INAUGURAL – DISSERTATION
zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich-Mathematischen Gesamtfakultät
der
Ruprecht-Karls-Universität
Heidelberg

vorgelegt von
Anja Bettendorf, Master of Science
aus Geizenburg bei Trier

Tag der mündlichen Prüfung: 24. Oktober 2019

Optimum experimental design for parameter estimation with 2D partial differential equation models

Betreuer:
Prof. Dr. Dr. h.c. mult. Hans Georg Bock
Prof. Dr. Guido Kanschat

Abstract

In this thesis, we investigate optimization problems with partial differential equation (PDE) constraints. In particular we are concerned with the efficient numerical solution of optimum experimental design (OED) problems for parameter estimation (PE) with PDE models, among them sampling design problems. We consider two dimensional (2D) stationary diffusion advection reaction PDE models, including the challenging case of an advection dominated PDE.

For the simulation of the PDE boundary value problem, we utilize discontinuous Galerkin finite element methods and adaptive spatial grid refinement. We solve the optimization problems with derivative-based algorithms. For the optimization algorithms to converge fast and to converge to the ”true“ optimum, we need to provide accurate sensitivities. It is a challenge to evaluate the sensitivities, which correspond to the approximate solution of the primal PDE model and are in this sense consistent. In this thesis we develop efficient and accurate methods for sensitivity generation. We transfer the principle of internal numerical differentiation (IND) from ordinary differential equations (ODE)s to PDEs. That means, we incorporate the sensitivity generation in the solution process. The standard upwind discontinuous Galerkin method is not differentiable. Therefore, we propose a differentiable discontinuous Galerkin method and give a rigorous convergence analysis of it. We develop methods for structure exploitation of the primal and tangential discretization schemes to efficiently generate the sensitivities with automatic differentiation (AD). Furthermore, we establish methods for frozen adaptivity to generate consistent sensitivities. We are especially concerned with frozen spatial grid refinement and the adaptive step number of the linear solver.

We implement the developed methods in the software `SeafaND-Optimizer`, short for *structure exploiting and frozen adaptivity numerical differentiation optimizer*. It is a software for efficient and accurate simulation, PE and OED with PDE models. We perform numerical case studies for PE and OED problems with advection dominated 2D diffusion advection PDE models. With the structure exploiting techniques developed in this thesis, the example problems are solved with efficient memory usage. Due to the frozen adaptivity methods, we computed efficiently the consistent sensitivities. We test the PE algorithm with different noise levels. We perform a case study with different diffusion coefficients for sequential OED. Finally, we investigate, whether the developed methods are stable under mesh refinements.

Zusammenfassung

In dieser Dissertation untersuchen wir Optimierungsprobleme mit partiellen Differentialgleichungen (PDE) als Nebenbedingungen. Wir beschäftigen uns insbesondere mit der effizienten numerischen Lösung von Problemen der optimalen Versuchsplanung (OED) für Parameterschätzung (PE) mit PDE Modellen, darunter das Problem der Stichprobenplanung. Es werden zweidimensionale (2D) stationäre Diffusion-Advektion-Reaktions-PDE Modelle betrachtet, einschließlich der anspruchsvolle Fall eines advektionsdominierten PDE Modells.

Um das PDE Randwertproblem zu simulieren, nutzen wir diskontinuierliche Galerkin Finite Elemente Methoden und adaptive räumliche Gitterverfeinerung. Die Optimierungsprobleme werden mit ableitungsbasierten Algorithmen gelöst. Damit die Optimierungsalgorithmen schnell und zum „wahren" Parameterwert konvergieren, müssen wir präzise Sensitivitäten bereitstellen. Es ist eine Herausforderung, Sensitivitäten auszuwerten, die konsistent zur approximativen Lösung des primalen PDE Problems passen. In dieser Arbeit entwickeln wir effiziente und präzise Methoden zur Ableitungserzeugung. Wir übertragen das Prinzip der Internen Numerischen Differentiation (IND) von gewöhnlichen Differentialgleichungen (ODE) auf PDE. Das heißt, die Ableitungserzeugung wird in den Lösungsprozess inkludiert. Die standardmäßige Upwind diskontinuierliche Galerkin Methode ist nicht differenzierbar. Eine differenzierbare Upwind diskontinuierliche Galerkin Methode wird vorgeschlagen und eine ausführliche Konvergenzanalyse der Methode wird durchgeführt. Wir entwickeln Methoden zur Strukturausnutzung der primalen und tangentialen Diskretisierungsschemata, um effizient die Sensitivitäten mit Automatischer Differentiation (AD) zu erzeugen. Außerdem werden Methoden mit eingefrorenen adaptiven Komponenten der PDE-Simulation erstellt, um konsistente Ableitungen zu erzeugen. Wir beschäftigen uns im Besonderen mit adaptiver räumlicher Gitterverfeinerung und der adaptiven Schrittanzahl des iterativen Lösers.

Die entwickelten Methoden haben wir in der neuen Software `SeafaND-Optimizer`, kurz für *structure exploiting and frozen adaptivity numerical differentiation optimizer*, implementiert. Es ist eine Software zur Simulation, Parameterschätung und optimalen Versuchsplanung mit PDE Modellen. Es werden numerische Fallstudien mit den PE und OED Problemen mit advektionsdominierten 2D Diffusion-Advections-PDE Modellen durchgeführt. Mit den in dieser Arbeit entwickelten strukturausnutzenden Methoden wurden die Beispielprobleme mit effizienter Speichernutzung gelöst. Durch

die Methoden mit eingefrorenen adaptiven Komponenten wurden die konsistenten Sensitivitäten berechnet. Wir testen den PE-Algorithmus mit unterschiedlich hohen Messstörungen. Für sequenzielles OED führen wir eine Fallstudie mit verschiedenen Diffusionskoeffizienten durch. Schließlich untersuchen wir, ob die entwickelten Methoden stabil unter Gitterverfeinerungen sind.

Acknowledgements (Danksagung)

I would like to express my gratitude to my advisors Prof. Dr. Dr. h. c. mult. Georg Bock, Prof. Dr. Guido Kanschat and Prof. Dr. Stefan Körkel for the opportunity to do research on this interesting topic and their support in many fruitful discussions. I would like to thank Prof. Dr. Ekaterina Kostina and Dr. Johannes Schlöder for their support and valuable advice on various subjects. My thanks go to my mentor Dr. Dörte Jando, for her continuous encouragement and many interesting and helpful discussions on the topic. Furthermore, I would like to thank Dr. Christoph Weiler for the initial ideas on the combination of `VPLAN` and `dealii` for sensitivity generation.

My thanks go to the Interdisciplinary Center for Scientific Computing and the Faculty of Mathematics and Computer Science at Heidelberg University for establishing such a stimulating academic environment for doing research. For the cooperative atmosphere I would like to thank my colleagues in the Simulation and Optimization group, the Junior Research Group Optimum Experimental Design and the Mathematical Methods of Simulation group. In particular, for many interesting professional and semi-professional discussions over cups of coffee, apple breaks and other occasions at the institute I would like to thank Dr. Daniel Arndt, Dr. Maria Barbarossa, Alexander Buchner, Dominik Cebulla, Dr. Holger Diedam, Asha Dond, Ph.D., Dr. Michael Engelhart, Dr. Lilli Frison, Dr. Markus Gahn, Dr. María Suárez Garcés, Dr. Joscha Gedicke, Ameli Gottstein, Dr. Jürgen Gutekunst, Dr. Kathrin Hatz, Johannes Herold, Dr. Christian Hoffmann, Dr. Dennis Janka, Felix Jost, Dr. Florian Kehrle, Arbaz Khan, Ph.D., Prof. Dr. Christian Kirches, Dr. Robert Kircheis, Dr. Tom Kraus, Manuel Kudruss, Dr. Huu Chuong La, Dr. Conrad Leidereiter, Dr. Felix Lenders, Dr. Simon Lenz, Dr. Pablo Lucero Lorca, Stefan Meggendorfer, Dr. Enrique Guerrero Merino, Andreas Meyer, PD Dr. Andreas Potschka, Nadia Said, Dr. Simeon Sauer, Marta Sauter, Matthias Schlöder, Dr. Andreas Schmidt, Robert Scholz, Dr. Noman Shakir, Dr. Andreas Sommer, Hilke Stibbe, Anastasia Walter, Dr. Sebastian Walter, Dr. Leonard Wirsching, Julius Witte and Lisa Wittmayer.

For commenting on parts of this thesis my gratitude goes to Dr. Daniel Arndt, Dr. Dörte Jando, Dr. Felix Lenders, Nadia Said and Dr. Andreas Sommer. For the successful teamwork in the executive board of the Heidelberg Chapter of SIAM I thank Dr. Felix Lenders, Dr. Conrad Leidereiter, Dr. Ole Klein, Dr. Jürgen Gutekunst, Dr. Diana-Patricia Danciu, Dominik Cebulla, Robert Scholz and Dr. Ruobing Shen.

My special thanks for immediate and professional help in administrative affairs go to Abir Al-Laham, Felicitas Hirsch, Anja Vogel, Jeanette Walsch, Anastasia Walter, Dorothea Heukäufer, Ria Hillenbrand-Lynott and Sarah Steinbach. Thanks for maintaining the computing facilities go to Thomas Klöpfer and Manfred Trunk.

Scientific and financial support is gratefully acknowledged by BASF SE within the Junior Research Group Optimum Experimental Design, Klaus Tschira Stiftung within the project SCIDATOS, the Heidelberg Graduate School Mathematical and Computational Methods for the Sciences (HGS) and the Society for Industrial and Applied Mathematics (SIAM).

Schließlich danke ich allen meinen Freunden und meiner Familie. Danke an meine Eltern Hedwig und Edgar, die mir neben vielem anderen schon früh die Freude an der Mathematik vermittelt haben. Insbesondere danke ich meiner Mutter Hedwig, meinem Bruder Andre und meinem Freund Andreas. Danke für die wertvolle Unterstützung, Motivation und eure Begeisterung für meine Arbeit!

Contents

List of acronyms

2D	two dimensional
AD	automatic differentiation
BFGS	Broyden-Fletcher-Goldfarb-Shanno
DAE	differential algebraic equation
DoFs	degrees of freedom
END	external numerical differentiation
EOC	experimental order of convergence
FE	finite element
IND	internal numerical differentiation
LICQ	linear independence constraint qualification
NLP	nonlinear programming problem
ODE	ordinary differential equation
OED	optimum experimental design
PDE	partial differential equation
PE	parameter estimation
`SeafaND-Optimizer`	structure exploiting and frozen adaptivity numerical differentiation optimizer
SQP	sequential quadratic programming
SR1	symmetric rank one

List of symbols

A_h	stiffness matrix
$a_h(y_h, v_h)$	diffusion part discrete PDE
$A_{F^{\Gamma}}$	local stiffness matrix for boundary faces
$A_{F^{int}}$	local stiffness matrix for interior faces
A_T	local stiffness matrix for cells
α	diffusion coefficient
α_r	statistical setting level of significance
β	advection factor
$b_h(p; y_h, v_h)$	advection part discrete PDE
$c_h(p; y_h, v_h)$	reaction part discrete PDE
c_m	constraint function OED
C	variance-covariance matrix
C_h	discrete variance-covariance matrix
δp_k	increment solution of linearized problem Gauss-Newton algorithm
D	operator form PDE
d	dimension of spatial domain
η	measurement data
ε	measurement errors
F	closed face of the grid cells
$\mathbb{F}_h$	set of faces of the grid cells
$\mathbb{F}_h^{\Gamma}$	set of boundary faces of the grid cells
$\mathbb{F}_h^{int}$	set of interior faces of the grid cells
$F(p; y, v)$	short notation PDE
$F_h(p; y_h, v_h)$	discrete short notation PDE
f	right hand side function of the PDE problem
$f_h(p; v_h)$	discrete right hand side PDE
$\tilde{f}_h$	load vector
$\tilde{f}_{F^{\Gamma}}$	local load vector for boundary faces
$\tilde{f}_{F^{int}}$	local load vector for interior faces
$\tilde{f}_T$	local load vector for cells
f_p	statistical setting joint probability distribution function
f_p^i	statistical setting probability distribution function
$\tilde{G}$	objective functional parameter estimation

G	reduced objective functional parameter estimation
G_h	discrete reduced objective functional parameter estimation
$g_h(p; y_h, v_h)$	discrete left hand side PDE
$g_{\Gamma}(p; y_h, v_h)$	structure exploitation boundary face integrand
$g_{\Gamma,r}(p; \varphi_j)$	structure exploitation right hand side contribution of boundary face integrand
$g_{int}(p; y_h, v_h)$	structure exploitation interior boundary face integrand
$g_{int,r}(p; \varphi_j)$	structure exploitation right hand side contribution of interior face integrand
$g_T(p; y_h, v_h)$	structure exploitation cell integrand
$g_{T,m}(p; \varphi_i, \varphi_j)$	structure exploitation matrix contribution of cell integrand
$g_{T,r}(p; \varphi_j)$	structure exploitation right hand side contribution of cell integrand
Γ	boundary of spatial domain Ω
γ	penalty of interior penalty discontinuous Galerkin method
γ_r^2	statistical setting quantile of the χ^2 distribution
∇	gradient
∇^2	Hessian
h	finite element grid size
$\bar{h}_i(p; y)$	model response
$h_i(p)$	reduced model response
$h_{i,h}(p)$	discrete reduced model response
h_e	characteristic length scale of the advection process
$\mathcal{H}$	Hilbert space
J	Jacobian
J_h	discrete Jacobian
$L(w, l)$	Lagrange function
l_j	Lagrange multiplier
$\mathcal{L}$	Lebesgue space
m	dimension measurement space
μ	variable in differentiable stabilization upwind discretization
n_b	number of basis functions
n_{dc}	number of degrees of freedom per cell
n_g	number of possible measurement functions
n	outward unit normal vector
n_p	dimension parameter space
n_q	number of quadrature points per element
Ω	spatial domain

$[\![\cdot]\!]$	jump operator
$\{\!\{\cdot\}\!\}$	average operator
P	parameter space
p	parameters
$\hat{p}$	estimated parameters
p^0	start values parameters
p^*	true parameters
∂	partial derivative
P_e	Péclet number
φ_i	basis function
$\mathit{\Phi}(C(w,p))$	OED criterion objective function of OED problem
$\mathit{\Phi}_h(C_h(w,p))$	discrete OED criterion objective function of OED problem
$\mathcal{P}(T)$	space of polynomials of degree $k \geq 0$
$\bar{r}_i(p;y)$	residuals of the parameter estimation problem
$r_i(p)$	reduced residuals of the parameter estimation problem
$r_{i,h}(p)$	discrete reduced residuals of the parameter estimation problem
ρ	reaction coefficient
σ_i^2	variance
$\sigma_\mu(\beta,n)$	differentiable stabilization upwind discretization
$\sigma_{upw}(\beta,n)$	standard stabilization upwind discretization
$\mathit{\Sigma}$	diagonal matrix of variances σ_i
S	solution operator
T	grid cell of triangulation
$\hat{T}$	reference element
t_k	step length Gauss-Newton algorithm
$\mathbb{T}_h$	triangulation
v	test function
v_h	discrete test function
$V(\Omega)$	test space
$V^\beta(\Omega)$	test space for advection
V_h	finite element space
$\hat{V}_h$	finite element space on reference element
$\mathcal{W}$	Sobolev space
$\mathcal{W}_p^k(\mathbb{T}_h)$	broken Sobolev space
w_i	sampling decision
w_q	quadrature weight
x_j^m	measurement point

x	spatial variable
x_q	quadrature point
y_D	Dirichlet boundary function
y	state variable
y_h	discrete state variable
Y	state space
Z	measurement space

1. Introduction

Mathematical models describe many processes in physics, chemistry, biology, engineering and even in social sciences and psychology. With a mathematical model it is possible to gain more insight into the underlying mechanisms and to predict future behavior. This is of great importance for practitioners and scientists.

Often, a mathematical model consists of a system of differential equations accompanied by a set of unknown, or little known, parameters. In this thesis we are particularly concerned with partial differential equation (PDE) models. To describe the process precisely, it is crucial to estimate the parameters accurately. We formulate a parameter estimation (PE) optimization problem, which minimizes the difference between experimentally obtained measurement data and simulated model response by varying parameter values. After estimating parameters, an important question is: how can we measure the quality of the estimation? One possibility to answer this question is to examine the statistical significance of the estimation. It can be described quantitatively by confidence regions, which depend on the variance of the estimates. Optimum experimental design (OED) aims at improving this statistical significance by minimizing the confidence regions of the parameters by changing the experimental conditions. The experimenter sets controls, which lead to a specific experimental design. These controls are included in a nonlinear optimization problem and those controls are searched for which the experimental setting leads to the statistically most significant parameter estimates. In particular, we treat the special case of sampling design, here the controls are sampling decisions to choose individual measurement points. Thus, in this thesis we consider optimization problems with PDE constraints, in particular OED problems for PE.

OED problems with PDE models are rarely investigated. So far, OED problems are mainly treated with ordinary differential equation (ODE) models. In [75], [16], [66], [96] and [56] the authors consider OED with ODE and differential algebraic equation (DAE) models. They treat the underlying PE problem as a constrained optimization problem and solve it by an all-at-once approach. They transfer the OED problem into a finite dimensional nonlinear programming problem (NLP) and solve it by sequential quadratic programming (SQP) methods. For an one-dimensional PDE constraint, which is reduced by the method of lines to an ODE constraint, [7] and [6] use a similar approach.

