{AMP1}'(t):=\text{v242}-\text{v241})\\
&(\text{NADH1}'(t):=\text{v081}-\text{v082}-\text{v15})\\
&(\text{NAD1}'(t):=-\text{v081}+\text{v082}+\text{v15})\\
&(\text{P1}'(t):=-\text{v081}+\text{v082}+\text{v15}+2\text{v22}+\text{v23})
\end{aligned}$$
Not present in this module
$$\begin{aligned}
&(\text{PEP1}'(t):=0)\\
&(\text{Pyr1}'(t):=0)\\
&(\text{ACA1}'(t):=0)\\
&(\text{EtOH1}'(t):=0)
\end{aligned}$$

$(\mathrm{EtOHx1}'(t){:=}0)$
$(\mathrm{ACAx1}'(t){:=}0)$
$(\mathrm{CNx1}'(t){:=}0)$

**Jacobi Matrix lower part of glycolysis.**
Only present in this module $(\mathrm{PEP2}'(t){:=}\mathrm{v091}-\mathrm{v092}-\mathrm{v10})$
$(\mathrm{Pyr2}'(t){:=}\mathrm{v10}-\mathrm{v11})$
$(\mathrm{ACA2}'(t){:=}\mathrm{v11}-\mathrm{v12}-\mathrm{v181}+\mathrm{v182})$
$(\mathrm{EtOH2}'(t){:=}\mathrm{v12}-\mathrm{v131}+\mathrm{v132})$
$(\mathrm{EtOHx2}'(t){:=}\mathrm{v131}-\mathrm{v132}-\mathrm{v14})$
$(\mathrm{ACAx2}'(t){:=}\mathrm{v181}-\mathrm{v182}-\mathrm{v19}-\mathrm{v20})$
$(\mathrm{CNx2}'(t){:=}-\mathrm{v20}+\mathrm{v211}-\mathrm{v212})$
Common species, present in both modules $(\mathrm{BPG2}'(t){:=}\mathrm{v092}-\mathrm{v091})$
$(\mathrm{ATP2}'(t){:=}\mathrm{v091}-\mathrm{v092}+\mathrm{v10}-\mathrm{v23})$
$(\mathrm{ADP2}'(t){:=}-\mathrm{v091}+\mathrm{v092}-\mathrm{v10}+\mathrm{v23})$
$(\mathrm{NADH2}'(t){:=}-\mathrm{v12})$
$(\mathrm{NAD2}'(t){:=}+\mathrm{v12})$
$(\mathrm{p2}'(t){:=}\mathrm{v23})$
Not present in this module $(\mathrm{Glycx2}'(t){:=}0)$
$(\mathrm{Glc2}'(t){:=}0)$
$(\mathrm{G6P2}'(t){:=}0)$
$(\mathrm{F6P2}'(t){:=}0)$
$(\mathrm{FBP2}'(t){:=}0)$
$(\mathrm{GAP2}'(t){:=}0)$
$(\mathrm{DHAP2}'(t){:=}0)$
$(\mathrm{Glyc2}'(t){:=}0)$
$(\mathrm{Glcx2}'(t){:=}0)$

### *A.3.2 Decomposition in Upper and Lower Matrix Part (with Eigenvalue Separation)*

**$\mathrm{Glcx1}'(t)$ $\mathrm{Glc1}'(t)$ $\mathrm{G6P1}'(t)$ $\mathrm{F6P1}'(t)$ $\mathrm{FBP1}'(t)$ $\mathrm{GAP1}'(t)$ $\mathrm{DHAP1}'(t)$ $\mathrm{BPG1}'(t)$ $\mathrm{PEP1}'(t)$ $\mathrm{Pyr1}'(t)$ $\mathrm{ACA1}'(t)$ $\mathrm{EtOH1}'(t)$ $\mathrm{EtOHx1}'(t)$ $\mathrm{Glyc1}'(t)$ $\mathrm{Glycx1}'(t)$ $\mathrm{ACAx1}'(t)$ $\mathrm{CNx1}'(t)$ $\mathrm{ATP1}'(t)$ $\mathrm{ADP1}'(t)$ $\mathrm{AMP1}'(t)$ $\mathrm{NADH1}'(t)$ $\mathrm{NAD1}'(t)$ $\mathrm{P1}'(t)$**
**Eigenvalues**

$$c-725830$$
$$-3402.31$$
$$-1873.06$$
$$-1288.99$$

$$
\begin{aligned}
&-620.438\\
&-174.753\\
&-97.0176\\
&-75.8312\\
&-54.2505\\
&-34.9159\\
&-0.481297 + 9.91003i\\
&-0.481297 - 9.91003i\\
&-5.68561\\
&-5.65028\\
&-3.02005\\
&-2.27007 + 0.447624i\\
&-2.27007 - 0.447624i\\
&-1.35615\\
&-0.946389\\
&-0.194037\\
&-1.2047299556964155 \times 10^{-12}\\
&1.7737943007470836 \times 10^{-15}\\
&0.
\end{aligned}
$$

We have the error estimates given for the parallel and sequential splitting, where $||\cdot||$ is the maximum absolute row sum of the matrix:
$||J_1J_2 + J_2J_1||_{\max} = \max_{1\le j\le n} \sum_{i=i}^{m} |a_{i,j}| = 5.85252 \times 10^6$
$||J_1J_2 - J_2J_1||_{\max} = \max_{1\le j\le n} \sum_{i=i}^{m} |a_{i,j}| = 5.85252 \times 10^6$
The time step is given as
$\tau = 1.26438 \times 10^{-7}$.

## Reference

1. L. Boudin, B. Grec, F. Salvarani, A mathematical and numerical analysis of the Maxwell-Stefan diffusion equation. Discret. Contin. Dyn. Syst. Ser. B **17**(5), 1427–1440 (2012)

# Glossary

In the following, we explain the notation and nomenclature that we used in our book.
**Notation** We deal with the following notation in our monograph.

| | |
|---|---|
| $D(B)$ | Domain of $B$ |
| $\mathbf{X}, \mathbf{X}_E$ | Banach spaces |
| $\mathbf{X}^n = \Pi_{i=1}^n \mathbf{X}_i$ | Product space of $\mathbf{X}$ |
| $W^{m,p}(\Omega)$ | Sobolev space consisting of all locally summable functions $u : \Omega \to \mathbb{R}$ such that for each multi-index $\alpha$ with $\|\alpha\| \leq m$, $\partial_\alpha u$ exists in the weak sense and belongs to $L^p(\Omega)$ |
| $\partial\Omega$ | Boundary of $\Omega$ |
| $\mathscr{L}(\mathbf{X}) = L(\mathbf{X}, \mathbf{X})$ | Operator space of $\mathbf{X}$, e.g. a Banach space |
| $\Omega_h$ | Discretized domain $\Omega$ with the underlying grid step $h$ |
| $H^m$ | Sobolev space $W^{m,2}$ |
| $H_0^1(\Omega)$ | The closure of $C_c^\infty(\Omega)$ in the Sobolev space $W^{1,2}$ |
| $\|\cdot\|_{L^p}$ | $L^p$-norm |
| $\|\cdot\|_{H^m}$ | $H^m$-norm |
| $\|\cdot\|$ | Maximum norm, if not defined otherwise |
| $\|\cdot\|_{\mathbf{X}}$ | Norm with respect to Banach space $\mathbf{X}$ |
| $\|\cdot\|_\infty = \sup_{t\in I} \|\cdot\|$ | Maximum norm on interval $I$ |
| $(x, y)$ | Scalar product of $x$ and $y$ in a Hilbert space |
| $\mathscr{O}(\tau)$ | Landau symbol, e.g. first order in time with time step $\tau$ |
| $U = (u, v)^T$ | Vectorial solutions of two components |
| $U = (u, v, w)^T$ | Vectorial solutions of three components |
| $(x_1, \ldots, x_n)^T = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ | Vectorial solutions of $n$ components |