For OED with PDE models and finitely many parameters, recently [79] and [50] solved an OED problem for an application in material science. In [39], the authors combined OED and shape optimization: they treat the shape as an experimental control. All of them use derivative-based optimization algorithms. Furthermore, in the recent paper [76], the authors propose an OED problem with a measurement setup based on a positive Borel measure. As solution method they present a generalized conditional gradient method in measure space.

Preliminary work similar to our approach for OED with PDE models has been done by [34], [33], [97], [63] and [70]. In contrast to the ODE case, they solve an unconstrained PE problem and use a reduced approach for the underlying PE problem. This reduced approach for PE problems with PDE models has been extensively researched, see for example [17], [18], [34], [22], [52], [3]. To solve the OED problem, which is a nonlinear constrained optimization problem, [34] and [33] use a derivative-free method, whereas [97], [63] and [70] utilize derivative based optimization algorithms. Particularly, they use SQP methods to solve the nonlinear OED problem.

Simulation of the PDE model

Before discussing methods for optimization problems, we concentrate on simulation problems for PDE models. We consider stationary 2D diffusion advection reaction PDE models. To cover a wide range of parameters, which is important in applications, we include advection dominated PDE models. The simulation of this advection dominated PDE model is a challenge per se, because the standard continuous Galerkin finite element method produces spurious oscillations [58]. A solution to that problem is the use of discontinuous Galerkin finite element methods. For the diffusion part of the PDE model, we select the symmetric interior penalty discontinuous Galerkin method [9], [10]. For the advection part, we choose the upwind discontinuous Galerkin method [84]. For the reaction part, we add an additional mass matrix.

A further challenge for the simulation of the PDE model is to compute a solution with low effort for a prescribed accuracy. If we achieve this, it is possible to apply our methods even to large-scale PDE models. Therefore, we investigate the well-established technique of adaptive grid refinement for the spatial finite element grid [83], [94].

Optimization: sensitivity generation

To solve PE and OED problems with derivative-based algorithms, we require in particular sensitivities of the underlying PDE boundary value problem, that means the forward or adjoint variational differential equations. The variational differential equations are also called tangential PDEs or adjoint PDEs. We need accurate and consistent sensitivities, otherwise the optimization algorithms could converge to a false parameter value or converge slowly. Furthermore, the sensitivity generation is, besides the simulation of the PDE model, the numerically most expensive part of the optimization algorithm. Thus for an efficient algorithm it is crucial to compute sensitivities with low numerical costs. Another error prone aspect is the operation of the program: if the user has to provide sensitivities and thus variational differential equations, this easily leads to errors. Therefore the sensitivities should be generated by the program. Thus the challenge here is to efficiently generate and compute consistent sensitivities.

Presently, two main approaches for sensitivity generation exist in the literature. First, analytically derive the variational differential equations by the sensitivity or adjoint approach. Then discretize these PDE problems and solve the originating systems. Second, use automatic differentiation (AD) to calculate sensitivities. That means differentiate the code, which means the programmed version of the discretization.

For the first approach some work exists for PE with non-advection dominated PDE models, for example [17]. Few work exists for PE with advection dominated PDE models: in [18] a PE problem with an advection dominated PDE model is investigated. The PDE model is spatially discretized by conforming finite elements. In [52], [22] and [34], the authors consider a PE problem with an advection dominated flow. The PDE model is discretized by a continuous Galerkin method with stabilization. Of the aforementioned publications, [17], [22] and [34] use the technique of adaptive grid refinement.

The second approach is rarely investigated, because in the PDE framework it quickly leads to memory issues, which makes the sensitivity generation with AD far too expensive or even impossible [91]. To our knowledge, there are no publications for PE or OED problems. In previous work the whole code has been processed by an AD tool to build the sensitivities [46]. When the problem is large-scale, which is easily the case in the PDE framework, the algorithm is not able to derive and store the sensitivities. In spite of this black box approach, there exist few publications where the structure of the discretization is considered [91], [55], [90], [49], [36]. In [91],

[55] and [90] an optimal control problem with a PDE model is treated. The PDE is discretized by a discontinuous Galerkin method. The authors exploit the structure of the discontinuous Galerkin method by differentiating only parts of the discretization with AD, namely the formula of the reference finite element. In [49] and [36], the authors consider a shape optimal design problem. The PDE model is discretized by a finite volume and finite element discretization. All of them differentiate only parts of the discretization with AD to avoid memory issues. They do not use adaptive grid refinement.

Both just presented approaches possess disadvantages. In the first approach, which is mostly used for PE with PDE models, the user has to differentiate and implement the sensitivities by hand. This procedure is very error prone. The second approach is rarely used with PDE models, because it easily leads to severe storage space issues, which makes it impossible to generate the sensitivities.

Furthermore both approaches do not consider the adaptive components of the solver when generating the sensitivities. The adaptive components change depending on the input, they are not differentiable. If the adaptive components are chosen differently for the solving of state and sensitivity equations, it is not clear, if consistent sensitivities are computed. The error in the generated sensitivities can become arbitrary large.

To circumvent these problems, we utilize a different way: we transfer the principle of internal numerical differentiation (IND) to PDEs. This principle was first introduced by Bock for ODEs [25], [26], [27] and extended amongst others by [4], [5], [15]. For a comprehensive overview see [4] and [89]. Following the principle of IND, we include the sensitivity evaluation into the numerical scheme. That means, we differentiate the discretization of the PDE model to get the accurate and consistent discrete sensitivities that approximate the continuous counterparts. First steps in this direction are made in [70], [63] and [97]. In [70] and [63] the discretization is differentiated by hand without AD. In [97], the author differentiates parts of the discretization by AD. In contrast to our problem, the PDE model is not advection dominated and solved with the continuous Galerkin method. All of them do not use adaptive grid refinement.

1.1. Results of this thesis

The aim of this thesis is the efficient numerical solution of OED problems for PE with PDE models. We especially include the case of advection dominated PDE models. As OED problem we treat a sampling design problem. To efficiently generate the consistent sensitivities, we transfer the principle of IND to PDE models. We include the sensitivity generation into the numerical scheme. We differentiate the discretization of the PDE model to get consistent sensitivities. More precisely we investigate three relevant topics regarding the transfer of IND to PDE models: First, we propose a differentiable upwind discontinuous Galerkin discretization for the advection part of the PDE model and give a rigorous theoretical analysis of it. Second, we propose two ways to exploit the common structure of primal and tangential discretization schemes. This results in an efficient memory usage and improved computational performance. The program generates the sensitivities automatically and accurately, the user does not have to provide them. Third, we freeze the adaptive components of the simulation algorithms. This must be done to generate consistent sensitivities. In contrast to existing literature, we apply adaptive grid refinement and investigate the influence of the adaptive iterative solver on the sensitivity generation. We have implemented all developed methods in the new software `SeafaND-Optimizer`, short for *structure exploiting and frozen adaptivity numerical differentiation optimizer (`SeafaND-Optimizer`)*. We demonstrate the efficiency and accurateness of the methods by several numerical test cases.

Thus, this thesis presents novel efficient methods for sensitivity generation in the field of PDE constrained optimization problems. In the following we describe the main findings in detail.

OED for PE with advection dominated PDE model

We propose advanced methods for sensitivity generation to solve OED and PE problems with PDE models. In contrast to existing approaches, we use derivative-based optimization methods combined with a discontinuous Galerkin discretization and adaptive grid refinement. That way, we develop algorithms, that can be applied to a broad class of problems. The discontinuous Galerkin method is robust, it is especially suited for advection dominated PDE models, but also suitable for non-advection dominated PDE models.

Differentiable discretization: theoretical analysis and numerical results

To transfer the principle of IND to PDE models, the discretization of the PDE model has to be differentiable. The standard upwind discontinuous Galerkin method [84], [74], [59] is non-differentiable, because of the choice of the numerical fluxes. In [91] the authors avoid the problems arising by this discontinuity in manually constructing an algorithm such that the fluxes are chosen correctly. To overcome this problem of non-differentiability in a more general way we propose a new differentiable stabilization for the advection in the upwind discontinuous Galerkin method. A detailed theoretical convergence analysis of the new differentiable upwind discontinuous Galerkin method is presented, including stability estimate and error estimate. The differentiable upwind method has the same properties and error estimates as the standard upwind method. In addition it is differentiable. Furthermore, numerical results show the predicted behavior.

Structure exploitation of primal and tangential discretization schemes

Following the principle of IND, we use the same discretization for primal and tangential PDE models. Therefore, we determine two possibilities to exploit the common structure of primal and tangential discretization schemes. First, we exploit the problem structure. We reuse parts of the discretized primal problem for the generation of the discretized tangential problems as well as for the computational solution of the discretized problems. Second, we exploit the structure of the finite element method. We employ tailored algorithmic differentiation for the elements of the discontinuous Galerkin method. In contrast to [91], we go one step further and differentiate the innermost formula with AD to generate code to approximate the tangential PDEs. That means, instead of differentiating the formula of the reference element such as [91], we differentiate the core part of the quadrature formula. In only differentiating core parts of the code, we save a considerable amount of memory space in comparison to black box AD, where the whole code is differentiated. Furthermore, we obtain a much higher accuracy than for example evaluating the sensitivities with finite differences. Furthermore, the user does not have to provide sensitivities. The program generates them automatically and accurately. We demonstrate the efficiency of the developed structure exploiting methods by numerical examples.

Freezing of adaptive components

To transfer the principle of IND to PDE models, we freeze all adaptive components to generate consistent sensitivities. Possible adaptive components are adaptive grid refinement with an error indicator of the spatial finite element grid and the adaptive step number of an iterative solver of the linear system, which we solve for the simulation of the PDE models. We must discretize all PDE problems, that means primal and tangential problems, on one common finite element (FE) grid to generate consistent sensitivities. We develop a heuristic: the *error sum strategy* for grid refinement. The error sum strategy generates one adaptively refined grid which is suitable for the simulation of primal and tangential PDE problems. With regard to the adaptive step number of an iterative solver, we analyze two possible options to solve the linear systems of all PDE models with an iterative solver. We select the option, which in our scenario approximates the consistent sensitivities. We demonstrate the developed methods for the evaluation of sensitivities by numerical examples.

Implementation: software `SeafaND-Optimizer`

We implemented all developed methods in a software called `SeafaND-Optimizer`, which is short for *structure exploiting and frozen adaptivity numerical differentiation optimizer*. It is a software for simulation, parameter estimation and optimum experimental design with PDE models. In existing software for optimization with PDE models, for example `DOpElib` [44] or `RoDoBo` [19], the user has to set up the tangential PDE problems by hand. The software `dolfin-adjoint` [40], [41] utilizes symbolic differentiation to compute the sensitivities. To our knowledge, the authors [40], [41] do not yet apply their technique to an optimization example with adaptive FE grids. In the software `SeafaND-Optimizer` we implemented our approach for sensitivity generation to transfer the principle of IND to PDE models. That means to develop a differentiable upwind discontinuous Galerkin method, to exploit the structure and to freeze all adaptive components to efficiently generate consistent sensitivities with AD. Furthermore, we utilize adaptive FE grids. For simulating the PDE models, all functionalities of `dealii` and `amandus` are available inside the `SeafaND-Optimizer`. For the optimization problems the `VPLAN` interface and optimization algorithms are provided. For PE we select a Gauss-Newton algorithm with step size control in the extension `PAREMERA` and for OED a SQP algorithm implemented in `SNOPT`. Thus with the `SeafaND-Optimizer` we efficiently and accurately solve PE and OED problems with PDE models.

Case studies for PE and OED

We illustrate the efficiency of the developed methods by several example PE and OED problems with advection dominated 2D diffusion advection reaction PDE models. Because of our developed structure exploitation methods, no memory issues occur while generating the tangential problems. With the error sum strategy for adaptive grid refinement, the PDE models are simulated with low computational effort for a prescribed accuracy. The number of degrees of freedom for each PDE model problem, one primal and two tangential problems, is approximately 131,500. The PE algorithm converges linearly for different noise levels. We perform successfully sequential OED for different diffusion coefficients. Thus the developed methods are suitable for a class of problems. Furthermore, we execute a numerical study on mesh independence for both, PE and OED problems. We conclude, that the study gives strong evidence that the developed methods are stable under grid refinements.

1.2. Thesis overview

This thesis is divided in five parts, which comprise twelve chapters.

Part I introduces the problem formulation. In Chapter 2 we depict a general class of PE problems with PDE models. We formulate two PDE model problems, which will be revisited at later points in this thesis. With that we formulate the PE problem as a constrained optimization problem. We reformulate this constrained optimization problem with the reduced approach to an unconstrained PE problem. As a numerical solution method for this unconstrained PE problem, we present a Gauss-Newton method. We conclude this chapter with a statistical setting and a sensitivity analysis of the PE problem. This sensitivity analysis is needed to formulate the OED problem in Chapter 3. We define the experimental design and formulate a nonlinear OED problem. As OED problem we treat the special case of sampling design, also called optimal placement problem, optimization of sensor locations or measurement selection. We vary measurement points to enhance the significance of parameter estimates and minimize the uncertainty of the parameter estimation. Furthermore, we treat the relaxation of integer constraints and survey problem variants of OED problems. After that, we give optimality conditions for the OED problem. As a numerical solution method we present the sequential quadratic programming (SQP) method.

Part II gives an overview of the status quo in PDE discretization and sensitivity evaluation techniques. Chapter 4 presents the foundations of the discontinuous Galerkin methods. We especially outline the standard upwind discontinuous Galerkin method, which we later on extend to a differentiable version. We present the discrete problem for the diffusion advection reaction model problem. With that, we introduce the discrete optimization problems. Finally, we recall the finite element algorithm to simulate the PDE models. In Chapter 5 we survey sensitivity evaluation techniques.

First, we state which sensitivities we would like to generate. Afterwards, we discuss the principle of IND, analytical sensitivity evaluation and automatic differentiation.

In Part III we propose and theoretically analyze a differentiable upwind discretization. In Chapter 6, we propose a differentiable upwind discontinuous Galerkin discretization and recapitulate some basic approximation formulas. We perform a theoretical analysis for the pure advection model problem based on a standard procedure for convergence analysis of finite element discretizations. This analysis includes consistency of the discretization, coercivity, a stability estimate, an error estimate for the $\mathcal{L}^2$ projection, an estimate for the bilinear form, an error estimate in the energy norm and a superconvergence result. Furthermore, we show that our analysis holds as well for a diffusion advection reaction model problem. In this case, we have to make changes. Therefore, we again follow the standard procedure to finally show an error estimate in the energy norm. For the case of a non normalized advection coefficient, we proof an error estimate in the energy norm. We show that in this case the convergence analysis changes, the constant is now dependent on the advection coefficient. All other properties remain unchanged for the differentiable upwind discretization for the non normalized advection coefficient. We close the chapter by numerical examples, which confirm the developed theory.

In Part IV we develop new methods for sensitivity generation. We transfer the principle of IND to PDE models. In Chapter 7 we propose structure exploiting methods for the primal and tangential discretization schemes. We exploit the problem structure by reusing common parts of the primal and tangential discretization schemes. Furthermore, we develop an algorithm to exploit the structure of the discontinuous FE method. We demonstrate the efficiency of the developed methods by numerical examples. In the next Chapter 8 we propose methods to freeze the adaptive components of primal and tangential discretization schemes. We investigate sensitivity generation with regard to adaptive grid refinement of the spatial FE grid. We give a literature overview and identify possible difficulties. We develop a heuristic, the error sum strategy for grid refinement, to generate a common spatial adaptively refined FE grid for primal and tangential problems. After that we investigate two options to apply an iterative solver to the linear system: the piggyback approach and the two-phase approach. Again, we close this chapter with numerical examples.

In the last Part V we present the software `SeafaND-Optimizer` and demonstrate the efficiency and accurateness of the developed methods for sensitivity generation by numerical results for PE and OED problems. In Chapter 9 we explain the functionality of the `SeafaND-Optimizer`. We give an overview over the software package, explain the program structure and the workflow of the `SeafaND-Optimizer`. In Chapter 10 we show numerical results for PE and OED problems with 2D advection dominated diffusion advection PDE models. We first test the developed methods by a PE problem with three different noise levels. After that, we perform a case study with different diffusion coefficients for sequential OED. That means, we test if the algorithms are applicable to a class of problems. In the last Chapter 11 we execute a numerical study on mesh independence. We begin with a PE problem. We compare the results of the optimization algorithm for different grid refinements of the underlying PDE simulation. Finally, we test an OED problem for different grid refinements of the PDE simulation to determine whether the developed algorithms are stable under mesh refinement.

Part I.

Problem formulation

2. Parameter estimation with PDE models

In this chapter we introduce the parameter estimation problem for PDE models. First we formulate PDE boundary value problems. After that we begin with a constrained parameter estimation problem and utilize a reduced approach to attain the unconstrained parameter estimation problem. To solve it numerically we utilize the Gauss-Newton method. Finally we investigate the statistical setting of the parameter estimation problem to establish the optimum experimental design problem, which we describe in the next chapter.