J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5

**Notation in the Models** In the following, we describe the notation that are used in our modelling problems.

| | |
|---|---|
| $R_i$: | Retardation factor $[-]$, which declares the portion of the porosities of the underlying aquifer, |
| $u_i$: | Mobile concentration of the $i$th species, e.g., transported species $Si$, $Ti$, $C$ in the plasma [mol/mm$^3$], |
| $g_i$: | Immobile concentration of the ith species, e.g., absorbed species $Si$, $Ti$, $C$ in the plasma [mol/mm$^3$], |
| $\mathbf{v}$: | Velocity of the underlying fluid e.g., direction and absolute value of the plasma flux in the apparatus [mm/s], |
| $D$: | Diffusion–dispersion tensor e.g., molecular and dispersive value of the plasma diffusion [mm$^2$/s], |
| $\lambda_i$: | Decay constant of the ith species e.g., decay rates of the transported species in the plasma [1/s], |
| $e_i(t)$, $\tilde{e}_i(t)$, $f_i(t)$: | Are the time-dependent convection and reaction terms, which are polynomials and $e_i(t)$, $f_i(t) : \mathbb{R}^+ \to \mathbb{R}^+,\ i = 1, \ldots, m$, |
| $i = 1, \ldots, M$: | $i$ denotes the species and $M$ denotes the number of species, |
| $\beta$ : | The exchange between the mobile and immobile part of the aquifer |

# Bibliography


1. C.T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*. SIAM Frontiers in Applied Mathematics, vol. 16 (SIAM, Philadelphia, 1995)
2. M. Abramowitz, I.A. Stegun, *Handbook of Mathematical Functions* (Dover Publications, New York, 1970)
3. N.I. Akhiezer, I.M. Glazman, *Theory of Linear Operators in Hilbert Space* (Dover Publications, New York, 1993)
4. I. Alonso-Mallo, B. Cano, J.C. Jorge, Spectral-fractional step Runge-Kutta discretisations for initial boundary value problems with time dependent boundary conditions. Math. Comput. **73**, 1801–1825 (2004)
5. Z.S. Alterman, A. Rotenberg, Seismic waves in a quarter plane. Bull. Seismol. Soc. Am. **59**, 347–368 (1969)
6. G. Ariel, B. Engquist, R. Tsai, A multiscale method for highly oscillatory ordinary differential equations with resonance. Math. Comput. **78**, 929–956 (2009)
7. G. Ariel, B. Engquist, S. Kim, Y. Lee, R. Tsai, A multiscale method for highly oscillatory dynamical systems using a Poincare map type technique. J. Sci. Comput. **54**(2–3), 247–268 (2013)
8. U.M. Ascher, S.J. Ruuth, R.J. Spiteri, Implicit-explicit Runge-Kutta methods for time-dependent partial differential equations. Appl. Numer. Math. **25**, 151–167 (1997)
9. O. Axelsson, *Iterative Solution Methods* (Cambridge University Press, Cambridge, 1996)
10. W. Balser, J. Mozo-Fernandez, Multisummability of formal solutions of singular perturbation problems. J. Differ. Equ. **183**(2), 526–545 (2002)
11. W. Balser, A. Duval, S. Malek, Summability of formal solutions for abstract Cauchy problems and related convolution equations. Manuscript, November 2006
12. V. Barbu, *Nonlinear Semigroups and Differential Equations in Banach Spaces* (Editura Academiei, Bucuresti and Noordhoff, Leyden, 1976)
13. D.A. Barry, C.T. Miller, P.J. Culligan-Hensley, Temporal discretization errors in non-iterative split-operator approaches to solving chemical reaction/groundwater transport models. J. Contam. Hydrol. **22**, 1–17 (1996)
14. P. Bastian, *Parallele adaptive Mehrgitterverfahren*. Doktor-Arbeit (Universität Heidelberg, 1994)
15. R.M. Beam, R.F. Warming, Alternating direction implicit methods for parabolic equations with a mixed derivative. SIAM J. Sci. Stat. Comput. **1**, 131–159 (1980)
16. R.E. Bellman, *Dynamic Programming* (Princeton University Press, Princeton, 1957)
17. J.P. Berenger, A perfectly matched layer for the absorption of electromagnetic waves. J. Comput. Phys. **111**, 185–220 (2005)




J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5

18. M. Bjorhus, Operator splitting for abstract Cauchy problems. IMA J. Numer. Anal. **18**, 419–443 (1998)
19. S. Blanes, F. Casas, J. Ros, Extrapolation of symplectic integrators. Celest. Mech. Dyn. Astron. **75**, 149–161 (1999)
20. C.K. Birdsall, A.B. Langdon, *Plasma Physics via Computer Simulation*. Series in Plasma Physics (Taylor & Francis, New York, 1985)
21. C.K. Birdsall, Particle in cell charged-particle simulations, Plus Monte Carlo collisions with neutral atoms, PIC-MCC. IEEE Trans. Plasma Sci. **19**(2), 65–85 (1991)
22. A.R. Bozorgmanesh, M. Otadi, A.A. Safe Kordi, F. Zabihi, M. Barkhordari Ahmadi, Lagrange two-dimensional interpolation method for modeling nanoparticle formation during RESS process. Int. J. Ind. Math. **1**(2), 175–181 (2009)
23. A. Brandt, Multi-level adaptive solutions to boundary-value problems. Math. Comput. **31**(138), 333–390 (1977)
24. M.D. Buhmann, *Radial Basis Functions: Theory and Implementations* (Cambridge University Press, Cambridge, 2003)
25. M.D. Buhmann, C.A. Micchelli, On radial basis approximation on periodic grids. University of Cambridge, Department of Applied Mathematics and Theoretical Physics (1991)
26. D. Buhmann, *Das Programmpaket EMOS. Ein Instrumentarium zur Analyse der Langzeitsicherheit von Endlagern*. Gesellschaft für Anlagen- und Reaktorsicherheit (mbH), GRS-159 (Braunschweig, 1999)
27. J.C. Butcher, Implicit Runge-Kutta processes. Math. Comput. **18**, 50–64 (1964)
28. J.C. Butcher, *Numerical Methods for Ordinary Differential Equations* (Wiley, Chichester, 2003)
29. C.X. Cao, A theorem on the separation of a system of coupled differential equations. J. Phys. A: Math. Gen. **14**, 1069–1074 (1981)
30. G.Z. Cao, H. Brinkman, J. Meijerink, K.J. DeVries, A.J. Burggraaf, Kinetic study of the modified chemical vapour deposition process in porous media. J. Mater. Chem. **3**(12), 1307–1311 (1993)
31. Y. Cao, D.T. Gillespie, L.R. Petzold, The slow-scale stochastic simulation algorithm. J. Chem. Phys. **122**(014116), 1–34 (2005)
32. F. Casas, A. Murua, M. Nadinic, Efficient computation of the Zassenhaus formula. Comput. Phys. Commun. **183**(11), 2386–2391 (2012)
33. M.A. Celia, J.S. Kindred, I. Herrera, Contaminant transport and biodegradation 1. a numerical model for reactive transport in porous media. Water Resour. Res. **25**, 1141–1148 (1989)
34. D. Chandler, *Introduction to Modern Statistical Mechanics* (Oxford University Press, Oxford, 1987)
35. A.K. Chaniotis, D. Poulikakos, High order interpolation and dierentiation using B-splines. J. Comput. Phys. **197**, 253–274 (2004)
36. Q.-S. Chen, H. Zhang, V. Prasad, C.M. Balkas, N.K. Yushin, Modeling of heat transfer and kinetics of physical vapor transport growth of silicon carbide crystals. Trans. ASME J. Heat Transf. **123**(6), 1098–1109 (2001)
37. K.L. Chien, J.A. Hrones, J.B. Reswick, On the automatic tuning of generalized passive systems. Trans. ASME **74**, 175–185 (1952)
38. S.A. Chin, A fundamental theorem on the structure of symplectic integrators. Phys. Lett. A **354**, 373–376 (2006)
39. S.A. Chin, C.R. Chen, Gradient symplectic algorithms for solving the Schrödinger equation with time-dependent potentials. J. Chem. Phys. **117**(4), 1409–1415 (2002)
40. W. Cheney, *Analysis for Applied Mathematics*. Graduate Texts in Mathematics, vol. 208 (Springer, New York, 2001)
41. P.D. Christofides, *Nonlinear and Robust Control of PDE Systems: Methods and Applications to Transport-Reaction Processes*. Systems & Control: Foundations & Applications (Birkhäuser, Boston, 2001)
42. A.J. Chorin, J.E. Marsden, *A Mathematical Introduction to Fluid Mechanics*. Texts in Applied Mathematics, 3rd edn. (Springer, Heidelberg, 1993)