2.1. PDE boundary value problem

Spaces, norms and derivatives We begin with defining two associated function spaces and their corresponding norms: the Lebesgue space and the Sobolev space.
2.1.1 Definition. For any number $p, 1 \leq p \leq \infty$, let the *Lebesgue space* $\mathcal{L}^p(\Omega)$ be the space of functions which are p-integrable on Ω. △

The corresponding *Lebesgue norm* is defined by

$$\|v\|_{\mathcal{L}^p(\Omega)} := \left(\int\limits_{\Omega} |v|^p dx \right)^{\frac{1}{p}}, \quad \text{if } 1 \leq p < \infty.$$

We are especially using the Lebesgue space for $p = 2$. The corresponding *Lebesgue norm* will be abbreviated by $\|.\| := \|.\|_{\mathcal{L}^2(\Omega)}$. If it is not clear on which domain the norm operates, we write $\|.\|_\Omega := \|.\|_{\mathcal{L}^2(\Omega)}$. We shorten the notation in the usual way by defining the $\mathcal{L}^2(\Omega)$ scalar product

$$(y, v)_\Omega := \int\limits_{\Omega} yvdx.$$

Similarly, we define a Lebesgue space for the boundary of the domain $\partial\Omega$, with the

Lebesgue norm

$$\|v\|_{\mathcal{L}^p(\partial\Omega)} := \left(\int\limits_{\partial\Omega} |v|^p ds \right)^{\frac{1}{p}}, \quad \text{if } 1 \leq p < \infty.$$

and $\mathcal{L}^2(\partial\Omega)$ scalar product

$$(y, v)_{\partial\Omega} := \int\limits_{\partial\Omega} yvds.$$

Furthermore, we define *Sobolev spaces* $\mathcal{W}_p^k(\Omega)$ to include information about the partial derivatives in the function space.

2.1.2 Definition. For any integer $k \geq 0$ and any number $p, 1 \leq p < \infty$, the *Sobolev space* $\mathcal{W}_p^k(\Omega)$ consists of all functions $v \in \mathcal{L}^p(\Omega)$ for which all partial derivatives $\partial^\omega v$ with $|\omega| \leq k$ belong to the space $\mathcal{L}^p(\Omega)$ [35], [1]. △

The corresponding *Sobolev norm* is defined by

$$\|v\|_{\mathcal{W}_p^k(\Omega)} := \left(\sum_{|\omega| \leq k} \int\limits_{\Omega} |\partial^\omega v|^p dx \right)^{\frac{1}{p}}, \quad \text{if } 1 \leq p < \infty,$$

and the *Sobolev semi-norm* is

$$|v|_{\mathcal{W}_p^k(\Omega)} := \left(\sum_{|\omega| = k} \int\limits_{\Omega} |\partial^\omega v|^p dx \right)^{\frac{1}{p}}, \quad \text{if } 1 \leq p < \infty.$$

For $p = 2$ together with the scalar product

$$(y, v)_{\mathcal{W}_2^k(\Omega)} := \sum_{|\omega| \leq k} \int\limits_{\Omega} \partial^\omega y \partial^\omega v dx,$$

the Sobolev spaces are *Hilbert spaces*. Following the commonly used notation we write $\mathcal{H}^k(\Omega) := \mathcal{W}_2^k(\Omega)$. We indicate by $\mathcal{H}_0^k(\Omega)$ the closure of $C_0^\infty(\Omega)$ in the space $\mathcal{H}^k(\Omega)$. For further definitions and basic properties of Sobolev spaces see [1], [35], [47].

Moreover, we define directional derivatives and Fréchet derivatives [54], [89].

2.1.3 Definition. Let $A : X \to Y$ be an operator. For Banach spaces X and Y, the operator $\frac{dA}{dx}\delta x$ is named *directional derivative* on $X_0 \subset X$ at $x \in X_0$, if the limit

$$\frac{dA}{dx}\delta x := \lim_{h\to 0} \frac{A(x+h\delta x) - A(x)}{h} \qquad \in Y,$$

exists for directions $\delta x \in X$. △

We use this notation for directional derivatives, because we will be concerned with directional derivatives of matrices.

2.1.4 Definition. Let $A : X \to Y$ be an operator with X, Y Banach spaces. The operator A is named *Fréchet differentiable* on $X_0 \subset X$ at $x \in X_0$, if there exists a linear bounded operator $A'(x) : X \to Y$, that means $A'(x) \in \mathcal{L}(X, Y)$, such that

$$\left\| A(x+\delta x) - A(x) - A'(x)\delta x \right\|_Y = o(\|\delta x\|_X), \quad \text{for } \|\delta x\|_X \to 0.$$

We indicate that A is continuously Fréchet differentiable if A is Fréchet differentiable and A is continuous. △

The definitions hold in our setting, because Hilbert spaces are a special case of Banach spaces.

PDE boundary value problems We start with two examples: a diffusion advection reaction PDE boundary value problem and a pure advection PDE boundary value problem.

2.1.5 Example. *Diffusion advection reaction boundary value problem.* We consider the following partial differential equation in diffusion advection reaction form, which acts as a constraint of the parameter estimation optimization problem,

$$-\nabla \cdot (\alpha \nabla y) + \beta(p) \cdot \nabla y + \rho(p) y = f(p) \qquad \text{on } \Omega, \tag{2.1a}$$

$$y = y_D(p) \qquad \text{on } \mathit{\Gamma}. \tag{2.1b}$$

Our goal is to estimate the unknown parameters $p \in P \subseteq \mathbb{R}^{n_p}$. The state variable $y \in Y$ is characterized by the partial differential equation boundary value problem (2.1). The state space Y is a Hilbert space $\mathcal{H}^k(\Omega), k \geq 1$. Let Ω be a convex domain in $\mathbb{R}^d$, $d = 2$, with boundary $\mathit{\Gamma} := \partial\Omega$.

The operator ∇y designates the gradient, $\nabla y := (\partial_1 y, ..., \partial_d y)^T$, with ∂_i partial derivative with respect to the spatial variable $x_i, i = 1, ..., d$. The divergence operator $\nabla \cdot y$ is defined by $\nabla \cdot y := \sum_{i=1}^d \partial_i y_i$. Furthermore, we establish the advection operator $\beta(p) \cdot \nabla y := \sum_{i=1}^d \beta_i(p)\partial_i y$, where the advection direction $\beta(p)$ is a constant vector in

$\mathbb{R}^d$.

The diffusion coefficient $\alpha > 0$ and the reaction coefficient $\rho(p) \geq 0$ are constants. Furthermore, let $f(p) \in \mathcal{L}^2(\Omega)$ be the right hand side function and $y_D(p) \in \mathcal{L}^2(\mathit{\Gamma})$ the Dirichlet boundary function with $\mathit{\Gamma}$ the boundary of the domain Ω. The reaction coefficient $\rho(p)$, the advection direction $\beta(p)$, the right hand side function $f(p)$ and the Dirichlet boundary function $y_D(p)$ are allowed to be parameter dependent.

2.1.6 Remark. The ratio between diffusion and advection rate can be expressed by the *Péclet number* $P_e := \frac{\|\beta\| h_e}{\alpha}$ [85]. The constant h_e is defined as the characteristic length scale of the problem setting [92]. To calculate it, the finite element grid size h is used. We are primarily interested in the advection dominated case with a large Péclet number P_e [85]. In practical applications that means P_e is much larger than 1. Conversely, P_e much smaller than 1 represents a diffusion dominated case. A Péclet number around 1 expresses that advection and diffusion are equally important [82], [92]. △

For a homogeneous Dirichlet boundary condition $y = 0$ on $\mathit{\Gamma}$, we get the *weak* or *variational form* of the partial differential equation boundary value problem (2.1) by multiplying with test functions $v \in V(\Omega) := \mathcal{H}_0^k(\Omega), k \geq 1$, building the integral over the domain Ω, using integration by parts and the fact that the advection direction $\beta(p)$ is defined as a constant vector, independent of the spatial variables. The weak form reads: Find $y \in \mathcal{H}_0^k(\Omega)$ such that

$$\alpha a(y, v) + b(p; y, v) + \rho c(p; y, v) = f(p; v), \quad \forall v \in V(\Omega). \tag{2.2a}$$

The left hand side splits into a diffusion part

$$a(y, v) = \int_\Omega (\nabla y, \nabla v) dx, \tag{2.2b}$$

an advection part

$$b(p; y, v) = -\int_\Omega (y, \beta(p) \cdot \nabla v) dx \tag{2.2c}$$

and a reaction part

$$c(p; y, v) = \int_\Omega (y, v) dx. \tag{2.2d}$$

The right hand side reads

$$f(p;v) = \int\limits_{\Omega} (f(p), v)dx. \tag{2.2e}$$

For a non-homogeneous Dirichlet boundary condition $y = y_D(p)$ on $\mathit{\Gamma}$, we assume that $y_D \in \mathcal{L}^2(\Omega)$ so that there exits an extension of y_D. Let this extension be $q_D \in \mathcal{H}^k(\Omega), k \geq 1$, such that $q_D = y_D(p)$ on $\mathit{\Gamma}$. With $y := q_D + y_w$, $y_w \in \mathcal{H}_0^k(\Omega)$, the variational form (2.2) also holds for the inhomogeneous case with $y \in \mathcal{H}^k(\Omega)$ [38].

We define a short notation by $F : P \times Y \times Y \to \mathbb{R}$,

$$F(p;y,v) := \alpha a(y,v) + b(p;y,v) + \rho c(p;y,v) - f(p;v).$$

The weak form (2.2) written in short notation reads

$$F(p;y,v) = 0, \quad \forall v \in V(\Omega).$$

This way, we are able to pose the parameter estimation problem in a more elegant form in the next paragraph. Furthermore the sensitivity generation in Chapter 7 and Chapter 8 will be easier to read and understand in this short notation. △

If y is the solution of the weak form (2.2), with sufficient regularity it is also a classical solution of the strong form of the differential equation boundary value problem (2.1) [38], [58, p.37ff], [82]. Thus it suffices to solve the weak form (2.2), to obtain a solution for the partial differential equation boundary value problem (2.1). The well posedness of the weak form of the PDE follows e.g. from [82], [38]. The approximation of the solution of the weak form will be explained in detail in Chapters 4 and 6.

2.1.7 Example. *Pure advection boundary value problem.* The pure advection boundary value problem reads

$$\beta(p) \cdot \nabla y = f(p) \qquad \text{on } \Omega, \tag{2.3a}$$

$$y = y_D(p) \quad \text{on } \mathit{\Gamma}_-(p), \tag{2.3b}$$

for $\|\beta(p)\| \neq 0$. As before, the unknown parameters p lie in the parameter space $P \subseteq \mathbb{R}^{n_p}$. The domain Ω is a convex domain in $\mathbb{R}^d$, $d = 2$.

The inflow boundary is defined as $\mathit{\Gamma}_-(p) := \{x \in \partial\Omega | n(x) \cdot \beta(p) < 0\}$, the outflow boundary is the complement of the inflow boundary $\mathit{\Gamma}_+(p) := \mathit{\Gamma} \setminus \mathit{\Gamma}_-(p)$. As before $y_D(p) \in \mathcal{L}^2(\mathit{\Gamma}_-(p))$ is the inflow boundary function and $f(p) \in \mathcal{L}^2(\Omega)$ is the right hand side function.

The variational form of (2.3) reads

$$b(p;y,v) = f(p;v), \quad \forall v \in V(\Omega). \tag{2.4a}$$

The left hand side consists only of an advection part

$$b(p;y,v) = -\int\limits_{\Omega} (y, \beta(p) \cdot \nabla v) dx + \int\limits_{\Gamma_+(p)} ((\beta(p) \cdot n) y, v) ds, \tag{2.4b}$$

and the right hand side additionally possesses an inflow boundary term

$$f(p;v) = \int\limits_{\Omega} (f(p), v) dx - \int\limits_{\Gamma_-(p)} ((\beta(p) \cdot n) y_D, v) ds. \tag{2.4c}$$

This variational form is well defined if y and v belong to the test space $V(\Omega) = V^{\beta}(\Omega)$, defined by

$$V^{\beta}(\Omega) := \{v \in \mathcal{L}^2(\Omega) : \beta \cdot \nabla v \in \mathcal{L}^2(\Omega)\}.$$

Again, we define a short notation, bilinear form $F^a(p;y,v)$ reads

$$F^a(p;y,v) := b(p;y,v) - f(p;v).$$

The weak form (2.4) in short notation is

$$F^a(p;y,v) = 0, \quad \forall v \in V(\Omega).$$

△

General operator form In the next step, we formulate the weak PDEs in operator form. Therefore we define an operator for the diffusion advection reaction Example 2.1.5 by $D : P \times Y \to Y^*$, here Y^* is the dual of Y, $< .,. >_{Y^* \times Y}$ denotes the duality pairing between Hilbert space Y and its dual Y^*,

$$< D(p;y), v >_{Y^* \times Y} = F(p;y,v), \quad \forall v \in V(\Omega).$$

The weak form (2.2) in operator form reads

$$D(p;y) = 0.$$

Analogously for the pure advection Example 2.1.7 the operator D^a reads

$$< D^a(p;y), v >_{Y^* \times Y} = F^a(p;y,v), \quad \forall v \in V(\Omega).$$

The weak form (2.4) in operator form is

$$D^a(p;y) = 0.$$

2.2. Parameter estimation problem

Constrained Parameter Estimation Problem The model responses, or in other words the measurement functions, $\bar{h} : P \times Y \to Z$ map the parameters p and the state variable y to the measurement space $Z = \mathbb{R}^m$. The vector $\bar{h}$ consists of single measurement functions $\bar{h}_i(p;y), i = 1, ..., m$. The dimension of the measurement space is assumed to be greater or equal than the dimension of the parameter space $m \geq n_p$. Both, parameter and measurement space are assumed to be finite dimensional.

The measurement data $\eta \in Z$ consists of single measurements $\eta_i \in \mathbb{R}, i = 1, .., m$, which are the values of measurement functions $\bar{h}_i$ with respect to the true parameters p^*, plus measurement errors ε_i,

$$\eta_i = \bar{h}_i(p^*;y) + \varepsilon_i, \quad i = 1, .., m.$$

We assume the errors to be independent and normally distributed with zero mean and variances σ_i^2, $\varepsilon_i \sim \mathcal{N}(0, \sigma_i^2)$.

The least squares functional $\bar{G} : P \times Y \to \mathbb{R}$ calculates the difference between the model response $\bar{h}_i(p;y), i = 1, ..., m$, and the measurement data η_i. We define a weighted least squares functional by

$$\bar{G}(p;y) := \frac{1}{2} \sum_{i=1}^{m} \left(\frac{\eta_i - \bar{h}_i(p;y)}{\sigma_i} \right)^2 = \frac{1}{2} \sum_{i=1}^{m} \bar{r}_i^2(p;y). \tag{2.5}$$

The residuals $\bar{r}_i(p;y)$ are functions from $P \times Y$ to $\mathbb{R}$. We assume, that the residuals are continuously differentiable.

With these ingredients we define a constrained parameter estimation problem:
2.2.1 Problem. ***(Constrained Parameter Estimation Problem)*** *Minimize the least squares functional $\bar{G}$*

$$\min_{p \in P, y \in Y} \bar{G}\left(p; y\right)$$

subject to

$$F(p; y, v) = 0, \quad \forall v \in V(\Omega),$$

which is given as a weak form of a partial differential equation boundary value problem defined in Section 2.1.

The least squares functional $\bar{G} : P \times Y \to \mathbb{R}$ is minimized subject to the parameters p and the state variables y. We are not including inequality constraints in the optimization problem. The weighted least squares functional $\bar{G}(p; y)$ is known to deliver a maximum likelihood estimate, see the following Section 2.4.

Unconstrained / Reduced Parameter Estimation Problem We reformulate the constrained parameter estimation Problem 2.2.1 as an unconstrained or reduced problem. Up to now, the optimization variables are the states y and the parameters p. In the reduced problem, the optimization variables are only the parameters p. In that way the optimization problem has a lower dimensionality. The storage requirements are reduced during the solution of the optimization problem, see for example [93], [54]. Let us make two assumptions: first let $\bar{G}(p; y)$ and $F(p; y, v)$ be continuously Fréchet-differentiable. Second let the state equation $F(p; y, v) = 0$ possess for each $p \in P$ a unique corresponding solution $y(p) \in Y$, which follows directly from the well-posedness of the weak formulation of the PDE. Then there exists the solution operator $S(p)$, $S : P \to Y$. It satisfies the state equation

$$F(p; S(p), v) = 0, \quad \forall v \in V(\Omega).$$

We insert this operator $S(p)$ into our constrained parameter estimation Problem 2.2.1 and obtain the corresponding unconstrained or reduced problem.

The reduced cost functional $G : P \to \mathbb{R}$ is defined by inserting the solution operator $S(p)$ in the constrained cost functional $\bar{G}(p; y)$

$$G(p) := \bar{G}\left(p; S(p)\right). \tag{2.7}$$

Next we define in the same manner the reduced measurement functions $h_i : P \to Z$, $h_i(p) := \bar{h}_i(p; S(p))$, and the reduced residuals $r_i : P \to \mathbb{R}$, $r_i(p) := \bar{r}_i(p; S(p))$. Thus the reduced cost functional $G(p)$ altogether reads

$$G(p) = \frac{1}{2} \sum_{i=1}^{m} \left(\frac{\eta_i - h_i(p)}{\sigma_i} \right)^2 = \frac{1}{2} \sum_{i=1}^{m} r_i^2(p).$$

We collect the single elements $h_i(p)$, $r_i(p), \eta_i$ and ε_i in vectors $h(p), r(p), \eta$ and ε, respectively.