43. D.J. Christie, Target material pathways model for high power pulsed magnetron sputtering. J. Vac. Sci. Technol. **23**(2), 330–335 (2005)
44. C.K. Chui, K. Jetter, J.D. Ward, Cardinal interpolation by multivariate splines. Math. Comput. **48**, 711–724 (1987)
45. P.G. Ciarlet, J.L. Lions (eds.), *Finite Difference Methods (Part 1)*. Handbook of Numerical Analysis, vol. I (North-Holland/Elsevier, Amsterdam, 1990)
46. P.C. Clemmow, J.P. Dougherty, *Electrodynamics of Particles and Plasmas* (Addison-Wesley, Redwood City, 1969)
47. N. Clisby, B. McCoy, Ninth and tenth order virial coefficients for hard spheres in D dimensions. J. Stat. Phys. **122**(1), 15–57 (2006)
48. K.H. Coats, B.D. Smith, Dead-end pore volume and dispersion in porous media. Soc. Pet. Eng. J. **4**(3), 73–84 (1964)
49. G.C. Cohen, *Higher-Order Numerical Methods for Transient Wave Equations* (Springer, Heidelberg, 2002)
50. Comsol Multiphysics. Comsol Multiphysics application. Online software (2010). http://www.comsol.de/
51. COMSOL. COMSOL, Multiphysics, Software-Package (2014). http://www.comsol.com/COMSOL. Burlington, USA
52. N. Crouseilles, M. Mehrenberger, E. Sonnendrücker, Conservative semi-Lagrangian schemes for the Vlasov equation. J. Comput. Phys. **229**, 1927–1953 (2010)
53. B. Davis, *Integral Transform and Their Applications*. Applied Mathematical Sciences, vol. 25 (Springer, New York, 1978)
54. S.M. Day et al., Test of 3D elastodynamic codes: final report for lifelines project 1A01. Technical report, Pacific Earthquake Engineering Center (2001)
55. S.M. Day et al., Test of 3D elastodynamic codes: final report for lifelines project 1A02. Technical report, Pacific Earthquake Engineering Center (2003)
56. A.M. Dimits, W.W. Lee, Partially linearized algorithms in Gyrokinetic particle simulation. J. Comput. Phys. **107**(2), 309323 (1993)
57. A.M. Dimits, B.I. Cohen, R.E. Caflisch, L. Ricketson, M.S. Rosin, Higher-order and Multilevel time integration of stochastic differential equations and application to Coulomb collisions. Lecture at the Workshop III: Mathematical and Computer Science Approaches to High Energy Density Physics, 7–11 May 2012, IPAM, UCLA, USA (2012)
58. C.R. Dohrmann, A. Klawonn, O.B. Widlund, Domain decomposition for less regular subdomains: overlapping Schwarz in two dimensions. SIAM J. Numer. Anal. **46**, 2153–2168 (2008)
59. J. Douglas Jr., S. Kim, Improved accuracy for locally one-dimensional methods for parabolic equations. Math. Models Methods Appl. Sci. **11**, 1563–1579 (2001)
60. F. Dupret, P. Nicodéme, Y. Ryckmans, P. Wouters, M.J. Crochet, Global modelling of heat transfer in crystal growth furnaces. Int. J. Heat Mass Transf. **33**(9), 1849–1871 (1990)
61. J. Duras, Instabilities in ion thrusters by plasma-wall interactions. Diploma Thesis, Brandenburgische Universität Cottbus, Germany (2011)
62. M.K. Dobkin, D.M. Zuraw, *Principles of Chemical Vapor Deposition*, 1st edn. (Springer, Heidelberg, 2003)
63. D.R. Durran, *Numerical Methods for Wave Equations in Geophysical Fluid Dynamics* (Springer, New York, 1998)
64. E.G. D'Yakonov, Difference schemes with splitting operator for multi-dimensional nonstationary problems. Zh. Vychisl. Mat. i. Mat. Fiz. **2**, 549–568 (1962)
65. E. Weinan, B. Engquist, Multiscale modelling and computations. Not. AMS **50**(9), 1062–1070 (2003)
66. G. Eason, J. Fulton, I.N. Sneddon, The generation of waves in an infinite elastic solid by variable body forces. Philos. Trans. R. Soc. Lond. **248**, 575–607 (1956)
67. P. Eklund, M. Beckers, J. Frodelius, H. Högberg, L. Hultman, Magnetron sputtering of Ti3SiC2 tin films from a compound target. JVST A **25**(5), 1381–1388 (2007)

68. P. Eklund, Multifunctional nanostructured Ti-Si-C thin films. Linköping studies in science and technology. Dissertation No. 1087 (2007)
69. P. Eklund, A. Murugaiah, J. Emmerlich, Z. Czigany, J. Frodelius, M.W. Barsoum, H. Högberg, L. Hultman, Homoepitaxial growth of Ti-Si-C MAX-phase thin films on bulk Ti3SiC2 substrates. J. Cryst. Growth **304**, 264–269 (2007)
70. K.-J. Engel, R. Nagel, *One-Parameter Semigroups for Linear Evolution Equations* (Springer, New York, 2000)
71. L.C. Evans, *Partial Differential Equations*. Graduate Studies in Mathematics, vol. 19 (AMS, Providence, 1998)
72. L. Evans, *Partial Differential Equations*. Graduate Studies in Mathematics (AMS, Providence, 2010)
73. G.R. Eykolt, Analytical solution for networks of irreversible first-order reactions. Water Res. **33**(3), 814–826 (1999)
74. G.R. Eykolt, L. Li, Fate and transport of species in a linear reaction network with different retardation coefficents. J. Contam. Hydrol. **46**, 163–185 (2000)
75. R. Eymard, T. Galluoët, R. Herbin, *Finite Volume Methods. Handbook of Numerical Analysis*, vol. 7 (North Holland, Amsterdam, 2000), pp. 713–1020
76. G. Fairweather, A.R. Mitchell, A high accuracy alternating direction method for the wave equations. J. Ind. Math. Appl. **1**, 309–316 (1965)
77. I.T. Famelis, F. Xanthos, G. Papageorgiou, Numerical solution of stochastic differential equations with additive noise by RungeKutta methods. J. Numer. Anal. Ind. Appl. Math. **4**(3–4), 171–180 (2009)
78. I. Farago, A. Havasi, On the convergence and local splitting error of different splitting schemes. Eötvös Lorand University, Budapest (2004)
79. I. Farago, *Splitting Methods for Abstract Cauchy Problems*. Lecture Notes in Computer Science, vol. 3401 (Springer, Berlin, 2005), pp. 35–45
80. I. Farago, Modified iterated operator splitting method. Appl. Math. Model. **32**(8), 1542–1551 (2008)
81. I. Farago, A. Havasi, Consistency analysis of operator splitting methods for $C_0$-semigroups. Semigroup Forum **74**, 125–139 (2007)
82. R. Fazio, A. Jannelli, Second order positive schemes by means of flux limiters for the advection equation. IANG Int. J. Appl. Math. **39**(1), 25–35 (2009)
83. E. Fein, A. Schneider, $d^3f$-Ein Programmpaket zur Modellierung von Dichteströmungen. Abschlussbericht, Braunschweig (1999)
84. E. Fein, T. Kühle, U. Noseck, Entwicklung eines Programms zur dreidimensionalen Modellierung des Schadstofftransportes. Fachliches Feinkonzept, Braunschweig (2001)
85. E. Fein, Beispieldaten für radioaktiven Zerfall. Private communications, Braunschweig (2000)
86. E. Fein, Physikalisches Modell und mathematische Beschreibung. Private communications, Braunschweig (2001)
87. C.A.J. Fletcher, *Computational Techniques for Fluid Dynamics*. Series in Computational Physics, 2nd edn. (Springer, Berlin, Heidelberg New York, 1997)
88. P. Frolkovič, J. Geiser, Numerical Simulation of Radionuclides Transport in Double Porosity Media with Sorption, in *Proceedings of Algoritmy 2000, Conference of Scientific Computing* (2000), pp. 28–36
89. M.J. Gander, H. Zhao, Overlapping Schwarz waveform relaxation for parabolic problems in higher dimension, ed. by A. Handlovičová, M. Komorníkova, K. Mikula *Proceedings of Algoritmy 14*, Slovak Technical University (1997), pp. 42–51
90. S.D. Gedney, An anisotropic perfectly matched layer-absorbing medium for the truncation of FDTD lattices. IEEE Trans. Antennas Propag. **44**(12), 1630–1639 (1996)
91. J. Geiser, Numerical simulation of a model for transport and reaction of radionuclides, in *Proceedings of the Third International Conference on Large-Scale Scientific Computing*. Lecture Notes in Computer Science, vol. 2179 (Sozopol, Bulgaria, 2001), pp. 487–496
92. J. Geiser, Gekoppelte Diskretisierungsverfahren für Systeme von Konvektions-Dispersions-Diffusions-Reaktionsgleichungen. Ph.D. thesis, University of Heidelberg, Germany (2004)