Now we can state the unconstrained parameter estimation problem.
2.2.2 Problem. ***(Unconstrained Parameter Estimation Problem)*** *Minimize the reduced least squares functional $G(p)$*

$$\min_{p \in P} G(p).$$

The solution of a weak form of a PDE boundary value problem enters the unconstrained parameter estimation Problem 2.2.2 via the solution operator $S(p)$. The only optimization variables are the parameters p.

The existence and uniqueness of a solution to the parameter estimation problems is highly problem dependent. We assume that solutions to both the constrained Problem 2.2.1 and the unconstrained Problem 2.2.2 exist. Later in this thesis we obtain discrete optimization problems, which are discrete with regard to the underlying PDE boundary value problem. We assume that solutions to these corresponding discrete optimization problems exist.

A detailed investigation of existence and uniqueness of the solution of the considered problems, the constrained Problem 2.2.1 and the unconstrained Problem 2.2.2, for various problem settings is found in [3], [54], [89], [93], [95]. For necessary optimality conditions of first and second order and for sufficient optimality conditions see [54], [93].

2.3. Numerical solution method: Gauss-Newton

To solve the unconstrained optimization problem described in Section 2.2 we utilize a Gauss-Newton type algorithm. The Gauss-Newton algorithm iteratively approximates the solution. This is required, because the least squares functional G is in general nonlinear. In the next paragraph, we mainly follow [77, pp.245ff].

Starting from an "initial guess" p_0, the new iterate p_{k+1} of the Gauss-Newton method is

$$p_{k+1} = p_k + t_k \delta p_k, \quad 0 < t_k \leq 1,$$

where t_k is the step length and the increment $\delta p_k \in \mathbb{R}^{n_p}$ is the solution of the linearized problem at $p = p_k$

$$\min_{\delta p \in \mathbb{R}^{n_p}} \frac{1}{2} \left\| r_i(p) + J_i(p)\delta p \right\|_2^2 . \tag{2.8}$$

As before, the residuals are functions $r : P \to \mathbb{R}^m$. The Jacobian $J(p) \in \mathbb{R}^{m \times n_p}$ is defined by

$$J(p) := \frac{dr(p)}{dp} = \left[\frac{dr_i(p)}{dp_j} \right]_{\substack{i=1,\dots,m \\ j=1,\dots,n_p}} = \begin{bmatrix} \nabla_p r_1(p)^T \\ \nabla_p r_2(p)^T \\ \vdots \\ \nabla_p r_m(p)^T \end{bmatrix} .$$

The optimality condition of the linearized problem (2.8) with solution δp is:

$$J(p)^T J(p) \delta p = -J(p)^T r(p). \tag{2.9}$$

Instead of solving the standard Newton equation $\nabla^2 G(p) p = -\nabla G(p) = -J(p)^T r(p)$, the Gauss-Newton algorithm approximates the Hessian by a product of the Jacobian: $\nabla^2 G(p) \approx J(p)^T J(p)$. This is possible, because of the special structure of the least squares problem. With help of the chain rule the Hessian $\nabla^2 G(p)$ can be expressed as:

$$\nabla^2 G(p) = J(p)^T J(p) + \sum_{i=1}^{m} r_i(y(p), p) \nabla^2 r_i(p). \tag{2.10}$$

If we omit the second term on the right side, we get the approximation made above.

Local convergence of the Gauss-Newton method The local convergence of the Gauss-Newton method is proven in the *local contraction theorem* by Bock [27]. The conditions introduced there can be interpreted such that the Gauss-Newton method only converges to statistical relevant minima [27, p.72]. The convergence is linear [27], [77]. In addition, for small residuals the approximation of the Hessian by the product of the Jacobian is a good approximation. This enables a fast convergence of the Gauss-Newton method, which is almost as good as a second order convergence [89], [75], [77]. Because of these two advantages, we choose the Gauss-Newton method to solve the parameter estimation problem numerically.

The Gauss-Newton algorithm proceeds as follows: we solve the linearized problem (2.8) to compute the increment δp. As long as the gradient of the least squares functional $\nabla G = J^T r$ is non-zero and the Jacobian has full rank the increment δp is a descent direction for our Newton-type iteration [77, p.254]. Hence, we compute the new iterate p_{k+1}. After that the loop starts again, until the stopping criterion $\|\delta p\|_2 \leq tol$ is fulfilled. For the Gauss-Newton algorithm in the unconstrained case this is a suitable stopping criterion, because of

$$\delta p = -(J(p^*)^T J(p^*))^{-1} J(p^*)^T r(p^*) = 0, \tag{2.11}$$

$$\Leftrightarrow \quad J(p^*)^T r(p^*) = \frac{d}{dp}\frac{1}{2}\left\|r(p^*)\right\|^2 = 0, \tag{2.12}$$

a zero increment δp, in (2.11), is equivalent to a zero derivative of the objective function with respect to the parameters (2.12), [89]. In particular, the stopping criterion

$$\|\delta p\|_2^2 \leq tol^2 \cdot c, \tag{2.13}$$

where tol is a given tolerance and c a constant scaling factor, for example the number of variables, has been successfully approved in practice in the softwares `PAREMERA` [63] and `PARFIT` [25], [88].

To solve equation (2.9) we need to calculate the entries of the Jacobian $J(p)$ of the residuals $r(p)$. In our setting, the Jacobian is equal to the directional derivatives of the model responses $h_i(y(p), p)$ with respect to the parameters, multiplied by the weighting factor $\frac{1}{\sigma_i}$. The entries of the Jacobian are

$$\begin{aligned} J_{i,j}(p) = \frac{d}{dp_j} r_i(p) &= \frac{d}{dp_j}\left(\frac{\eta_i - h_i(p)}{\sigma_i}\right) \\ &= -\frac{1}{\sigma_i}\frac{dh_i(p)}{dp_j}. \end{aligned}$$

The main computational effort of the algorithm, besides the evaluation of the partial differential equation boundary value problem, lies in the building and evaluation of these derivatives. Especially the accurate, automatic and efficient derivation of the needed derivatives is considered in Chapters 7 and 8.

2.4. Statistical setting and sensitivity analysis

The measurement data η_i are uncertain in the way that measurement errors ε_i are introduced. Through the least squares functional G this uncertainty is embedded in the parameter estimation optimization problem. Let us take a closer look at the statistical framework. A comprehensive treatment of the topic is found in [13], [14]. A similar approach is taken in [97], [27], [66].

We first give a statistical derivation of the least squares approach. After that, we approximate the covariance matrix in order to analyze the significance of the parameter estimates. Finally, we approximate the confidence regions to get an explicit expression for the statistical uncertainty of the estimates.

Statistical derivation of the least squares approach A *probability distribution function* $f_p^i(\varepsilon_i)$ describes the uncertainty of the measurement errors $\varepsilon_i, i = 1, .., m$, given the true parameter values p^*. Let us make the first of two assumptions.
2.4.1 Assumption. The errors are statistically independent. △

With this Assumption 2.4.1 we define a *joint probability distribution function* for all errors by the product of the single probability distribution functions

$$f_p(\varepsilon) := \prod_{i=1}^{m} f_p^i(\varepsilon_i).$$

Furthermore, we model the measurements by the following relation

$$\eta_i = h_i(p^*) + \varepsilon_i,$$

the measurements consist of model response plus measurement errors. This measurement model is specified along with a joint probability distribution function $f_p(\varepsilon)$.

As we have seen before, the residuals represent the error between the measurements and the model response $r_i(p) = \eta_i - h_i(p)$. This error should be near the true measurement error, in case we are near the true values of the parameters. Thus, we insert the residuals in the joint probability distribution function and replace the errors

ε by the residuals $r(p)$

$$L(p, \varepsilon) := f_p(r(p)) = f_p(\eta - h(p)). \tag{2.14}$$

This function is called *likelihood function.* We get the *maximum likelihood estimate* by taking the maximum of this likelihood function over the parameters

$$\max_p L(p, \varepsilon). \tag{2.15}$$

This is done for given measurements η with measurement errors ε. Thus, by maximizing with respect to p we get the most likely parameters $\hat{p}$ for given measurements η.

What is missing is the shape of the probability distribution function for our errors ε. Let us make a second assumption.

2.4.2 Assumption. The errors ε_i are normally distributed with zero mean and variance σ_i^2, $\varepsilon_i \sim \mathcal{N}(0, \sigma_i^2)$. △

For the probability distribution function for the normal distribution $\mathcal{N}(0, \sigma_i^2)$ the joint probability distribution function is

$$f_p(\varepsilon) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{1}{2\sigma_i^2}\varepsilon_i^2\right). \tag{2.16}$$

If we insert the joint probability distribution function for the normal distribution (2.16) in the likelihood function (2.14), we get

$$\begin{aligned} L(p, \varepsilon) &= f_p(\eta - h(p)) \\ &= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(\eta_i - h_i(p))^2}{2\sigma_i^2}\right). \end{aligned}$$

The goal is to maximize this function $L(p, \varepsilon)$ over the parameters p. If we instead maximize the logarithm of $L(p, \varepsilon)$ over the parameters p, we obtain the same maximum for p. We use the fact that the logarithm of a product is the sum of the logarithm of the factors and get two terms. The first one is independent of p. The second one is the negative counterpart of our least squares functional. Therefore we rewrite the maximization problem as a minimization problem.

$$\begin{aligned}
\operatorname*{argmax}_p L(p,\varepsilon) &= \operatorname*{argmax}_p \log(L(p,\varepsilon)) \\
&= \operatorname*{argmax}_p \log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(\eta_i - h_i(p))^2}{2\sigma_i^2}\right) \\
&= \operatorname*{argmax}_p \sum_{i=1}^{m} \log\left(\frac{1}{\sqrt{2\pi\sigma_i^2}}\right) - \frac{1}{2}\sum_{i=1}^{m} \frac{(\eta_i - h_i(p))^2}{\sigma_i^2} \\
&= \operatorname*{argmin}_p \frac{1}{2}\sum_{i=1}^{m} \frac{(\eta_i - h_i(p))^2}{\sigma_i^2}.
\end{aligned}$$

The maximum likelihood estimation problem (2.15) is equivalent to solving the minimum least squares problem. Or in other words: our minimizing least squares problem is a maximum likelihood problem. If we solve it, we obtain an estimator $\hat{p}$, which is the most likely value of p given the measurements η with measurements errors ε.

Approximation of the covariance matrix In the next step, we want to verify the statistical significance of the estimated parameters. If the estimated parameter value is influenced strongly by insignificant variations in the data, it is ill-determined. This is expressed by a large variance of the estimate. To analyze the significance of the estimates we therefore compute the variance-covariance matrix, or shorter covariance matrix, C. The idea behind this is, if we repeat the experiments often, how would the estimates differ from one repetition to another [13]?

We examine the solution $\hat{p}$ of the Gauss-Newton algorithm. The increment δp, which is the solution of the linearized problem (2.8), is a random variable, because the measurements are random variables. The increment is determined by the optimality condition for the linearized problem (2.9). We compute the expected value of the increment, by first inserting this formulation (2.9). After some transformations we arrive at an expected value of zero

$$\begin{aligned}
E\left[\delta p\right] &= E\left[-(J^TJ)^{-1}J^Tr\right] \\
&= -(J^TJ)^{-1}J^TE\left[r\right] \\
&= -(J^TJ)^{-1}J^TE\left[\Sigma^{-1}(\eta - h)\right] \\
&= -(J^TJ)^{-1}J^T0
\end{aligned}$$

$$= 0,$$

where $\Sigma := \operatorname{diag}(\sigma_i, i = 1, .., m)$. For clarity, all arguments are omitted here.

For the expected value of the product of the residuals we have

$$\begin{aligned} E\left[rr^T\right] &= E\left[\Sigma^{-1}(\eta - h)(\eta - h)^T \Sigma^{-1}\right] \\ &= \Sigma^{-1} E\left[\eta\eta^T\right] \Sigma^{-1} = \Sigma^{-1}\Sigma^2\Sigma^{-1} = I. \end{aligned}$$

With that, we get for the covariance matrix, which is defined as the expected value of a product of increments,

$$\begin{aligned} C &:= E\left[\delta p \delta p^T\right] \\ &= E\left[(-(J^TJ)^{-1}J^Tr)(-(J^TJ)^{-1}J^Tr)^T\right] \\ &= E\left[(J^TJ)^{-1}J^Trr^TJ(J^TJ)^{-1}\right] \\ &= (J^TJ)^{-1}J^TE\left[rr^T\right]J(J^TJ)^{-1} \\ &= (J^TJ)^{-1}J^TJ(J^TJ)^{-1} \\ &= (J^TJ)^{-1}. \end{aligned} \tag{2.17}$$

We arrive at a term, which is only dependent on the Jacobi matrix (2.17). Thus we are able to compute the covariance matrix by evaluating the Jacobi matrix.

2.4.3 Remark. Another way to arrive at this representation of the covariance matrix is to use a Taylor expansion of the optimality condition of the reduced parameter estimation problem instead of using the linearized problem of the Gauss-Newton algorithm. Retaining only terms of first order and applying these terms to compute the covariance matrix leads to representation (2.17), see [13], [33], [97]. △

Confidence region For a more explicit expression of the significance of the parameter estimation, we employ the concept of confidence regions.

A nonlinear confidence region for the true parameter p^* in our setting is defined by [97]

$$G_N(\alpha_r, p) := \{p \in \mathbb{R}^{n_p} : \|r(p)\|^2 - \|r(p^*)\|^2 \leq \gamma_r^2(\alpha_r)\}, \tag{2.18}$$

where $\gamma_r^2(\alpha_r)$ is the quantile of the χ^2 distribution for value $\alpha_r \in [0, 1]$ and n_p degrees of freedom. That means, that with probability $1 - \alpha_r$ the true parameter values p^*

lie inside the nonlinear confidence region.

Because we do not know the true parameter values p^*, we perform the computation with the estimated parameters $\hat{p}$. If we would repeat the experiments one hundred times, each experiment would produce a different estimate $\hat{p}$. For each such estimate we could build the confidence region (2.18). Then the values of the true parameters p^* should be included in about $(1 - \alpha_r) \cdot 100$ of these confidence regions. For example $\alpha_r = 0.1$, then the true parameter values should be included in ninety of these confidence regions [13].

We approximate the nonlinear confidence region by first linearizing it and after that showing that the linearized confidence region is part of a cuboid. A linearized confidence region reads

$$G_L(\alpha_r, p) := \{p \in \mathbb{R}^{n_p} : \|r(p^*) + J(p^*)(p - p^*)\|^2 - \|r(p^*)\|^2 \leq \gamma_r^2(\alpha_r)\}.$$

The residual $r(p^*)$ vanishes, thus it follows

$$\begin{aligned} G_L(\alpha_r, p) &= \{p \in \mathbb{R}^{n_p} : (p - p^*)^T (J^T J)(p - p^*) \leq \gamma_r^2(\alpha_r)\} \\ &= \{p \in \mathbb{R}^{n_p} : (p - p^*)^T C^{-1}(p - p^*) \leq \gamma_r^2(\alpha_r)\}. \end{aligned}$$

Note, that the inverse of the covariance matrix C^{-1} is a core part of the formula for the linearized confidence region. Instead of the unknown true parameter values p^*, we approximate the confidence region with the estimated parameters $\hat{p}$

$$G_L(\alpha_r, p) \approx \{p \in \mathbb{R}^{n_p} : (p - \hat{p})^T C^{-1}(p - \hat{p}) \leq \gamma_r^2(\alpha_r)\}. \tag{2.19}$$

In the next step, let us show that the linearized confidence region $G_L(\alpha_r, p)$ is part of a cuboid, whose side lengths are dependent on the diagonal elements of the covariance matrix C.

2.4.4 Lemma. *Let*

$$\theta_i := \gamma_r(\alpha_r)\sqrt{C_{ii}}, \quad i = 1, ..., n_p$$

with C_{ii} the diagonal elements of the covariance matrix C. Then the linearized confidence region is part of a cuboid

$$G_L(\alpha_r, p) \subset [\hat{p}_1 - \theta_1, \hat{p}_1 + \theta_1] \times \ldots \times [\hat{p}_{n_p} - \theta_{n_p}, \hat{p}_{n_p} + \theta_{n_p}].$$

Proof. The definition of the linearized confidence region $G_L(\alpha_r, p)$ contains the inverse of the covariance matrix

$$(p - \hat{p})^T C^{-1} (p - \hat{p}) \leq \gamma_r^2(\alpha_r).$$

We treat the inequality component-wise and apply transformations

$$\begin{aligned} (p_i - \hat{p}_i)^T C_{ii}^{-1} (p_i - \hat{p}_i) &\leq \gamma_r^2(\alpha_r) \\ \Leftrightarrow \quad (p_i - \hat{p}_i)^2 \qquad &\leq \gamma_r^2(\alpha_r) C_{ii} \\ \Leftrightarrow \quad |p_i - \hat{p}_i| \qquad &\leq \gamma_r(\alpha_r)\sqrt{C_{ii}} = \theta_i, \end{aligned}$$

which lead to the claim that the linearized confidence region is part of a cuboid with side length $2 \cdot \theta_i$. □

The significance of the estimated parameters is expressed by the “size” of the confidence region for given α_r. The “smaller” the confidence region, the more significant are the estimated parameters. As we have seen, the confidence regions can be approximated with the help of the covariance matrix. We will utilize this relation in nonlinear optimum experimental design in the next Chapter 3 by assigning a scalar value to the “size” of the confidence region. For that, we will use an information function or criterion, which operates on the covariance matrix.