93. J. Geiser, $R^3T$ : Radioactive-Retardation-Reaction-Transport-Program for the Simulation of radioactive waste disposals. Technical report, ISC-04-03-MATH, Institute for Scientific Computation, Texas A & M University, College Station, TX (2004)
94. J. Geiser, Discretisation methods with embedded analytical solutions for convection dominated transport in porous media, in *Proceedings of the 3rd International Conference, NAA 2004*. Lecture Notes in Mathematics, vol. 3401 (Springer, Rousse, 2005), pp. 288–295
95. J. Geiser, R.E. Ewing, J. Liu, Operator splitting methods for transport equations with nonlinear reactions, in *Proceedings of the Third MIT Conference on Computational Fluid and Solid Mechanic*. Cambridge, MA, 14–17 June 2005
96. J. Geiser, J. Gedicke. Nonlinear iterative operator-splitting methods and applications for nonlinear parabolic partial differential equations. Preprint No. 2006-17 of Humboldt University of Berlin, Department of Mathematics, Germany
97. J. Geiser, Discretization methods with analytical solutions for convection-diffusion-dispersion-reaction-equations and application. J. Eng. Math. **57**, 79–98 (2007)
98. J. Geiser, Weighted iterative operator-splitting methods: stability-theory, in *Proceedings of the 6th International Conference, NMA 2006*. Lecture Notes in Computer Science, vol. 4310 (Springer, Borovets, 2007), pp. 40–47
99. J. Geiser, C. Kravvaritis, Weighted iterative operator-splitting methods and applications, in *Proceedings of the 6th International Conference, NMA 2006*. Lecture Notes in Computer Science, vol. 4310 (Springer, Borovets, 2007), pp. 48–55
100. J. Geiser, S. Nilsson, A Fourth Order Split Scheme for Elastic Wave Propagation. Preprint 2007–08, Humboldt University of Berlin, Department of Mathematics, Germany (2007)
101. J. Geiser, L. Noack, Iterative operator-splitting methods for wave equations with stability results and numerical examples. Preprint 2007-10, Humboldt-University of Berlin (2007)
102. J. Geiser, V. Schlosshauer, Operator-splitting methods for wave-equations. Preprint 2007-06, Humboldt University of Berlin, Department of Mathematics, Germany (2007)
103. J. Geiser, S. Sun, *Multiscale Discontinuous Galerkin Methods for Modeling Flow and Transport in Porous Media*. Lecture Notes in Computational Science, vol. 4487 (Springer, New York, 2007), pp. 890–897
104. J. Geiser, Operator splitting methods for wave equations. Int. Math. Forum, Hikari Ltd. **2**(43), 2141–2160 (2007)
105. J. Geiser, L. Noack, Iterative operator-splitting methods for nonlinear differential equations and applications of deposition processes. Preprint 2008-04, Humboldt-University of Berlin (2008)
106. J. Geiser, L. Noack, Operator-splitting methods respecting eigenvalue problems for nonlinear equations and application in Burgers-equations. Preprint 2008-13, Humboldt-University of Berlin (2008)
107. J. Geiser, Fourth-order splitting methods for time-dependent differential equations. Numer. Math.: Theory Methods Appl. **1**(3), 321–339 (2008)
108. J. Geiser, M. Arab, Modelling, Optimization and Simulation for a Chemical Vapor Deposition. J. Porous Media, Begell House Inc., Redding, USA, **2**(9), 847–867 (2009)
109. J. Geiser, C. Kravvaritis, Overlapping operator splitting methods and applications in stiff differential equations. Special issue: Novel Difference and Hyprod Methods for Differential and Integro-Differential Equations and Applications, Guest editors: Qin Sheng and Johnny Henderson, Neural, Parallel, and Scientific Computations (NPSC), vol. 16 (2008), pp. 189–200
110. J. Geiser, *Discretization and Simulation of Systems for Convection-Diffusion-Dispersion Reactions with Applications in Groundwater Contamination*, Monograph, Series: Groundwater Modelling, Management and Contamination (Nova Science Publishers, Inc., New York, 2008)
111. J. Geiser, Stability of iterative operator-splitting methdods. Int. J. Comput. Math. 1029–0265, First published on 26 June 2009. http://www.informaworld.com
112. J. Geiser, Computation of iterative operator-splitting methods. Preprint 2009–21, Humboldt University of Berlin, Department of Mathematics, Germany (2009)