3. Optimum experimental design with PDE models

The present chapter deals with the optimum experimental design problem for parameter estimation with PDE models. First the optimum experimental design (OED) problem, specifically the sampling design problem, is presented. We first depict the measurement design. After that we present the OED problem formulation. Thereafter we relax the integer constraints for the numerical solution of the OED problem. Then we present two problem variants. After that the optimality conditions for the relaxed OED problem are formulated. Afterwards a numerical solution method for the OED problem based on sequential quadratic programming (SQP) methods is depicted.

A similar OED setting for ordinary differential equations is presented in [66], [96] and [56]. For PDEs, [97] and [63] investigate a similar setting. We follow them in most points in this chapter.

3.1. Measurement design

Measurement grid A finite grid of possible measurement points

$$x_1^m, x_2^m, ..., x_{n_m}^m \in \Omega \tag{3.1}$$

is constructed, see Figure 3.1, where n_m is the number of possible measurement grid points. The point measurements at these grid points are incorporated in our measurement functions

$$h_i(p; x_j^m) = \bar{h}_i(p; S(p); x_j^m), \quad i = 1, .., n_g, j = 1, ..., n_m,$$

with n_g the number of possible measurement functions. Every measurement function describes a point measurement at a measurement point. We assume the function y to have sufficiently many weak derivatives, such that the state space $Y = \mathcal{H}^k(\Omega)$ embeds in continuous functions. With that, the point measurements are well-defined.

The measurement points, which exhibit the most information for the parameter estimation, are chosen by the optimum experimental design algorithm. It is possible, that two measurement functions h_1 and h_2 are available at the same measurement point x_j^m: $h_1(p; x_j^m)$, $h_2(p; x_j^m)$. The construction of the grid may be prescribed by the measurement methods present or by the process under investigation.

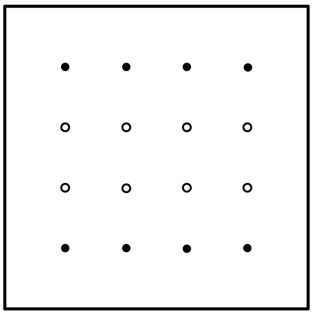


Figure 3.1.: Setup sampling design. The picture shows an example domain Ω with circles for possible measurement points and points for realized measurement points.

Sampling decisions For the algorithmic selection of measurement points we introduce a vector of sampling decisions $w = (w_1, ..., w_{n_g})^T$. The sampling decisions are assumed to be integer variables $w_i \in \{0, 1\}, i = 1, ..., n_g$. If the sampling decision w_i is zero, the measurement is not realized in measurement function $h_i(p; x_j^m)$ in an optimum experimental design and if the sampling decision w_i is one, the measurement is realized. Out of n_g possible measurements those measurements are selected, which contain the most information for the parameter estimation.

Inequality constraints The number of measurements to select is constrained by an upper bound m. The sum of all sampling decisions should be below the bound

$$\sum_{i=1}^{n_g} w_i \leq m.$$

This guarantees that only m measurements are chosen by our algorithm. Additional constraints on the measurement design, for example a maximum number of measurements per measurement method, can be formulated by linear functions of w_i.

Another possibility to constrain the number of measurements is via costs. For example we have costs c_i per measurement i. With a linear cost model we get

$$\sum_{i=1}^{n_g} c_i w_i \leq m_c,$$

where m_c is the maximum costs we would like to spend. This concept is flexible with regard to different costs for different measurement points or different costs for different measurement methods. The constants c_i are set before the optimization, the sampling decisions w_i are the free variables.

We combine all inequality constraints, which depict the measurement design, in the linear constraint function c_m with

$$0 \leq c_m(w).$$

Variances We weight the variances σ_i of the measurement errors with the sampling decisions w_i. Thus the measurement errors are normally distributed with variances $\frac{\sigma_i^2}{w_i}$, i.e. $\varepsilon_i \sim N(0, \frac{\sigma_i^2}{w_i})$.

PE problem for OED Let us state the parameter estimation problem, which underlies the optimum experimental design problem.

3.1.1 Problem. ***(Unconstrained parameter estimation problem for OED)*** *Minimize the reduced least squares functional $G(p)$*

$$\min_{p \in P} G(p) = \frac{1}{2} \sum_{i=1}^{n_g} w_i \left(\frac{\eta_i - h_i(p)}{\sigma_i} \right)^2 = \frac{1}{2} \sum_{i=1}^{n_g} r_i^2(p).$$

As before, the solution of the weak form of the PDE boundary value problem enters the Problem 3.1.1 via the solution operator $S(p)$, where $G(p) = \bar{G}(p; S(p))$.

Notice the slight difference to the preceding parameter estimation Problem 2.2.2: variance is $\frac{\sigma_i^2}{w_i}$ instead of σ_i^2 before. This way the sampling decisions w_i enter the parameter estimation problem. Because of that the residuals change as well to

$$r_i(p) = \frac{\eta_i - h_i(p)}{\frac{\sigma_i}{\sqrt{w_i}}} = \sqrt{w_i} \frac{\eta_i - h_i(p)}{\sigma_i}.$$

Additionally the sum adds the set of all possible measurement functions $i = 1, ..., n_g$ instead of m functions before. The selection of m functions is done via the sampling decisions w_i. Only the number of m sampling decisions is equal to 1, the remainder is equal to 0. The sampling decisions w_i are fixed during the parameter estimation. Contrary the parameters p are fixed during the optimum experimental design.

3.2. OED problem formulation

Criteria The goal of optimum experimental design is to improve the statistical significance of the estimated parameters. The ”size” of a confidence region expresses the quality of the estimated parameters. The ”smaller” the region, the more significant are the estimated parameters. A question arises: How to measure the ”size” of the confidence region? We will make use of information functions [81], which are in the context of OED also called *criteria*.

The classical optimum experimental design criteria operate on the covariance matrix C of the parameter estimation problem. Let us introduce three classical optimum experimental design criteria [81, p.135ff]

- *average-variance criterion (A-criterion)*

$$\Phi_A(C) := \frac{1}{n_p}\,\mathrm{tr}(C),$$

- *determinant criterion (D-criterion)*

$$\Phi_D(C) := \det(C)^{\frac{1}{n_p}},$$

- *smallest-eigenvalue criterion (E-criterion)*

$$\Phi_E(C) := \max\{\lambda_i | \lambda_i \text{ eigenvalue of } C\} = \|C\|_2,$$

and one additional criterion, indroduced by [75],

- *confidence region criterion (M-criterion)*

$$\Phi_M(C) := \max\{\sqrt{C_{ii}}, i = 1, ..., n_p\}.$$

In Figure 3.2 a geometrical interpretation of these four criteria in comparison to the confidence region (2.19), or confidence ellipsoid in two dimensions, is shown. The

A-criterion is proportional to the average half-axis length of the confidence ellipsoid. The D-criterion can be visualized by the volume of the confidence ellipsoid. The E-criterion is proportional to the largest half-axis length of the confidence ellipsoid. The M-criterion can be depicted by a box around the confidence ellipsoid, we have shown this connection in Lemma 2.4.4. By minimizing one of the criteria, we minimize the confidence region [66], [15], [96].

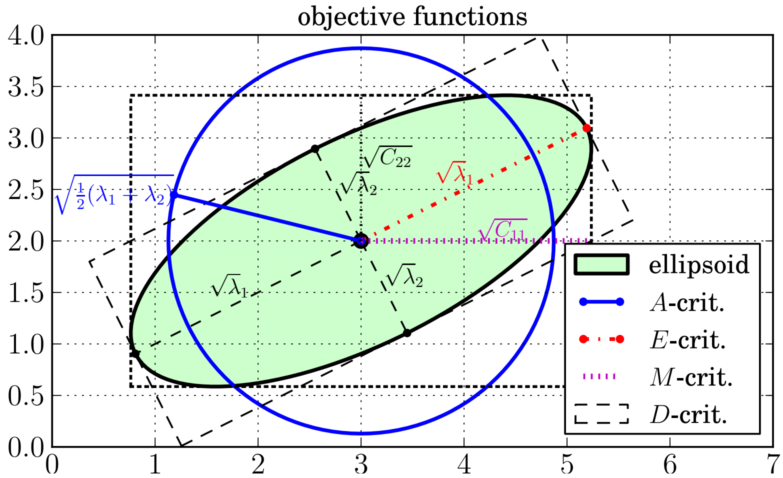


Figure 3.2.: Confidence ellipsoid and geometrical interpretation of A-, D-, E- and M-criteria, taken from [96].

Other criteria based on the Fisher information matrix, which is the inverse of our covariance matrix C, are also possible. In that case, we get a maximization problem, see [81], [87].

A different idea is the minimization of a *key performance indicator* also referred to as *quantity of interest*. Instead of minimizing a function of the covariance matrix of the parameter estimation problem, another important indicator in the problem formulation is minimized. That means, not the significance of the parameter estimation is optimized, but an important output of the model. For OED with DAE and ODE this idea was first introduced by [67]. In [72], [73] OED and optimal control objectives are combined for ODE models. A user defined interest functional is presented in [22] for model calibration with a PDE model.

3.2.1 Remark. *Scaling.* The A-, D-, E- and M-criteria, which we will use in this work, are not invariant to the size of the parameters. A difference in magnitude of the individual parameters leads to a unilateral preference of the parameters with the biggest absolute value. A solution to this problem is scaling. In most cases the parameters are all scaled to one. Parameters, which are considered more important, could be scaled to a higher value than the other parameters. That way, these parameters are estimated with higher accuracy than the rest [15], [66]. △

OED problem Let us formulate the OED problem. The OED or sampling design problem is a constraint nonlinear optimization problem. It consists in minimizing criterion $\Phi(C(w,p))$, under constraints on the sampling decisions w_i:

3.2.2 Problem. ***(Optimum experimental design problem)***

$$\min_{w} \Phi(C(w,p))$$

subject to

$$\begin{aligned} 0 &\leq c_m(w), & \\ w_i &\in \{0,1\}, \qquad & i = 1, ..., n_g. \end{aligned}$$

Every sampling decision w_i corresponds to one potential measurement function $h_i(p; x_j^m), i = 1, .., n_g$, which describes a point measurement at a spatial measurement point $x_j^m, j = 1, ..., n_m$. Finding the minimizing sampling decisions w_i leads to a selection of measurement functions $h_s(p; x_j^m), s = 1, .., m$, and thus a selection of spatial measurement points.

As defined before the objective function of Problem 3.2.2, i.e. the optimum experimental design criterion, is dependent on the covariance matrix C

$$C = (J^T J)^{-1}$$

in the solution point of the underlying reduced parameter estimation Problem 3.1.1. The entries of the Jacobian J are

$$J_{i,j}(p) = \frac{d}{dp_j} r_i(p) = -\frac{\sqrt{w_i}}{\sigma_i} \frac{dh_i(p)}{dp_j}.$$

The solution of the PDE model enters here via the solution operator $S(p)$, according to the definition of the reduced residuals $r(p) = \bar{r}(p; S(p))$ and the reduced measurement functions $h(p) = \bar{h}(p; S(p))$, see Section 2.2.

Regarding the evaluation or generation of the Jacobian J there exist different possibilities. These possibilities will be investigated in detail later in the separate Chapters 5, 7 and 8.

3.2.3 Remark. *Parameter dependence.* The covariance matrix C depends on the parameters p. Usually we evaluate the covariance matrix in the solution point of the parameter estimation Problem 3.1.1. Thus it is dependent on the parameter estimate and indirectly also on the measurements η_i. Solution strategies to take into account bad parameter estimates are *sequential OED* and *robust OED*, for details see the following Section 3.4. △

The sampling design problem investigated in this thesis is a special case of the general optimum experimental design problem. The optimization variables only enter the least squares functional of the parameter estimation problem, not additionally the partial differential equation boundary value problem.

3.3. Relaxation of integer constraints

Relaxed OED problem For the numerical solution of the optimum experimental design problem, we relax the mixed-integer constraint $w \in \{0, 1\}$ to

$$
\begin{aligned}
& w \in conv(\{0,1\}), && \text{(3.2a)}\\
\Leftrightarrow \quad & 0 \leq w_i \leq 1, \qquad i = 1, ..., n_g, && \text{(3.2b)}
\end{aligned}
$$

where $conv(.)$ is the convex hull. The formulation (3.2) is in contrast to the mixed-integer variant continuous. The resulting relaxed optimum experimental design problem reads

3.3.1 Problem. ***(Relaxed optimum experimental design problem)***

$$\min_{w} \Phi(C(w,p))$$

subject to

$$
\begin{aligned}
& 0 \leq c_m(w) \\
& 0 \leq w_i \leq 1, \qquad i = 1, ..., n_g.
\end{aligned}
$$

An important question is, if the solution of the relaxed OED Problem 3.3.1, with the relaxed sampling decisions formulation (3.2) as a constraint, is a reasonable solution for the OED problem with the mixed-integer constraint, Problem 3.2.2.

There are two possibilities to obtain an integer solution from a fractional solution: use a rounding strategy [16], [66] or refine the possible measurement grid. As a rounding heuristic we could use a *simple round up and off strategy*, that means round up the biggest sampling decisions and round off the smallest ones, keeping the sum of all sampling decisions equal or below the maximum number m. Another possibility is the *sum up rounding strategy*, where the sampling decisions are summed up until the sum reaches one. This strategy is used in time-dependent problems. We could adapt it to our spatial setting. In [87] it is showed, that using the sum up rounding strategy is possible. But it could have a so-called chattering behavior, that means switching often between yes and no. That is, why the authors in [87] recommend to refine the possible measurement grid.

Another point investigated in [87] is the ill-posedness of the OED problem, if the maximum number of measurements m is set too high. Additional measurements contribute little to the minimization of the objective function, because the placed measurements are already placed optimal. As a solution they propose a $\mathcal{L}^1$ penalization in the objective function, which couples the cost of a measurement to a minimum amount of information it has to contribute. Another possible solution is a sequential or greedy placement of the measurement points [51].

Regarding the existence and uniqueness of a solution of the mixed-integer constrained OED Problem 3.2.2 and of the relaxed OED Problem 3.3.1, to our knowledge results only have been proven for DAE and ODE constraints [16], [56], [66], [75], [96]. For PDE constraint OED problems, in [76] a unconstrained version of the OED problem is investigated. In this thesis we do not answer the open question of existence and uniqueness and assume, that a unique solution to both problems exists.

3.4. Problem variants

Parameter dependence As addressed in Remark 3.2.3 the covariance matrix and thereby the objective function of the OED problem, is dependent on the parameter estimate $\hat{p}$. To decrease this influence, two solution approaches are possible, *sequential OED* and *robust OED*. Let us sketch them briefly.

In *sequential OED* a cycle of practical experiments, parameter estimation and optimum experimental design is performed. The estimates are enhanced by performing new optimized experiments in practice. The OED objective is no longer dependent on one set of measurements, but on multiple ones, which are performed in different experimental settings. For further details, see [15], [16], [66], [68].

The idea of *robust OED* is to take a closer look at the worst case scenario. Namely the parameters are estimated extremely poor. A min-max optimization problem is considered: the minimum of the maximum value of the objective function $\Phi(C)$ over the confidence region is computed. Fur further details, see [28], [69].

Multiple experiments Instead of performing one single experiment, we could perform multiple experiments. From multiple experiments we gain more information, which we can use to estimate the unknown parameters. Especially when applying OED to optimize the experimental setting, the optimized experiment should be performed in practice. With the new measurements from the optimized experiment we once again estimate the parameters.

All methods, which we present and discuss in this thesis, are also applicable and available for the multiple experiment setup. For details on PE with multiple experiments see [63], [64], [65], [88] and for OED with multiple experiments see [56], [57], [66]. In the following we present the case for a single experiment.

3.5. Optimality conditions

In this section we follow [77, chapter 12] and [97, section 4.3]. We set up the optimality conditions for the relaxed OED Problem 3.3.1.

We reformulate the relaxed OED Problem 3.3.1 as a general constrained optimization problem. Therefore we distinguish equality and inequality constraints. We aggregate each of them in index sets, index set $\mathcal{E}$ for the equality constraints and $\mathcal{I}$ for the inequality constraints.

3.5.1 Problem. ***(Relaxed optimum experimental design problem revisited)***

$$\min_{w} \Phi(C(w,p))$$

subject to

$$\begin{aligned} c_j(w) &= 0, \quad j \in \mathcal{E}, \\ c_j(w) &\geq 0, \quad j \in \mathcal{I}. \end{aligned}$$

We need this distinction between equality and inequality constraints, because the inequality constraints need special treatment in the setting up of optimality conditions.

Therefore we use the concept of active and inactive inequality constraints and define the *active set* $\mathcal{A}(w)$ as follows.