113. J. Geiser, C. Kravvaritis, A domain decomposition method based on iterative operator splitting method. Appl. Numer. Math. **59**, 608–623 (2009)
114. J. Geiser, Iterative operator-splitting with time overlapping algorithms: theory and application to constant and time-dependent wave equations, in *Wave Propagation in Materials for Modern Applications*, ed. by A. Petrin. ISBN: 978-953-7619-65-7, INTECH (2009)
115. J. Geiser, Operator-splitting methods in respect of eigenvalue problems for nonlinear equations and applications to Burgers equations. J. Comput. Appl. Math. Elsevier, Amsterdam, North Holland **231**(2), 815–827 (2009)
116. J. Geiser, R. Steijl. Coupled Navier Stokes—molecular dynamics simulation using iterative operator-splitting methods. Preprint 2009-11, Humboldt University of Berlin, Department of Mathematics, Germany (2009)
117. J. Geiser, M. Arab, Modelling, Optimization and Simulation for a Chemical Vapor Deposition. J. Porous Media, Begell House Inc., Redding, USA, **12**(9), 847–867 (2009)
118. J. Geiser, F. Krien, Iterative operator-splitting methods for time-irreversible systems: theory and application to advection-diffusion equations. Preprint 2009–18, Humboldt University of Berlin, Department of Mathematics, Germany (2009)
119. J. Geiser, S. Blankenburg, Monte Carlo simulations concerning elastic scattering with application to DC and high power pulsed magnetron sputtering for $Ti_3SiC_2$. Preprint 2009–20, Humboldt University of Berlin, Department of Mathematics, Germany (2009)
120. J. Geiser, C. Fleck, Adaptive Step-size Control in Simulation of Diffusive CVD Processes. Mathematical Problems in Engineering, vol. 2009, Art. ID 728105. Hindawi Publishing Corporation, New York, USA (2009) 34 p
121. J. Geiser, R. Röhle, Kinetic processes and phase-transition of CVD processes for $Ti_3SiC_2$. JCIT: J. Converg. Inf. Technol. **5**(6), 9–32 (2010)
122. J. Geiser, Iterative operator-splitting methods for nonlinear differential equations and applications. NMPDE, published online, March 2010
123. J. Geiser, Mobile and immobile fluid transport: coupling framework. Int. J. Numer. Methods Fluids, accepted as Review October 2009. Online published http://www3.interscience.wiley.com/cgi-bin/fulltext/123276563/PDFSTART (2010)
124. J. Geiser, Consistency of iterative operator-splitting methods: theory and applications. Numer. Methods Part. Differ. Equ. **26**(1), 135–158 (2010)
125. J. Geiser, V. Buck, M. Arab, Model of PE-CVD apparatus: verification and simulations. Math. Probl. Eng. **2010**, Article ID 407561 (2010)
126. J. Geiser, M. Arab, Simulation of a chemical vapor deposition: mobile and immobile zones and homogeneous layers. Spec. Top. Rev. Porous Media, Begell House Inc., Redding, USA, **1**(2), 123–143 (2010)
127. J. Geiser, M. Arab, Porous media based modeling of PE-CVD apparatus: electrical fields and deposition geometries. Spec. Top. Rev. Porous Media, Begell House Inc., Redding, USA, **1**(3), 215–229 (2010)
128. J. Geiser, Magnus integrator and successive approximation for solving time-dependent problems. Preprint 2010–10, Humboldt University of Berlin, Department of Mathematics, Germany (2010)
129. J. Geiser, *Decomposition Methods in Multiphysics and Multiscale Problems*. Physics Research and Technology (Nova Science Publishers, Inc., New York, 2010). (Monograph)
130. J. Geiser, M. Elbiomy, Splitting method of convection-diffusion methods with disentanglement methods. Preprint 2010-2, Humboldt University of Berlin, Department of Mathematics, Germany (2010)
131. J. Geiser, G. Tanoglu, Operator-splitting methods via Zassenhaus product formula. Appl. Math. Comput. **217**, 4557–4575 (2011)
132. J. Geiser, G. Tanoglu, N. Guecueyenen, Higher order operator-splitting methods via Zassenhaus product formula: theory and applications. Comput. Math. Appl., Elsevier, North Holland, **62**(4), 1994–2015 (2011)
133. J. Geiser, T. Zacher, Time-dependent fluid transport: coupling framework. Preprint 2011-5, Humboldt University of Berlin, Department of Mathematics, Germany (2011)

134. J. Geiser, R. Calov, Operator-splitting methods respecting eigenvalue problems for shallow shelf equations with basal drag. Coupled Syst. Mech., Techno-Press, Yuseong-gu Daejeon, Korea **1**(4), 325–343 (2012)
135. J. Geiser, Splitting approach to coupled Navier Stokes and molecular dynamics simulation. J. Comput. Model. (JCoMod), Commun. Math. Appl. Scienpress Ltd, UK, **2**(2), 1–34 (2012)
136. J. Geiser, *Multiscale Methods for Levitron Problems: Theory and Applications*. Comput. Struct. **122**, 27–32 (2012) (Elsevier, North Holland)
137. J. Geiser, Operator splitting methods combined with multi-grid methods. J. Mod. Math. Front. **1**(2), 1–10 (2012)
138. J. Geiser, Simulation of a Heat Transfer in Porous Media (2012). Preprint, arXiv:1205.2449
139. J. Geiser, M. Arab, Simulation of a chemical vapor deposition: four phase model. Spec. Top. Rev. Porous Media, Begell House Inc., Redding, USA, **3**(1):55–68 (2012)
140. J. Geiser, Modelling and simulation of transport problems with mathematical splitting techniques. Cumulative Habilitation Thesis, University of Bochum, Germany (2013)
141. J. Geiser, Multiscale methods for Levitron problems: theory and applications. Comput. Struct. **122**, 27–32 (2013)
142. J. Geiser, Multiscale modeling of PE-CVD apparatus: simulations and approximations. Polymers **5**, 142–160 (2013)
143. J. Geiser, An iterative splitting approach for linear integro-differential equations. Appl. Math. Lett. Elsevier, Amsterdam, The Netherlands, **26**(11), 1048–1052 (2013)
144. J. Geiser, Iterative splitting methods for multiscale problems. Distrib. Comput. Appl. Bus. Eng. Sci. (DCABES), 2–4 September 2013, London, UK, 3–6 (2013)
145. J. Geiser, Embedded Zassenhaus expansion to splitting schemes: theory and multiphysics applications. Int. J. Differ. Equ., Hindawi Publishing Corporation, New York, USA, August 2013
146. J. Geiser, Multiscale splitting method for the Boltzmann-Poisson equation: application to the dynamics of electrons. Int. J. Differ. Equ. **2014**, Article ID 178625 (2014), 8 p
147. J. Geiser, M. Beauregard, An extrapolated splitting method for solving semidiscretized parabolic differential equations. Int. J. Comput. Math., Taylor and Francis, accepted March 2014
148. P. George, *Chemical Vapor Deposition: Simulation and Optimization*, 1st edn. (VDM Verlag Dr. Müller, Saarbrücken, 2008)
149. P. George, P.T. Lin, H.C. Gea, Y. Jaluria, Reliability-based optimisation of chemical vapour deposition process. Int. J. Reliab. Saf. **3**(4), 363–383 (2009)
150. M.E. Glicksman, *Diffusion in Solids: Field Theory, Solid-State Principles, and Applications* (Wiley, New York, 2000)
151. W.B. Gragg, The pade table and its relation to certain algorithms of numerical analysis. SIAM Rev. **14**(1), 1–62 (1972)
152. GRAPE. GRAphics Programming Environment for Mathematical Problems, Version 5.4. Institut für Angewandte Mathematik, Universität Bonn und Institut für Angewandte Mathematik, Universität Freiburg (2001)
153. C. Grossmann, H.-G. Ross, Numerik partieller Differentialgleichungen. Teubner Studienbücher, Mathematik (1994)
154. W. Hackbusch, *Multi-Grid Methods and Applications* (Springer, Berlin, 1985)
155. W. Hackbusch, *Iterative Solution of Large Sparse Systems of Equations*. Applied Mathematical Sciences (Springer, Berlin, 1994)
156. J. Hadamard, Sur les problemes aux derivees partielles et leur signification physique. Princeton University Bulletin, 4952 (1902)
157. F. Haefner, D. Sames, H.-D. Voigt, *Heat and Mass Transfer* (Springer, Berlin, 1992)
158. E. Hairer, S.P. Norsett, G. Wanner, *Solving Ordinary Differential Equations I*. SCM, vol. 8 (Springer, Berlin, 1992)
159. A. Harten, High resolution schemes for hyperbolic conservation laws. J. Comput. Phys. **135**(2), 260–278 (1997)
160. A. Havasi, J. Bartholy, I. Farago, Splitting method and its application in air pollution modeling. Q. J. Hung. Meteorol. Serv. **105**(1), 39–58 (2001)