3.5.2 Definition. The *active set* $\mathcal{A}(w)$ at any feasible w contains the indices of the equality index set $\mathcal{E}$ and the indices of the inequality constraints, which take equality in w

$$\mathcal{A}(w) := \mathcal{E} \cup \{j \in \mathcal{I} | c_j(w) = 0\}.$$

The inequality constraint $j \in \mathcal{I}$ is said to be *active* at w, if $c_j(w) = 0$. △

Let us define a condition on the gradients of the constraints c_j, which we will need to set up the optimality conditions.

3.5.3 Definition. (LICQ) Given a vector of sampling decisions w and the corresponding active set $\mathcal{A}(w)$, Definition 3.5.2, the linear independence constraint qualification (LICQ) holds, if the set

$$\{\nabla c_j(w), j \in \mathcal{A}(w)\}$$

is linearly independent. △

Furthermore the *Lagrange function* $L(w, l)$ of the relaxed OED Problem 3.5.1 reads

$$L(w, l) := \Phi(C(w, p)) - \sum_{j \in \mathcal{E} \cup \mathcal{I}} l_j c_j(w),$$

with the vector l of *Lagrange multipliers* $l_j, j \in \mathcal{E} \cup \mathcal{I}$. With these ingredients, the necessary optimality conditions for the relaxed OED Problem 3.5.1 are

3.5.4 Lemma. *Necessary optimality conditions (Karush-Kuhn-Tucker conditions) Let w^* be a local solution of Problem 3.5.1, the functions $\Phi(w, p)$ and $c_j(w)$ be continuously differentiable and let LICQ in Definition 3.5.3 hold at w^*. Then there exists a Lagrange multiplier l^*, with components $l_j^*, j \in \mathcal{E} \cup \mathcal{I}$, such that the following conditions are satisfied at (w^*, l^*)*

$$\begin{aligned} \nabla_w L(w^*, l^*) &= 0, \\ c_j(w^*) &= 0, \quad \textit{for all } j \in \mathcal{E}, \\ c_j(w^*) &\geq 0, \quad \textit{for all } j \in \mathcal{I}, \\ l_j^* &\geq 0, \quad \textit{for all } j \in \mathcal{I}, \\ l_j^* c_j(w^*) &= 0, \quad \textit{for all } j \in \mathcal{E} \cup \mathcal{I}. \end{aligned}$$

Proof. A proof can be found in [77, section 12.4]. □

3.6. Numerical solution methods: Sequential quadratic programming

In this section we follow [77, chapter 18] and [97, pp.67ff]. To solve the nonlinear problem with equality and inequality constraints, the relaxed OED Problem 3.5.1, effectively, we approximate it by a quadratic program. Iteratively a sequence of constrained quadratic subproblems of the form

$$\begin{aligned} \min_{\delta w} \quad & \frac{1}{2}\delta w^T H(w,l)\delta w + \nabla \Phi(w,p)^T \delta w \\ \text{s.t.} \quad & \nabla c_j(w)^T \delta w + c_j(w) = 0, \qquad j \in \mathcal{E}, \\ & \nabla c_j(w)^T \delta w + c_j(w) \geq 0, \qquad j \in \mathcal{I}, \end{aligned} \tag{3.3}$$

are solved. The objective function of the subproblem (3.3) is composed of a approximation of the Hessian of the Lagrange function with respect to sampling decisions w

$$H(w,l) \approx \nabla_w^2 L(w,l),$$

and the gradient of the objective function of the relaxed OED Problem 3.5.1. The constraints are the linearized constraints of the relaxed OED Problem 3.5.1. We minimize the quadratic subproblem (3.3) with respect to search direction δw.

In Algorithm 1 the complete sequential quadratic programming (SQP) algorithm is presented. First, start values are set, functions and gradients of the objective function $\Phi(C(w,p))$ and the constraints $c_j(w)$ are evaluated. The Jacobian of the PE Problem 3.1.1 is computed for a fixed parameter value p. This value p could originate from a preceding parameter estimation. A convergence test is performed before each iteration, to check if we can stop the algorithm. After that the search direction is obtained by solving the quadratic subproblem (3.3) with the approximation of the Hessian of the Lagrange function. The next step is to find a step size and finally we update the iterates and compute the function and gradient values of the objective function $\Phi(C(w,p))$ and the constraints $c_j(w)$ for the new iterate.

There are many variants of SQP methods: they differ in the choice of the Hessian approximation, the approximation of the quadratic subproblems, the step size computation and the choice of the convergence test.

Computing the Hessian of the Lagrange function is computationally expensive. Therefore we approximate it. One possibility is a quasi-Newton approximation, which

Algorithm 1 SQP algorithm for relaxed OED problem

```
1: Start values $w_0, l_0, H_0$. Set $k = 0$.
2: Evaluate $\Phi(C(w_0,p)), \nabla\Phi(C(w_0,p)), c_j(w_0), \nabla c_j(w_0), j \in \mathcal{E} \cup \mathcal{I}$
   and the Jacobian $J(p)$, with value $p$ fixed.
3: Convergence test, if satisfied stop, if not satisfied go to 4.
4: Compute search direction
   if k > 0 then
   Compute an approximation of the Hessian of the Lagrange function
       $H_k(w_k, l_k) \approx \nabla^2_w \mathcal{L}(w_k, l_k).$
   end if
   Solve
       $\min_{\delta w} \quad \frac{1}{2}\delta w^T H^k(w_k, l_k)\delta w + \nabla\Phi(w_k,p)^T \delta w$
       $\text{s.t.} \quad \nabla c_j(w_k)^T \delta w + c_j(w_k) = 0, \qquad j \in \mathcal{E},$
       $\qquad\quad \nabla c_j(w_k)^T \delta w + c_j(w_k) \geq 0, \qquad j \in \mathcal{I},$
   obtain $\delta w_k$ search direction and $\tilde{l}_k$ Lagrange multiplier. Go to 5.
5: Find step size $\alpha_k$. Go to 6.
6: Iterate
       $w_{k+1} := w_k + \alpha_k \, \delta w_k,$
       $l_{k+1} := l_k + \alpha_k \, (\tilde{l}_k - l_k).$                    (3.4)
   Evaluate
       $\Phi(C(w_{k+1},p)), \quad \nabla\Phi(C(w_{k+1},p)), \quad c_j(w_{k+1}), \quad \nabla c_j(w_{k+1}), \quad j \in \mathcal{E} \cup \mathcal{I},$
   $k := k + 1$, go to 3.
```

means the use of Broyden-Fletcher-Goldfarb-Shanno (BFGS) or symmetric rank one (SR1) update formulae. For the approximation of the quadratic subproblems (3.3) active set or interior point methods can be utilized. We will use a active set strategy [56], [77]. For choice of step size and convergence test we refer to literature [56], [77]. As a stopping criterion the Karush-Kuhn-Tucker conditions in Lemma 3.5.4 are eligible.

Specially in the utilized software in `SNOPT` [43], [42] a limited memory BFGS is implemented, where former iterates are ignored occasionally. In `blockSQP` [56] partitioned quasi-Newton updates are implemented, including SR1 and BFGS updates.

Local convergence SQP The convergence properties of a SQP method are mostly dependent on the choice of the approximation of the Hessian of the Lagrangian. With an exact computation of the Hessian, the SQP method is equivalent to Newton's method with local quadratic convergence. When approximating the Hessian with a BFGS update, under some additional conditions the local convergence rate is superlinear. For global convergence, the choice of the step size is crucial. Hence for proofs and further investigation of the convergence properties see [77].

Part II.

Status quo

4. Discontinuous Galerkin finite element methods

In this chapter, we focus on the discretization of the underlying PDE boundary value problem, which is a stationary 2D diffusion advection reaction PDE boundary value problem, see Example 2.1.5. We are particularly interested in an advection dominated case, with a Péclet number much larger than 1. For a definition of Péclet number number see Remark 2.1.6. This leads to a symmetry loss, which in turn results in a coercivity loss. Therefore, the solutions of continuous Galerkin methods oscillate [38, p.166]. A standard finite element method is not suitable for this problem [58]. Hence, we use a discontinuous Galerkin method. We discretize the diffusion advection reaction PDE boundary value problem by two discontinuous Galerkin methods: the advection part is discretized by the upwind method [84] and the diffusion part by the interior penalty method [9], [10]. For the reaction part an additional mass matrix is added.

In this chapter, we first present the standard upwind discontinuous Galerkin method for the advection part. In Chapter 6 we extend this method to a differentiable version. After that, we state a discretization for the diffusion advection reaction PDE model. With this dicretization for the PDE model, we formulate discrete optimization problems. Finally, we give a short overview over the finite element algorithm. We start with notation and definitions.

Grid cells and faces The convex domain Ω in $\mathbb{R}^d, d = 2$, with boundary $\mathit{\Gamma}$, is subdivided into a triangulation $\mathbb{T}_h$. The triangulation $\mathbb{T}_h$ consists of closed quadrilateral grid cells T. The set of closed faces F of the grid cells is denoted by $\mathbb{F}_h$. We define the subsets of boundary faces and interior faces of $\mathbb{F}_h$, respectively:

$$\mathbb{F}_h^{\mathit{\Gamma}} := \{F \in \mathbb{F}_h | F \in \mathit{\Gamma}\},$$
$$\mathbb{F}_h^{int} := \{F \in \mathbb{F}_h | F \notin \mathit{\Gamma}\}.$$

With that, we set $\mathbb{F}_h = \mathbb{F}_h^{\mathit{\Gamma}} \cup \mathbb{F}_h^{int}$. The index h stands for the mesh size function $h_T := diam(T) = \max_{x_1,x_2 \in T} \|x_1 - x_2\|, T \in \mathbb{T}_h$. The mesh size function represents a level of refinement of the grid. For the overall triangulation, it is defined by $h := \max_{T \in \mathbb{T}_h} h_T$.

Jumps and Averages Let the values y_1 and y_2 denote the traces of the function y on F taken from T_1 and T_2, respectively. On an interior face F separating the two cells T_1 and T_2, we define the average operator $\{\!\{\cdot\}\!\}$ as follows

$$\{\!\{y\}\!\}(x) := \frac{1}{2}\Big(y_1(x) + y_2(x)\Big).$$

Let $n = n(x)$ be the outward unit normal vector and n_1 and n_2 the corresponding outward unit normal vectors for T_1 and T_2. Then, the term

$$2\{\!\{yn\}\!\} = y_1 \cdot n_1 + y_2 \cdot n_2 = (y_1 - y_2)n_1 = (y_2 - y_1)n_2, \tag{4.1}$$

defines a vector valued jump operator on y in a fashion that is oblivious to the choice of T_1 and T_2. Additionally, we define the short hand notation for a jump operator $[\![\cdot]\!]$ as follows

$$[\![y]\!]\,(x) = y_1(x) - y_2(x),$$

which will only be used squared, such that its sign does not matter. In particular, we will use this short form for the term

$$\Big([\![y]\!]\,, [\![v]\!]\Big)_F = 4\Big(\{\!\{yn\}\!\}, \{\!\{vn\}\!\}\Big)_F. \tag{4.2}$$

In the same way as before $(y, v)_F$ is the $\mathcal{L}^2(F)$ scalar product on a face F: $(y, v)_F = \int_F yvds$.

Norms and spaces The Lebesgue space $\mathcal{L}^p(\Omega)$ and the Sobolev space $\mathcal{W}_p^k(\Omega)$ are defined in Chapter 2, Section 2.1. For the discontinuous Galerkin discretization, we additionally need the *broken Sobolev space* on the triangulation $\mathbb{T}_h$. It consists of functions v that belong to the Sobolev space $\mathcal{W}_p^k(T)$ for each grid cell T.

4.0.1 Definition. Thus, the *broken Sobolev space* $\mathcal{W}_p^k(\mathbb{T}_h)$ is defined by [58]

$$\mathcal{W}_p^k(\mathbb{T}_h) := \{v \in \mathcal{L}^p(\Omega)|\ v|_T \in \mathcal{W}_p^k(T), T \in \mathbb{T}_h\},$$

with corresponding *broken Sobolev norm* defined by

$$\|v\|_{\mathcal{W}_p^k(\mathbb{T}_h)} := \left(\sum_{T\in\mathbb{T}_h} \|v\|^p_{\mathcal{W}_p^k(T)}\right)^{\frac{1}{p}}.$$

△

For $p = 2$, we similarly abbreviate $\mathcal{H}^k(\mathbb{T}_h) := \mathcal{W}_2^k(\mathbb{T}_h)$.

4.0.2 Definition. Let us define the *finite element space* V_h. Limited to each grid cell T the function v_h belongs to the space of polynomials

$$V_h := \{v_h \in \mathcal{L}^2(\Omega) : v_h|_T \in \mathcal{P}(T), T \in \mathbb{T}_h\}.$$

Where $\mathcal{P}(T)$ is a polynomial space on T, for instance the space of polynomials of degree $k \geq 0$ with no continuity requirements across interelement boundaries [58]. △

4.0.3 Definition. We define the *reference element* $\hat{T}$ with vertices $(-1,-1)$, $(1,-1)$, $(1,1)$ and $(-1,1)$. For each physical element T exists a mapping M from the reference element to the physical element

$$\hat{v}_h = v_h \circ M.$$

If T is a parallelogram, the map M is affine. △

We perform all computations on the reference element. Hence, we use the finite element space on the reference element $\hat{V}_h$.

4.0.4 Definition. *Finite element space on the reference element.* Limited to each grid cell T the function v_h belongs to the space of polynomials

$$\hat{V}_h := \{v_h \in \mathcal{L}^2(\Omega) : v_h \circ M \in \mathcal{P}(\hat{T}), T \in \mathbb{T}_h\}.$$

Where $\mathcal{P}(\hat{T})$ is a space of polynomials on the reference element $\hat{T}$. △

For parallelograms in 2D, the finite element space on the reference element has the same approximation properties as the finite element space on the physical element from Definition 4.0.2 [85].

To simplify the notation we omit the dependency on the parameters p in this chapter.

4.1. Upwind method for advection problems

We begin by discretizing the advection part. The pure advection model problem Example 2.1.7 reads

4.1.1 Example. *Pure advection model problem revisited.*

$$\beta \cdot \nabla y = f \qquad \text{on } \Omega,$$

$$y = y_D \quad \text{on } \Gamma_-,$$

for $\|\beta\| \neq 0$. As before we have the inflow boundary $\Gamma_- = \{x \in \partial\Omega | n(x) \cdot \beta < 0\}$, an inflow boundary function $y_D \in \mathcal{L}^2(\Gamma_-)$ and a right hand side function $f \in \mathcal{L}^2(\Omega)$. △

For this pure advection model problem the upwind discontinuous Galerkin discretization is: Find $y_h \in V_h$ such that

$$b_h(y_h, v_h) = f_h^a(v_h), \quad \forall v_h \in V_h, \tag{4.4a}$$

where

$$\begin{aligned} b_h(y_h,v_h) :=& - \sum_{T \in \mathbb{T}_h} (y_h, \beta \cdot \nabla v_h)_T + \sum_{F \in \mathbb{F}_h^\Gamma} \left(\frac{1}{2}\beta \cdot n y_h + \frac{1}{2}\sigma_{upw}(\beta, n) y_h, v_h \right)_F \\ &+ \sum_{F \in \mathbb{F}_h^{int}} (\{\!\{y_h\}\!\}, 2\{\!\{v_h \beta \cdot n\}\!\})_F + \sum_{F \in \mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_{upw}(\beta, n) [\![y_h]\!], [\![v_h]\!] \right)_F, \end{aligned} \tag{4.4b}$$

and right hand side

$$f_h^a(v_h) := \sum_{T \in \mathbb{T}_h} (f, v_h)_T - \sum_{F \in \mathbb{F}_h^\Gamma} \left(\frac{1}{2}\beta \cdot n y_D - \frac{1}{2}\sigma_{upw}(\beta, n) y_D, v_h \right)_F, \tag{4.4c}$$

with stabilization function

$$\sigma_{upw}(\beta, n) := |\beta \cdot n| \,. \tag{4.5}$$

Instead of two separate terms for inflow and outflow element boundaries, we choose a flux function [32], which selects inflow or outflow element boundary by addition and subtraction

$$H(y_1, y_2, n) := \beta \cdot n \{\!\{y\}\!\} + \frac{1}{2}\sigma_{upw}(\beta, n) [\![y]\!] \,.$$

We see, that this numerical flux function can also be written as

$$H(y_1, y_2, n) = \begin{cases} \beta \cdot n \; y_2, & if \;\; \beta \cdot n(x) < 0, \\ \beta \cdot n \; y_1, & if \;\; \beta \cdot n(x) \geq 0. \end{cases}$$

As before, y_1 and y_2 are the traces of y on elements T_1 and T_2, respectively.

The numerical flux function is already included in the discretization (4.4) on interior faces $F \in \mathbb{F}_h^{int}$ and on boundary faces $F \in \mathbb{F}_h^\Gamma$. On the interior faces, we arrive at the first term by using the symmetry of the $\mathcal{L}^2$ scalar product and the definition of the vector valued jump operator:

$$\sum_{F\in\mathbb{F}_h^{int}} \left(\beta \cdot n\{\!\{y_h\}\!\}, [\![v_h]\!]\right)_F = \sum_{F\in\mathbb{F}_h^{int}} \left(\{\!\{y_h\}\!\}, 2\{\!\{v_h\beta \cdot n\}\!\}\right)_F .$$

4.2. Diffusion advection reaction discrete problem

In this section we consider the diffusion advection reaction model problem Example 2.1.5. It reads

4.2.1 Example. *Diffusion advection reaction model problem revisited.*

$$\begin{aligned} -\nabla \cdot (\alpha \nabla y) + \beta(p) \cdot \nabla y + \rho(p) y = f(p) &\quad \text{on } \Omega \\ y = y_D(p) &\quad \text{on } \mathit{\Gamma} = \partial\Omega. \end{aligned}$$

for diffusion coefficient $\alpha > 0$ and reaction coefficient $\rho \geq 0$. △

The discretization of this model problem is obtained by combining the individual discretizations for diffusion, advection and reaction. We discretize the diffusion part with the interior penalty method. The advection part is discretized by the upwind method from the last Section 4.1. For the reaction part we utilize an additional mass matrix. The right hand side of the discretization consists of the term for the right hand side function f and the face terms resulting from the interior penalty and the upwind discretization, which contain the Dirichlet boundary function y_D.

For the discrete test space we choose the finite element space V_h. For the interior penalty method we define a penalty factor γ. It is dependent on the polynomial degree k_i and the mesh size function $h_i(x) = \operatorname{diam}(T), x \in T$, of both adjacent cells $i = 1, 2$,

$$\gamma := \frac{1}{2}\left(\gamma_1 + \gamma_2\right), \quad \gamma_i := \frac{k_i(k_i+1)}{h_i}, \quad i = 1, 2.$$

The discrete problem for the diffusion advection reaction model problem thus reads: Find $y_h \in V_h$ such that

$$\alpha a_h(y_h, v_h) + b_h(p; y_h, v_h) + \rho(p) c_h(p; y_h, v_h) = f_h(p; v_h), \quad \forall v_h \in V_h, \tag{4.7a}$$

where the diffusion part reads

$$
\begin{aligned}
a_h(y_h, v_h) &:= \sum_{T\in\mathbb{T}_h} (\nabla y_h, \nabla v_h)_T \\
&+ \sum_{F\in\mathbb{F}_h^{int}} \Big[-2\left(\{\!\{\nabla y_h\}\!\}, \{\!\{v_h n\}\!\}\right)_F - 2\left(\{\!\{y_h n\}\!\}, \{\!\{\nabla v_h\}\!\}\right)_F + \gamma\left([\![y_h]\!], [\![v_h]\!]\right)_F \Big] \\
&\qquad + \sum_{F\in\mathbb{F}_h^{\Gamma}} \Big[-(\partial_n y_h, v_h)_F - (y_h, \partial_n v_h)_F + 2\gamma\,(y_h, v_h)_F \Big],
\end{aligned}
\tag{4.7b}
$$

the advection part reads

$$
\begin{aligned}
b_h(p; y_h, v_h) = &- \sum_{T\in\mathbb{T}_h} (y_h, \beta(p)\cdot\nabla v_h)_T + \sum_{F\in\mathbb{F}_h^{int}} \left(\{\!\{y_h\}\!\}, 2\{\!\{v_h\,\beta(p)\cdot n\}\!\}\right)_F + \\
\sum_{F\in\mathbb{F}_h^{int}} &\left(\frac{1}{2}\sigma_{upw}(\beta(p), n)\,[\![y_h]\!], [\![v_h]\!]\right)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(\frac{1}{2}\beta(p)\cdot n y_h + \frac{1}{2}\sigma_{upw}(\beta(p), n) y_h, v_h\right)_F,
\end{aligned}
\tag{4.7c}
$$

the reaction part reads

$$
c_h(p; y_h, v_h) := \sum_{T\in\mathbb{T}_h} (y_h, v_h)_T\,, \tag{4.7d}
$$

and the right hand side is

$$
\begin{aligned}
f_h(p; v_h) := \sum_{T\in\mathbb{T}_h} (f(p), v_h)_T &+ \sum_{F\in\mathbb{F}_h^{\Gamma}} \Big[2\gamma\alpha\,(y_D(p), v_h)_F - (\alpha y_D(p), \partial_n v_h)_F \Big] \\
&- \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(\frac{1}{2}\beta(p)\cdot n y_D(p) - \frac{1}{2}\sigma_{upw}(\beta(p), n) y_D(p), v_h\right)_F.
\end{aligned}
\tag{4.7e}
$$

For shorter notation we define

$$
F_h(p; y_h, v_h) := \alpha a_h(y_h, v_h) + b_h(p; y_h, v_h) + \rho(p) c_h(p; y_h, v_h) - f_h(p; v_h). \tag{4.8}
$$

The discrete problem (4.7) in this short notation reads: Find $y_h \in V_h$ such that

$$
F_h(p; y_h, v_h) = 0, \quad \forall v_h \in V_h. \tag{4.9}
$$

4.3. Discrete optimization problems

With this discrete PDE problem the optimization problems change. Similar to the continuous case in Problem 2.2.1 the discrete constrained PE problem reads

4.3.1 Problem. ***(Discrete constrained parameter estimation problem)*** *Minimize the least squares functional $\bar{G}$*

$$\min_{p \in P, y_h \in V_h} \bar{G}\left(p; y_h\right)$$

subject to

$$F_h(p; y_h, v_h) = 0, \quad \forall v_h \in V_h,$$

which is given as the discretized partial differential equation problem defined in equations (4.7), (4.8) and (4.9).

With the same assumptions as in Section 2.2 we define a discrete solution operator $S_h(p)$. The two assumptions for the continuous bilinear form $F(p; y, v)$, should now hold for the discrete bilinear form $F_h(p; y_h, v_h)$. Then the discrete solution operator $S_h : P \to V_h$ exists and satisfies the discrete state equation

$$F_h(p; S_h(p), v_h) = 0, \quad \forall v_h \in V_h.$$

By inserting this operator $S_h(p)$ into Problem 4.3.1 we obtain the corresponding discrete unconstrained or discrete reduced problem.

4.3.2 Problem. ***(Discrete unconstrained parameter estimation problem)*** *Minimize the discrete reduced least squares functional $G_h(p) := \bar{G}\left(p; S_h(p)\right)$*

$$\min_{p \in P} G_h(p).$$

Similar to the continuous problem, the solution of the discretized PDE problem enters the discrete unconstrained parameter estimation Problem 4.3.2 via the discrete solution operator $S_h(p)$.

In the same manner we define the discrete reduced measurement functions $h_{i,h} : P \to \mathbb{R}$, $h_{i,h}(p) := \bar{h}_i(p; S_h(p))$, and the discrete reduced residuals $r_{i,h} : P \to \mathbb{R}$, $r_{i,h}(p) := \bar{r}_i(p; S_h(p))$. With that, the entries of the dicrete Jacobian J_h of the discrete unconstrained PE problem, Problem 4.3.2, are

$$J_{i,j,h}(p) = -\frac{\sqrt{w_i}}{\sigma_i}\frac{dh_{i,h}(p)}{dp_j}.$$

The discrete Jacobian enters the discrete covariance matrix $C_h = (J_h^T J_h)^{-1}$. Thus we arrive at the discrete OED problem including the discrete OED criterion $\Phi_h(C_h(w,p))$:
4.3.3 Problem. ***(Discrete optimum experimental design problem)***

$$\min_w \Phi_h(C_h(w,p))$$

subject to

$$\begin{aligned} 0 &\leq c_m(w), \\ w_i &\in \{0,1\}, \qquad i = 1, ..., n_g. \end{aligned}$$

As before the OED problem includes sampling decisions $w_i, 1, ..., n_g$ and linear constraint function $c_m(w)$.

As mentioned before in Chapters 2 and 3, existence and uniqueness of solutions are highly problem dependent. We assume, that solutions to the discrete parameter estimation problems exist. Detailed investigations of existence and uniqueness of a solution can be found in [3], [54], [89], [93], [95]. For the discrete OED problem we assume, that a solution exists.

4.4. Finite element algorithm

In this section, we explain the generation of a finite dimensional problem and the process of assembling stiffness matrix and load vector. We follow [47], [85].

Generating the finite dimensional problem To solve the discrete PDE problem (4.7) or (4.9) computationally, we reformulate it into a system of equations. We introduce a basis $\{\varphi_i\}_{i=1}^{n_b}$, with n_b number of basis functions. The discrete solution y_h has the representation

$$y_h := \sum_{i=1}^{n_b} \varphi_i \tilde{y}_i.$$

We use this basis representation to reformulate the discrete problem, with $g_h(p; y_h, v_h) :=$ $\alpha a_h(y_h, v_h) + b_h(p; y_h, v_h) + \rho(p) c_h(p; y_h, v_h)$ including all terms on the left hand side of the equation (4.7),

$$\begin{aligned} & F_h(p; y_h, v_h) = 0, \\ \Leftrightarrow \quad & g_h(p; y_h, v_h) = f_h(p; v_h), \\ \Leftrightarrow \quad & \sum_{i=1}^{n_b} g_h(p; \varphi_i, \varphi_j) \tilde{y}_i = f_h(p; \varphi_j), \quad \forall j = 1, ..., n_b. \end{aligned} \tag{4.11}$$

We rewrite the last line (4.11) as a system of equations

$$A_h \tilde{y} = \tilde{f}_h, \tag{4.12}$$

with entries

$$\begin{aligned} a_{j,i} &:= g_h(p; \varphi_i, \varphi_j), \quad j, i = 1, .., n_b, \\ \tilde{f}_{h,j} &:= f_h(p; \varphi_j), \quad j = 1, ..., n_b. \end{aligned}$$

The matrix A_h is called *stiffness matrix* , the vector $\tilde{f}_h$ is called *load vector.* The included integrals are approximated by numerical quadrature. Furthermore, we compute the integrals on a reference element and map it afterwards to the real elements. This way we efficiently compute the integrals.

Assembling of stiffness matrix and load vector Before we solve this system of equations (4.12) with an iterative solver, we compute the entries of the stiffness matrix A_h and the load vector $\tilde{f}_h$. Therefore, we first compute the single contributions of the cells, interior faces and boundary faces. Then we sum all single contributions to arrive at the entries of the stiffness matrix and of the load vector. This process of summation is called *assembling* of A_h and $\tilde{f}_h$.

In detail, the assembling algorithm proceeds as follows: We first compute values of the basis functions φ_i and φ_j at the quadrature points. Then we compute the full discrete terms for the single elements, that means for cells, interior faces and boundary faces. We name a local matrix for cells $A_T \in \mathbb{R}^{n_{dc} x n_{dc}}$, it comprises the contribution of one cell T, with n_{dc} number of degrees of freedom (DoFs) per cell. Similar, matrices $A_{F^{int}} \in \mathbb{R}^{n_{dc} x n_{dc}}$ and $A_{F^{\Gamma}} \in \mathbb{R}^{n_{dc} x n_{dc}}$ comprise the contributions of an interior face or a boundary face, respectively. The last step is the summation of single summands. We sum up the single contributions (local matrices) to arrive at entries of A_h and $\tilde{f}_h$

(global matrices). An entry $a_{T[i,j]}$ of a local matrix corresponds to the global matrix entry $a_{j,i}$. Similar an entry of a local interior face matrix $a_{F^{int}[i,j]}$ or a local boundary face matrix $a_{F^{\Gamma}[i,j]}$ corresponds to a global matrix entry $a_{j,i}$. Thus by summation of the local entries we arrive at a global entry

$$a_{j,i} = \sum_{T \in \mathbb{T}_h} a_{T[i,j]} + \sum_{F \in \mathbb{F}_h^{int}} a_{F^{int}[i,j]} + \sum_{F \in \mathbb{F}_h^{\Gamma}} a_{F^{\Gamma}[i,j]}.$$

5. Sensitivity evaluation

In this chapter we consider sensitivity evaluation techniques. We begin with presenting the sensitivities for the two optimization problems, the PE problem and the OED problem. After that, we introduce the principle of internal numerical differentiation (IND), which we will further develop in this thesis. We explain the approach of analytical sensitivity evaluation, especially the sensitivity approach. Finally, we treat automatic differentiation (AD).

5.1. Sensitivities for parameter estimation and optimum experimental design

We are interested in the sensitivities for PE and OED, which involve the PDE model problem. The remaining sensitivities, which are independent of the PDE model problem, are computed in the same way as in the ODE or DAE setting. That is, the derivative of the model response with respect to the parameters $\frac{\partial \bar{h}_i(p;S(p))}{\partial p_j}$ and the gradient of the constraints $\nabla c_j, j \in \mathcal{E} \cup \mathcal{I}$, of the relaxed OED Problem 3.5.1. For details on these derivatives see [66], [97], [56].

Sensitivities for parameter estimation As depicted in the problem setting Chapter 2, Section 2.3, to solve the optimization parameter estimation Problem 2.2.2 with the Gauss-Newton method, we need to calculate the entries of the Jacobian $J(p)$ of the residuals $r(p)$. The Jacobian consists of the directional derivatives of the model responses $h_i(p)$ with respect to the parameters p, multiplied by the weighting factor $\frac{1}{\sigma_i}$. Thus the entries of the Jacobian are

$$J_{i,j}(p) = \frac{d}{dp_j} r_i(p) \delta p_j = \frac{d}{dp_j} \left(\frac{\eta_i - h_i(p)}{\sigma_i} \right) \delta p_j = -\frac{1}{\sigma_i} \frac{dh_i(p)}{dp_j} \delta p_j. \tag{5.1}$$

The solution of the PDE model problem $S(p)$ enters the sensitivity (5.1) via the model response function $h_i(p) = \bar{h}_i(p; S(p))$.

Sensitivities for optimum experimental design In the OED setting, the evaluation of the derivative of the objective function $\Phi(C(w,p))$ with respect to the sampling decisions w is of interest. Again we compute a directional derivative. Here we use the definition of directional derivatives for matrices [66]. We utilize the chain rule for matrices

$$\frac{d\Phi}{dw}\delta w = \frac{d\Phi}{dC}\frac{dC}{dJ}\frac{dJ}{dw}\delta w.$$

We are mainly interested in the last term $\frac{dJ}{dw}\delta w$. The entries of the Jacobian of the PE problem for OED, Problem 3.1.1, are

$$J_{i,j}(p) = \frac{d}{dp_j}\left(\sqrt{w_i}\frac{\eta_i - h_i(p)}{\sigma_i}\right) = -\frac{\sqrt{w_i}}{\sigma_i}\frac{dh_i(p)}{dp_j}.$$

Thus the directional derivative of the Jacobian with respect to sampling decisions w reads

$$\frac{dJ}{dw}\delta w = -\operatorname{diag}\left(\frac{d}{dw_i}\frac{\sqrt{w_i}}{\sigma_i}\delta w_i\right)\frac{dh(p)}{dp} = -\frac{1}{2}\operatorname{diag}\left(\frac{\delta w_i}{\sqrt{w_i}\sigma_i}\right)\frac{dh(p)}{dp}.$$

We observe that the PDE model problem only enters the derivative via the last term $\frac{dh(p)}{dp}$. Moreover the sensitivity of the PE problem (5.1) already contains this term. Thus we need to compute the same derivative involving the PDE model problem for PE and OED.

Notice furthermore that the sensitivity $\frac{dh(p)}{dp}$ only changes with different parameters, not with different sampling decisions. Thus for OED we only need to compute that part of the sensitivity once. In OED the parameters stay fixed, we perform the optimization with the sampling decisions.

5.2. Principle of internal numerical differentiation (IND)

Let us review methods to calculate the sensitivities for PE and OED from Section 5.1. We could treat the solver as a black box and use finite differences to calculate the sensitivities. That approach is called *external numerical differentiation (END)*. It is the easiest way, but it has some disadvantages: the calculated sensitivities introduce an error, they are not accurate. On top, the adaptive components of the solver are not considered. The adaptive components change depending on the input and they are not differentiable. This introduces an additional error, which can become arbitrary large.

A second way to calculate the sensitivities is to analytically differentiate the primal PDE problem by for example the sensitivity approach. There, we apply the implicit function theorem and arrive at tangential equations. These tangential PDE problems can be derived analytically and after that discretized and solved. The analytically differentiate approach is extensively used, see for example [17], [18], [22], [23], [34], [83]. A drawback of this approach is, first, that we have to derive the equations analytically by hand, and, second, that again the adaptive components could be differently chosen for primal and tangential equations. That way it is not clear if the computed sensitivities are consistent. The error can become arbitrary large.

A third way to calculate sensitivities is automatic differentiation (AD). Here, the computational code is processed by an AD tool, which generates new code to calculate the sensitivities [46]. In black box AD, the complete code is processed. In the context of PDE-constrained optimization, this easily leads to storage space problems [91]. Despite of this black box approach, there exist few publications where the structure of the discretization is considered [36], [48], [55], [91]. Again a problem with this approach are the adaptive components. Either there are no adaptive algorithms used, which leads to more computing costs than necessary, or the adaptive components are not considered. As explained before, this means that it is not clear if consistent sensitivities are computed.

Because of these problems, we utilize a different way: the principle of *internal numerical differentiation (IND)*, first introduced by Bock for ODEs [25], [26], [27]. It was extended amongst others by [4], [5], [15], see [4] and [89] for an extensive overview. We follow the principle of IND and therefore include the sensitivity evaluation into the numerical scheme. We transfer the principle of IND to PDEs.

We characterize the principle of IND by two aspects

1) We freeze all adaptive components and differentiate the discretized PDE problem with fixed spatial grid, order and step size.
2) We choose the adaptive components such that all PDE problems, the primal problem and the tangential problems, are solved well with similar accuracy.