161. D. Henry, *Geometric Theory of Semilinear Parabolic Equations*. Lecture Notes in Mathematics (Springer, Berlin, 1981)
162. J. Herzer, W. Kinzelbach, Coupling of transport and chemical processes in numerical transport models. Geoderma **44**, 115–127 (1989)
163. K. Higashi, T. Pigford, Analytical models for migration of radionuclides in geologic sorbing media. J. Nucl. Sci. Technol. **17**(9), 700–709 (1980)
164. T. Hirono, W. Lui, S. Seki, Y. Yoshikuni, A three-dimensional fourth-order finite-difference time-domain scheme using a symplectic integrator propagator. IEEE Trans. Microw. Theory Tech. **49**(9), 1640–1648 (2001)
165. V. Hlavacek, J. Thiart, D. Orlicki, Morphology and film growth in CVD reactions. J. Phys. IV Fr. **5**, 3–44 (1995)
166. M. Hochbruck, C. Lubich, Exponential integrators for large systems of differential equations. SIAM J. Sci. Comput. **19**(5), 1552–1574 (1998)
167. M. Hochbruck, C. Lubich, On Krylov subspace approximations to the matrix exponential operator. SIAM J. Numer. Anal. **34**(5), 1911–1925 (1997)
168. A.P.W. Hodder, Geothermal waters: a source of energy and metals. New Zealand Institute of Chemistry, published online (2005). http://nzic.org.nz/ChemProcesses/water/13A.pdf
169. I. Holod, Z. Lin, Statistical analysis of fluctuations and noise-driven transport in particle-in-cell simulations of plasma turbulence. Phys. Plasmas **14**, 032306 (2007)
170. T.Y. Hou, D. Liang, Multiscale analysis for convection dominated transport equations. Discret. Contin. Dyn. Syst. **23**(1–2), 281–298 (2009)
171. W. Hundsdorfer, L. Portero, A note on iterated splitting schemes. J. Comput. Appl. Math. **201**(1), 146–152 (2007)
172. E. Huenges, *Geothermal Energy Systems: Exploration, Development, and Utilization* (Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim, 2010)
173. Y. Igitkhanov, Modelling of multi-component plasma for TOKES. KIT Scientific Report Nr. 7564, Institut für Hochleistungsimpuls- und Mikrowellentechnik (IHM), KIT Scientific Publishing, Karlsruhe (2011)
174. W. Hundsdorfer, S.J. Ruuth, IMEX extensions of linear multistep methods with general monotonicity and boundedness properties. J. Comput. Phys. **225**(2), 2016–2042 (2007)
175. S. Jiang, L. Wang, J. Hong, Stochastic multi-symplectic integrator for stochastic nonlinear Schrodinger equation. Commun. Comput. Phys. (CiCP) **14**(2), 393–411 (2013)
176. H.A. Jakobsen, *Chemical Reactor Modeling: Multiphase Reactive Flows*, 1st edn. (Springer, Heidelberg, 2008)
177. J.D. Jackson, *Classical Electrodynamics*, 3rd edn. (Wiley, New York, 1999)
178. T.B. Jones, M. Washizu, R.F. Gans, Simple theory for the Levitron. J. Appl. Phys. **82**, 883–888 (1997)
179. J. Janssen, S. Vandewalle, Multigrid waveform relaxation on spatial finite-element meshes: the continuous case. SIAM J. Numer. Anal. **33**, 456–474 (1996)
180. Y.L. Jiang, Periodic waveform relaxation solutions of nonlinear dynamic equations. Appl. Math. Comput. **135**(2–3), 219–226 (2003)
181. K. Johannsen, Robuste Mehrgitterverfahren für die Konvektions-Diffusions Gleichung mit wirbelbehafteter Konvektion. Ph.D. thesis, University of Heidelberg, Germany (1999)
182. K. Johannsen, An aligned 3D-finite-volume method for convection-diffusion problems, in *Modeling and Computation in Environmental Sciences*, vol. 59, ed. by R. Helmig, W. Jäger, W. Kinzelbach, P. Knabner, G. Wittum (Vieweg, Braunschweig, 1997), pp. 227–243
183. S.L. Johnson, Y. Saad, M. Schultz, Alternating direction methods on multiprocessors. SIAM J. Sci. Stat. Comput. **8**(5), 686–700 (1987)
184. E.J. Kansa, Multiquadrics—a scattered data approximation scheme with applications to computational fluid dynamics I: surface approximations and partial derivative estimates. Comput. Math. Appl. **19**(8/9), 127–145 (1990)
185. E.J. Kansa, Multiquadrics—a scattered data approximation scheme with applications to computational fluid dynamics II: solutions to parabolic, hyperbolic, and elliptic partial differential equations. Comput. Math. Appl. **19**(8/9), 147–161 (1990)

186. E.J. Kansa, R.E. Carlson, Improved accuracy of multiquadric interpolation using variable shape parameters. Comput. Math. Appl. **24**, 99–120 (1992)
187. E.J. Kansa, Y.C. Hon, Circumventing the ill-conditioning problem with multiquadric radial basis functions: applications to elliptic partial differential equations. Comput. Math. Appl. **39**, 123–137 (2000)
188. E.J. Kansa, J. Geiser, Numerical solution to time-dependent 4D inviscid Burgers' equations. Eng. Anal. Bound. Elem. **37**, 637–645 (2013)
189. S. Karaa, High-order compact ADI methods for parabolic equations. J. Comput. Math. Appl. **52**(8–9), 1343–1356 (2006)
190. S. Karaa, High-order difference schemes for 2-d elliptic and parabolic problems with mixed derivatives. Wiley InterSciences **23**(2), 366–378 (2007)
191. K.H. Karlsen, K.-A. Lie, J.R. Natvig, H.F. Nordhaug, H.K. Dahle, Operator splitting methods for systems of convection-diffusion equations: nonlinear error mechanisms and correction strategies. J. Comput. Phys. **173**(2), 636–663 (2001)
192. C.T. Kelley, *Solving Nonlinear Equations with Newton's Method*. Computational Mathematics, vol. XIV (SIAM, Philadelphia, 2003)
193. G.M. Kepler, H.T. Tran, H.T. Banks, Reduced order model compensator control of species transport in a CVD reactor. Optim. Control Appl. Methods **21**, 143–160 (1999)
194. R. Kettler, Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods, in *Multigrid Methods*. Lecture Notes in Mathematics, vol. 960, ed. by W. Hackbusch, U. Trottenberg (Springer, Berlin, 1981), pp. 33–53
195. S. Kim, H. Lim, High-order schemes for acoustic waveform simulation. Appl. Numer. Math. **57**(4), 402–414 (2007)
196. W. Kinzelbach, Numerische Methoden zur Modellierung des Transports von Schadstoffen im Grundwasser. Schriftenreihe Wasser-Abwasser, Oldenburg (1992)
197. R. Kozlov, B. Owren, Order reduction in operator splitting methods. Preprint N6-1999, Department of Mathematical Sciences, Norwegian University of Science and Technology, Trondheim, Norway (1999)
198. R. Kozlov, A. Kvarno, B. Owren, The behaviour of the local error in splitting methods applied to stiff problems. J. Comput. Phys. **195**, 576–593 (2004)
199. R. Lafore, *Object-Oriented Programming in C++*, 4th edn. (Sams Publishing, Indianapolis, 2001)
200. A.B. Langdon, B.I. Cohen, A. Friedman, Direct implicit large time-step particle simulation of plasmas. J. Comput. Phys. **51**(1), 107138 (1983)
201. C. Lange, M.W. Barsoum, P. Schaaf, Towards the synthesis of MAX-phase functional coatings by pulsed laser deposition. Appl. Surf. Sci. **254**, 1232–1235 (2007)
202. D. Lanser, J.G. Verwer, Analysis of operator splitting for advection-diffusion-reaction problems from air pollution modelling. J. Comput. Appl. Math. **111**(1–2), 201–216 (1999)
203. G. Lapenta, Automatic adaptive multi-dimensional particle in cell, in *Advanced Methods for Space Simulations*, ed. by H. Usui, Y. Omura (2007), pp. 61–76
204. L. Lapidus, G.F. Pinder, *Numerical Solution of Partial Differential Equations in Science and Engineering* (Wiley/Incorporation, Hoboken, 1996)
205. M. Laroussi, X. Lu, Room-temperature atmospheric pressure plasma plume for biomedical applications. Appl. Phys. Lett. **87**(11), 113902 (2005)
206. M. Laroussi, T. Akan, Arc-free atmospheric pressure cold plasma jets: a review. Plasma Process. Polym. **4**(9), 777–788 (2007)
207. E. Larrson, B. Fornberg, Theoretical and computational aspects of multivariate interpolation with increasing flat radial basis functions. Comput. Math. Appl. **49**(1), 103–130 (2005)
208. P.D. Lax, *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves* (SIAM, Philadelphia, 1973)
209. M. Lees, Alternating direction methods for hyperbolic differential equations. J. Soc. Ind. Appl. Math. **10**(4), 610–616 (1962)
210. H.H. Lee, *Fundamentals of Microelectronics Processing* (McGraw-Hill, New York, 1990)

211. J. Lee, T.F. Edgar, Continuation method for the modified Ziegler-Nichols tuning of multiloop control systems. Ind. Eng. Chem. Res. **44**(19), 7428–7434 (2005)
212. E. Lelarasmee, A. Ruehli, A. Sangiovanni-Vincentelli, The waveform relaxation methods for time domain analysis of large scale integrated circuits. IEEE Trans. CAD IC Syst. **1**, 131–145 (1982)
213. R.J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations, Steady State and Time Dependent Problems*. (Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2007)
214. R.W. Lewis, P. Bettess, E. Hinton, *Numerical Methods in Coupled Systems*. Wiley Series in Numerical Methods in Engineering (Wiley, New York, 1984)
215. N.A. Libre, A. Emdadi, E.J. Kansa, M. Shekarchi, A multiresolution prewavelet-based adaptive refinement scheme for RBF approximations of nearly singular problems. Eng. Anal. Bound. Elem. **33**, 901–914 (2009)
216. P. Lindelöf, Sur l'application des methodes d'approximations successives a l'etude de certaines equations differentielles ordinaires. J. de Math. Pures et Appl. **4**(9), 217–271 (1893)
217. P. Lindelöf, Sur l'application des methodes d'approximations successives a l'etude des integrales reelles des equations differentielles ordinaires. J. de Math. Pures et Appl. **4**(10), 117–128 (1894)
218. K. Lipnikov, M. Shashkov, D. Svyatskiy, The mimetic finite difference discretization of diffusion problem on unstructured polyhedral meshes. J. Comput. Phys. **211**, 473491 (2006)
219. C. Lubich, A variational splitting integrator for quantum molecular dynamics. Appl. Numer. Math. **48**(3–4), 355–368 (2004)
220. H. Lutz, W. Wendt, Taschenbuch der Regelungstechnik. Harri-Deutsch Verlag, Issue 6, Frankfurt, Germany (2005)
221. T.E. Magin, G. Degrez, Transport algorithms for partially ionized and unmagnetized plasmas. J. Comput. Phys. **198**(2), 424–449 (2004)
222. Maple Software. The essential tool for mathematics and modelling. Maplesoft, Waterloo Maple Inc. (2013). http://www.maplesoft.com/products/maple/
223. G.I. Marchuk, Splitting and alternating direction methods, in *Handbook of Numerical Analysis*, vol. 1, ed. by P.G. Ciarlet, J.L. Lions (Elsevier Science Publishers, B. V., North-Holland, 1990)
224. K. Matyash, R. Schneider, R. Sydora, F. Taccogna, Application of a grid-free kinetic model to the collisionless sheath. Contributions to Plasma Physics, in *11th International Workshop on Plasma Edge Theory in Fusion Devices*, vol. 48(1–3) (2008), pp. 116–120
225. S. McKinley, M. Levine, Cubic Spline Interpolation. Published online (1998). http://online.redwoods.cc.ca.us/instruct/darnold/laproj/fall98/skymeg/proj.pdf
226. I.A. Melamies, Atmospheric-pressure plasma in medical technology. Adapted version of German original: MedPLAST (2008), pp. 38–40
227. R.V.N. Melnik, *Mathematical and Computational Models for Transport and Coupled Processes in Micro- and Nanotechnology*, ed. by R.V.N. Melnik, A. Povitsky, D. Srivastava. Special Issue of J. Nanosci. Nanotechnol. **8**(7) (2008)
228. E. Messerschmid, S. Fasoulas, *Raumfahrtsysteme: Eine Einführung mit Übungen und Lösungen*, 3rd edn. (Springer, Berlin, 2009)
229. S. Middleman, A.K. Hochberg, *Process Engineering Analysis in Semiconductor Device Fabrication* (McGraw-Hill, New York, 1993)
230. U. Miekkala, O. Nevanlinna, Convergence of dynamic iteration methods for initial value problems. SIAM J. Sci. Stat. Comput. **8**(4), 459–482 (1987)
231. C. Moler, The Origins of MATLAB. Cleve's Corner (in the MathWorks Newsletter) (2004)
232. C. Moler, The Growth of MATLAB and MathWorks over Two Decades. Cleve's Corner (in The MathWorks Newsletter) (2006)
233. T. Moritaka, M. Nunami, H. Usui, Development of full particle-in-cell simulation code with adaptive mesh refinement technique. J. Plasma Fusion Res. Ser. **9**, 586 (2010)
234. N. Morosoff, *Plasma Deposition, Treatment and Etching of Polymers*, 1st edn., ed. by R. d'Agostino (Academic Press, Boston, 1990)

235. F. Neri, Lie algebras and canonical integration. Technical report. University of Arizona, Department of Physics (1987), 25 p
236. N. Neuss, A new sparse matrix storage method for adaptive solving of large systems of reaction-diffusion-transport equations, in *Scientific Computing in Chemical Engineering II*, ed. by F. Keil, et al. (Springer, Berlin, 1999), pp. 175–182
237. O. Nevanlinna, Remarks on Picard-Lindelöf iteration, part II. BIT **29**, 535–562 (1989)
238. O. Nevanlinna, Linear acceleration of Picard-Lindelöf. Numer. Math. **57**, 147–156 (1990)
239. X.B. Nie, S.Y. Chen, E. Weinan, M.O. Robbins, A continuum and molecular dynamics hybrid method for micro- and nano-fluid flow. J. Fluid Mech. **500**, 55–64 (2004)
240. NIST, Chemical Kinetic Database. Source for kinetic rate (2013). http://kinetics.nist.gov/kinetics
241. J. Nolen, Partial Differential Equations and Diffusion Processes. Lecture Notes, Department of Mathematics, Stanford University (2009)
242. OFELI. An Object Finite Element Library (2003). http://ofeli.sourceforge.net/
243. M. Ohlberg, A posteriori error estimates for vertex centered finite volume approximations of convection-diffusion-reaction equations. Preprints 12/2000, Mathematische Fakultät, Freiburg, May 2000
244. M. Ohring, *Materials Science of Thin Films*, 2nd edn. (Academic Press, San Diego, 2002)
245. A.M. Ostrowski, *Collected Mathematical Papers: Determinants, Linear Algebra, Algebraic Equations* (Birkhäuser Verlag, Basel, 1983)
246. J.A. Oteo, The Baker-Campbell-Hausdorff formula and nested commutator identities. J. Math. Phys. **32**, 419 (1991)
247. A. Pazy, *Semigroups of Linear Operators and Applications to Partial Differential Equations*, vol. 44, Applied Mathematical Sciences (Springer, Berlin, 1983)
248. A.D. Polyanin, V.F. Zaitsev, *Handbook of Nonlinear Partial Differential Equations* (Chapman & Hall/CRC Press, Boca Raton, 2004)
249. J. Prüss, Maximal regularity for evolution equations in Lp-spaces. Conf. Sem. Mat. Univ. Bari **285**, 139 (2003)
250. C. Quesne, Disentangling q-exponentials: a general approach. Int. J. Theor. Phys. **43**, 545–559 (2004)
251. S. Reuter, K. Niemi, V. Schulz-von der Gathen, H.F. Döbele, Generation of atomic oxygen in the effluent of an atmospheric pressure plasma jet. Plasma Sources Sci. Technol. **18**, 015006 (2009)
252. G.F. Roach, *Green's Functions* (Cambridge University Press, Cambridge, 1970)
253. H. Rouch, M. Pons, A. Benezech, J.N. Barbier, C. Bernard, R. Madar, Modelling of CVD reactors: thermochemical and mass transport approaches for $Si_{1-x}Ge_x$ deposition. J. Phys. IV **3**, 17–23 (1993)
254. H. Rouch, MOCVD Research Reactor Simulation, in *Proceedings of the COMSOL Users Conference*, Paris, France (2006)
255. L. Rudniak, Numerical simulation of chemical vapour deposition process in electric field. Comput. Chem. Eng. **22**(7), 755–758 (1998)
256. M. Rumpf, A. Wierse, GRAPE, Eine interaktive Umgebung für Visualisierung und Numerik. Informatik, Forschung und Entwicklung (1990)
257. Y. Saad, Analysis of some Krylov subspace approximation to the matrix exponential operator. SIAM J. Numer. Anal. **29**(1), 209–228 (1992)
258. Y. Saad, *Iterative Methods for Sparse Linear Systems*, 2nd edn. (SIAM Publications, Philadelphia, 2003)
259. H. Schmidt, P. Buchner, A. Datz, K. Dennerlein, S. Lang, M. Waidhas, Low-cost air-cooled PEFC stacks. J. Power Sources **105**, 243–249 (2002)
260. R. Schneider, Plasma-wall interaction: a multiscale problem. Phys. Scr. **T126**, 76–79 (2006)
261. D. Scholz, M. Weyrauch, A note on the Zassenhaus product formula. J. Math. Phys. **47**, 033505 (2006)
262. D. Scholz, V.G. Voronov, M. Weyrauch, Disentangling exponential operators. Preprint No. 2009-19, Georg-August Universität Göttingen, Institute of Numerical and Applied Mathematics (2009)

263. W. Sha, X. Wu, M. Chen, Z. Huang, Application of the higher-order symplectic FDTD scheme to the curved three-dimensional perfectly conducting objects. Microw. Opt. Technol. Lett. **49**(4), 931–934 (2007)
264. Q. Sheng, Solving linear partial differential equations by exponential splitting. IMA J. Numer. Anal. **9**, 199–212 (1989)
265. Q. Sheng, Global error estimates for exponential splitting. IMA J. Numer. Anal. **14**(1), 27–56 (1994)
266. Q. Sheng, R. Agarwal, A note on asymptotic splitting and its applications. Math. Comput. Model. **20**(12), 45–58 (1994)
267. A. Sidi, *Practical Extrapolation Methods: Theory and Applications*. Cambridge Monographs on Applied and Computational Mathematics, vol. 10 (Cambridge University Press, Cambridge, 2003)
268. M.D. Simon, L.O. Helfinger, S.L. Ridgway, Spin stabilized magnetic levitation. Am. J. Phys. **65**(4), 286–292 (1997)
269. B. Smith, P. Bjorstad, W. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations* (Cambridge University Press, Cambridge, 2004)
270. M.O. Steinhauser, *Multiscale Modeling of Fluids and Solids—Theory and Applications* (Springer, Berlin, 2009)
271. J. Stoer, R. Bulirsch, *Introduction to Numerical Analysis* (Springer, Berlin, 2001)
272. Y. Sun, J. Petersen, T. Clement, Analytical solutions for multiple species reactive transport in multiple dimensions. J. Contam. Hydrol. **35**, 429–440 (1999)
273. M. Suzuki, On the convergence of exponential operators—the Zassenhaus formula, BCH formula and systematic approximants. Commun. Math. Phys. **57**, 193–200 (1977)
274. M. Suzuki, General theory of fractal path-integrals with applications to many-body theories and statistical physics. J. Math. Phys. **32**(2), 400–407 (1991)
275. F. Taccogna, R. Schneider, S. Longo, M. Capitelli, Kinetic simulations of a plasma thruster. Plasma Sources Sci. Technol. **17**, 024003 (2008)
276. Y. Tanaka, Two-temperature chemically non-equilibrium modelling of high-power Ar-$N_2$ inductively coupled plasmas at atmospheric pressure. J. Phys. D: Appl. Phys. **37**, 1190–1205 (2004)
277. C. Theodoropoulos, Y.-H. Qian, I.G. Kevrekidis, Coarse stability and bifurcation analysis using time-steppers: a reaction-diffusion example. Proc. Natl. Acad. Sci. **97**(18), 9840–9843 (2000)
278. A.-K. Tornberg, B. Engquist, Numerical approximations of singular source terms in differential equations. J. Comput. Phys. **200**, 462–488 (2004)
279. J.P. Trelles, Computational study of flow dynamics from a dc arc plasma jet. J. Phys. D: Appl. Phys. **46**(25), 255201 (2013)
280. F. Tröltzsch, *Optimal Control of Partial Differential Equations: Theory, Methods, and Applications* (American Mathematical Society, Providence, 2010)
281. H.F. Trotter, On the product of semi-groups of operators. Proc. Am. Math. Soc. **10**(4), 545–551 (1959)
282. A.M.P. Valli, G.F. Carey, A.L.G.A. Coutinho, Control strategies for timestep selection in simulation of coupled viscous flow and heat transfer. Commun. Numer. Methods Eng. **18**(2), 131–139 (2002)
283. E. Vanden-Eijnden, Tutorial: problems with multiple time-scales: theoretical and computational aspects, in *Multiscale Modeling and Simulation of Complex Fluids, CXF07 Workshop*, (University of Maryland, 2007)
284. V.S. Varadarajan, *Lie Groups, Lie Algebras, and Their Representation*. Graduate Texts in Mathematics (Springer, Berlin, 1984)
285. A.E.P. Veldman, K. Rinzema, Playing with nonuniform grids. J. Eng. Math. **26**(1), 119–130 (1992)
286. J.G. Verwer, B. Sportisse, A note on operator splitting in a stiff linear case. MAS-R9830, ISSN 1386-3703 (1998)

287. J. Waldén, On the approximation of singular source terms in differential equations. Numer. Methods Part. Differ. Equ. **15**, 503–520 (1999)
288. H.S. Wall, *Analytic Theory of Continued Fractions* (Chelsea Publishing Company, New York, 1973), pp. 335–361
289. Website: http://portal.mytum.de/studium/studiengaenge/computational_science_and_engineering_master. TU München (2011)
290. E.W. Weisstein, *CRC Concise Encyclopedia of Mathematics* (CRC Press, Boca Raton, 1998)
291. K.D. Weltmann, E. Kindel, T. von Woedtke, M. Hähnel, M. Stieber, R. Brandenburg, Atmospheric-pressure plasma sources: prospective tools for plasma medicine. Pure Appl. Chem. **82**(6), 1223–1237 (2010)
292. J. Wertz, E.J. Kansa, L. Ling, The role of the multiquadric shape parameter in solving elliptic partial differential equations. Comput. Math. Appl. **51**(8), 1335–1348 (2006)
293. Wikipedia Reference: http://en.wikipedia.org/wiki/Multiphysics. Wikipedia, March 2011
294. Wikipedia Reference: http://en.wikipedia.org/wiki/Multiscale_modeling Wikipedia, March 2011
295. Wikipedia Reference: http://de.wikipedia.org/wiki/Computational_Engineering_Science. Wikipedia, March 2011
296. S.F. Wojtkiewicz, L.A. Bergman, Numerical solution of high-dimensional Fokker-Planck equations, in *8th ASCE Speciality Conference on Probabilistic Mechanics and Structural Reliability, PMC2000-167* (2000)
297. T. Yamaguchi, K. Shimizu, Asymptotic stabilization by PID control: stability analysis based on minimum phase and high-gain feedback. Electr. Eng. Jpn. **156**(1), 783–791 (2006)
298. H. Yoshida, Construction of higher order symplectic integrators. Phys. Lett. A **150**(5–7), 262–268 (1990)
299. K. Yoshida, *Functional Analysis*. Classics in Mathematics (Springer, Berlin, 1980)
300. H. Yserentant, On the multi-level splitting of finite element spaces. Numerische Mathematik **49**(4), 379–412 (1986)
301. A. Zagaris, H.G. Kaper, T.J. Kaper, Analysis of the computational singular perturbation reduction method for chemical kinetics. J. Nonlinear Sci. **14**, 59–91 (2004)
302. Y. Zeng, C. Tian, J. Liu, Convection-diffusion derived gradient films on porous substrates and their microstructural characteristics. J. Mater. Sci. **42**(7), 2387–2392 (2007)
303. N.B. Nichols, J.G. Ziegler, Optimum settings for automatic controllers. Trans. ASME **64**, 759–768 (1942)

# Index





J. Geiser, *Multicomponent and Multiscale Systems*,
DOI 10.1007/978-3-319-15117-5