This definition is suitable with the characterizations in [80], [89]. With the principle of IND we obtain the exact derivative of the solution of the discretized PDE. This is referred to as the "analytical limit of IND" [26].

We have two possibilities to differentiate the discretized PDE: finite differences or automatic differentiation (AD). Again finite differences introduce an error, while AD does not introduce truncation errors. Therefore we will use AD.

In the following two sections we depict analytical sensitivity evaluation and automatic differentiation, because we will make use of both for realizing the transfer of the principle of IND to PDEs. In the following Chapters 6, 7 and 8 we will investigate the transfer in detail and develop methods to apply the principle of IND to PDEs and especially the discontinuous Galerkin method.

5.3. Analytical sensitivity evaluation: sensitivity approach

To evaluate the sensitivities analytically we have two options: the *sensitivity approach*, which is also called the *forward mode*, and the *adjoint approach*, also called the *reverse mode* [54], [17]. Because of the small number of parameters compared to the number of states the sensitivity approach is an efficient way to compute the derivatives. Thus to compute the directional derivatives of the model responses we utilize the sensitivity approach. Let us recapitulate it shortly, we follow [54].

We utilize the chain rule for the directional derivative of the measurement function $h_i(p) = \bar{h}_i(p; S(p))$ with respect to the parameter p_j. For clarity we omit the direction δp_j in the following. We need the partial derivatives with respect to the second argument $\frac{\partial \bar{h}_i(p;S(p))}{\partial S} : Y \to \mathbb{R}$ and with respect to the first argument $\frac{\partial \bar{h}_i(p;S(p))}{\partial p_j} : P \to \mathbb{R}$. We define the partial derivative of the solution operator with respect to a parameter p_j by $S_{p_j} := \frac{\partial S(p)}{\partial p_j}$. We obtain

$$\frac{d\bar{h}_i(p; S(p))}{dp_j} = \frac{\partial \bar{h}_i(p; S(p))}{\partial S} S_{p_j} + \frac{\partial \bar{h}_i(p; S(p))}{\partial p_j}. \tag{5.2}$$

In equation (5.2) the only derivative we are not able to calculate directly is S_{p_j}.

In Section 2.2 we assumed that the bilinear form $F(p; S(p), v)$ is continuously Fréchet-differentiable. Let us make two additional assumptions.

5.3.1 Assumption. The partial derivative of F with respect to S

$$\frac{\partial F(p; S(p), v)}{\partial S} \tag{5.3}$$

is in the normed space of all linear and continuous mappings from Y to $\mathbb{R}$, that means $\frac{\partial F(p;S(p),v)}{\partial S} \in \mathcal{L}(Y, \mathbb{R}) = Y^*$. △

5.3.2 Assumption. The derivative (5.3) is continuously invertible. △

With these assumptions, we can apply the implicit function theorem. It ensures that the solution operator $S(p)$ is continuously differentiable with respect to p. The partial derivative of the solution operator S with respect to the j-th parameter S_{p_j} solves the *tangential equation*, also called *sensitivity equation*, which is a direct result of the implicit function theorem:

$$\frac{dF(p;S(p),v)}{dp_j} = \frac{\partial F(p;S(p),v)}{\partial S} S_{p_j} + \frac{\partial F(p;S(p),v)}{\partial p_j} = 0, \quad \forall v \in V(\Omega). \tag{5.4}$$

To solve the tangential equation (5.4) we need to calculate two derivatives, $\frac{\partial F(p;S(p))}{\partial S}$ and $\frac{\partial F(p;S(p))}{\partial p_j}$. After calculating these two derivatives, we obtain the requested one $S_{p_j} = \frac{\partial S(p)}{\partial p_j}$ by solving the tangential equation (5.4). Note that the tangential equation (5.4) is a weak form of a PDE problem.

For every parameter p_j one tangential equation has to be prepared and solved. The calculation of the Jacobian consists of two steps:

1) solve tangential equations (5.4) for $p_j, j = 1, .., n_p$,

2) insert solutions S_{p_j} in equation (5.2), respectively, and compute the entries of the Jacobian (5.1).

Discrete setting We derive the tangential equation in a similar way for the discrete setting, which we introduced in Section 4.3.

The discrete directional derivative of the measurement function $h_{i,h}(p) = \bar{h}_i(p; S_h(p))$ with respect to the parameter p_j reads

$$\frac{d\bar{h}_i(p;S_h(p))}{dp_j} = \frac{\partial \bar{h}_i(p;S_h(p))}{\partial S_h} S_{h,p_j} + \frac{\partial \bar{h}_i(p;S_h(p))}{\partial p_j}. \tag{5.5}$$

We make similar Assumptions 5.3.1 and 5.3.2 for the discrete derivative

$$\frac{\partial F_h(p;S_h(p),v_h)}{\partial S_h}.$$

This ensures that the discrete solution operator $S_h(p)$ is continuously differentiable with respect to p. With the discrete analogue $S_{h,p_j} := \frac{\partial S_h(p)}{\partial p_j}$, we obtain the *discrete* *tangential equation*

$$\frac{dF_h(p;S_h(p),v_h)}{dp_j} = \frac{\partial F_h(p;S_h(p),v_h)}{\partial S_h} S_{h,p_j} + \frac{\partial F_h(p;S_h(p),v_h)}{\partial p_j} = 0, \quad \forall v_h \in V_h. \tag{5.6}$$

The partial derivative of the discrete solution operator S_h with respect to p_j solves the tangential equation: S_{h,p_j}.

5.4. Automatic differentiation

We now present the approach of automatic differentiation (AD). We follow [77] in this section. For an extensive treatment of the topic of AD see the textbook of Griewank and Walther [46].

Automatic differentiation (AD) calculates derivatives by directly differentiating the programming code, which evaluates the function value. An AD tool processes the programming code. Thus the derivatives are calculated automatically, not by hand as in the analytical setting depicted in the preceding Section 5.3. A big advantage of AD is that compared to finite differences no truncation errors arise [46].

Any function is evaluated by performing a sequence of simple elementary operations, which contain one or two arguments. Two argument operations are for example addition, multiplication and division. Single argument operations are trigonometric, exponential and logarithmic functions. We segment the function into these elementary operations. After that, the chain rule is used to arrive at the derivative.

We distinguish two basic modes of AD: the forward mode and the reverse mode. In the *forward mode*, we split the function evaluation in simple elementary operations and compute one after another. The results of intermediate computations are called intermediate variables. The input variables are called independent variables. We evaluate function values at the independent and intermediate variables and compute with the help of the chain rule the derivatives of the intermediate variables. Finally, we arrive at the function and gradient values of the overall function.

Contrary, in the *reverse mode*, we do not evaluate function and gradient values concurrently. Instead, after the evaluation of the function value, the partial derivatives with respect to intermediate and independent variables are computed by a reverse sweep and the chain rule is applied backwards. Here we have to save the function values, that means we need more memory than in the forward mode.

Computational costs The costs of the forward mode grow linearly with the number of directions and thus independent variables. In the reverse mode the cost grow linearly with the number of backward directions and thus with the number of dependent variables [46]. In the parameter estimation setting, we have a low number of parameters, that means, independent variables, compared to the number of dependent variables. Thus the forward mode is more efficient and we choose it in the following.

Software Regarding AD software, we distinguish two approaches: source code transformation and operator overloading. In source code transformation, the AD tool processes the code before compile time and creates new code for the derivative evaluation. In operator overloading, for each elementary operation the meaning of the corresponding operator is redefined. The operator does not only evaluate the elementary operation, but also the associated gradient object [24].

We shortly recap advantages and disadvantages of these two approaches from [24]. An advantage of operator overloading is that only one additional class is needed and changes in the differentiation procedure only need to be done in this class, the source code remains unaffected. Compared to that, in source code transformation the implementation is complex. If the given source code exceeds a certain level of complexity, a black-box source code transformation is infeasible. Disadvantages of operator overloading are the lack of transparency and dependent on the compiler the runtime overhead can be substantial. In contrast, the code generated by source code transformation is simple, which also facilitates compiler optimizations. Furthermore, in source code transformation, there is more flexibility in applying derivative rules. The context of the specific computation is available, not only one elementary operation [24].

By transferring the principle of IND and therefore exploiting the structure and only differentiating parts of the code, we circumvent the problem of high complexity in the implementation of source code transformation. With that, the advantages of source code transformation exceed the disadvantages. Thus, we choose code transformation.

Part III.

Discretization: New differentiable discontinuous Galerkin finite element method

6. Analysis and numerical results of differentiable upwind method

In this chapter we propose a differentiable discretization for the advection part of the PDE problem. The standard upwind discontinuous Galerkin method is not continuously differentiable. Therefore, we first propose a differentiable upwind discontinuous Galerkin method. Thereafter, the analysis of the new method follows, including consistency and coercivity of the discretization, stability estimate, error estimate for the $\mathcal{L}^2$-projection, error estimate in the energy norm and superconvergence result. After that, we consider the diffusion advection reaction PDE model and show that the error estimates also hold in this setting. Thereafter we investigate the case of a non normalized advection coefficient, we show that the convergence analysis changes, the constant is now dependent on the advection coefficient. We show, that all other properties of the error estimates remain unchanged. Finally, numerical results for the developed differentiable discretization are presented.

6.1. Upwind method with differentiable stabilization

In this section we are concerned with the upwind discretization of the advection part. To simplify the notation we omit the dependency on the parameters p in this chapter. We are analyzing in the next subsections the pure advection problem Example 2.1.7:

$$\beta \cdot \nabla y = f \qquad \text{on } \Omega, \tag{6.1a}$$

$$y = y_D \quad \text{on } \mathit{\Gamma}_-, \tag{6.1b}$$

for $\|\beta\| \neq 0$.

The corresponding discrete problem derived in Section 4.1 is: Find $y_h \in V_h$ such that

$$b_h(y_h, v_h) = f_h^a(v_h), \quad \forall v_h \in V_h, \tag{6.2a}$$

where

$$b_h(y_h,v_h) = -\sum_{T\in\mathbb{T}_h} (y_h, \beta \cdot \nabla v_h)_T + \sum_{F\in\mathbb{F}_h^{\mathit{\Gamma}}} \left(\frac{1}{2}\beta \cdot n y_h + \frac{1}{2}\sigma_{upw}(\beta, n) y_h, v_h\right)_F$$

$$+\sum_{F\in\mathbb{F}_h^{int}} (\{\!\{y_h\}\!\}, 2\{\!\{v_h\beta\cdot n\}\!\})_F + \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_{upw}(\beta,n)\,[\![y_h]\!]\,,[\![v_h]\!]\right)_F, \tag{6.2b}$$

and right hand side

$$f_h^a(v_h) = \sum_{T\in\mathbb{T}_h} (f, v_h)_T - \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(\frac{1}{2}\beta\cdot n y_D - \frac{1}{2}\sigma_{upw}(\beta,n)y_D, v_h\right)_F. \tag{6.2c}$$

In Section 6.3, we come back to the whole diffusion advection reaction problem and show that our analysis can be generalized.

We replace the absolute value function in the stabilization term of the standard upwind discretization (4.5)

$$\sigma_{upw}(\beta,n) = |\beta\cdot n|\,,$$

by a continuously differentiable approximation $\sigma_\mu(\beta,n)$ with variable $\mu\in\mathbb{R}$,

$$\sigma_\mu(\beta,n) := \frac{\sqrt{(\beta\cdot n)^2 + \|\beta\|^2\mu^2}}{\sqrt{1+\mu^2}} \tag{6.3}$$

Let us make one assumption and let us prove two properties of the proposed stabilization function $\sigma_\mu(\beta,n)$. The following assumption simplifies the analysis.

6.1.1 Lemma. *We assume that β is normalized, i.e.*

$$\|\beta\| = 1.$$

The differentiable approximation $\sigma_\mu(\beta,n)$ reduces to

$$\sigma_\mu(\beta,n) = \frac{\sqrt{(\beta\cdot n)^2 + \mu^2}}{\sqrt{1+\mu^2}}. \tag{6.4}$$

At the end of this chapter, in Section 6.4, the case $\|\beta\|\neq 1$ will be addressed.

6.1.2 Lemma. *The following inequality holds*

$$|\beta\cdot n| \le \sigma_\mu(\beta,n), \quad \textit{for} \quad \beta\cdot n\in[-1,1].$$

Proof. Because it holds $(\beta \cdot n)^2 \leq 1$ for $\beta \cdot n \in [-1, 1]$, the inequality holds:

$$\begin{aligned}\sigma_\mu(\beta, n) &= \frac{\sqrt{(\beta \cdot n)^2 + \mu^2}}{\sqrt{1+\mu^2}} \\ &= \sqrt{(\beta \cdot n)^2} \left(\sqrt{\frac{1 + \left(\frac{\mu}{\beta \cdot n}\right)^2}{1+\mu^2}} \right) \geq |\beta \cdot n| \,.\end{aligned}$$

□

This will be important in the following proofs of error estimates.

6.1.3 Lemma. *Furthermore, $\sigma_\mu(\beta, n)$ is bounded from above*

$$\sigma_\mu(\beta, n) \leq 1.$$

Proof. This holds true, because

$$\begin{aligned} & |\beta \cdot n| && \leq \|\beta\| \, \|n\| = 1 \\ \Rightarrow \quad & (\beta \cdot n)^2 && \leq 1 \\ \Rightarrow \quad & (\beta \cdot n)^2 + \mu^2 && \leq 1 + \mu^2 \\ \Rightarrow \quad & \sqrt{(\beta \cdot n)^2 + \mu^2} && \leq \sqrt{1+\mu^2} \\ \Rightarrow \quad & \frac{\sqrt{(\beta \cdot n)^2 + \mu^2}}{\sqrt{1+\mu^2}} && \leq 1. \end{aligned}$$

□

6.1.4 Remark. *Numerical flux function for selection of inflow and outflow boundaries.* To ensure that equation (6.2) holds for all faces of the boundary $\mathbb{F}_h^{\varGamma}, \varGamma = \varGamma_- \cup \varGamma_+$, the inflow boundary function y_D is continued on the outflow boundary $\varGamma_+$. For the upwind stabilization $\sigma_{upw}(\beta, n)$ equation (4.5) this continuation will never actually be evaluated. We choose the inflow and outflow faces via the numerical flux function, see Section 4.1. In the case of an outflow face $F \in \varGamma_+$, the terms including y_D cancel out and contribute a zero. For the proposed differentiable stabilization $\sigma_\mu(\beta, n)$ equation (6.3), on the outflow faces the terms including y_D do not cancel out completely, depending on the size of the variable μ. This results in an artifact due to this continuation.

We use a continuation instead of defining separate terms for inflow and outflow boundary, because of the optimization problem. If we estimate the components of the advection direction, the direction changes in the course of the optimization. The inflow boundary is dependent on this direction and thus changes, too. With a continuation it is not necessary to assign the boundary faces to inflow or outflow boundary. Thus if the inflow and outflow boundary change, we do not have to change this assignment. △

6.1.1. Basics

We recapitulate some basic approximation formulas and the reformulation of the advective bilinear form. Both ingredients are used at several places in the proofs in this chapter.

Approximation The $\mathcal{L}^2$-projection $\mathit{\Pi}_h : \mathcal{L}^2(\Omega) \to V_h$ is defined by the orthogonality condition [58]

$$(y - \mathit{\Pi}_h y, v_h)_\Omega = 0, \quad \forall v_h \in V_h. \tag{6.5}$$

The following inequalities are used at several places of the proofs.

Cauchy-Schwarz inequality: $y, v \in \mathcal{L}^2(\Omega)$,

$$\left|(y,v)\right| \leq \|y\| \, \|v\| \quad \Leftrightarrow \quad -(y,v) \geq -\|y\| \, \|v\| \quad \wedge \quad (y,v) \geq -\|y\| \, \|v\| \, . \tag{6.6}$$

Inequality of Schwarz: $y_1, y_2, v_1, v_2 \in \mathbb{R}$,

$$y_1 v_1 + y_2 v_2 \leq (y_1^2 + y_2^2)^{\frac{1}{2}} \cdot (v_1^2 + v_2^2)^{\frac{1}{2}}, \tag{6.7}$$

It is the algebraic form of the Cauchy-Schwarz inequality.

Young's inequality: $\nu > 0$, $a, b \in \mathbb{R}$,

$$a \cdot b \leq \frac{1}{2\nu} a^2 + \frac{\nu}{2} b^2. \tag{6.8}$$

Trace inequality: $y \in \mathcal{H}^k(\mathbb{T}_h)$,

$$\|y\|_{\partial T} \leq C h^{\frac{1}{2}} \|\nabla y\|_T + C h^{-\frac{1}{2}} \|y\|_T \tag{6.9}$$

$$\Leftrightarrow \quad h^{\frac{1}{2}} \|y\|_{\partial T} \leq C h \|\nabla y\|_T + C \|y\|_T \, , \tag{6.10}$$

with constant C independent of the mesh size function h.

Trace inequality for polynomials: $y_h \in \mathcal{P}(T)$,

$$\|\beta \cdot \nabla y_h\|_{\partial T} \leq Ch^{-1/2} \|\beta \cdot \nabla y_h\|_T . \tag{6.11}$$

Inverse estimate: $y_h \in \mathcal{P}(T)$, $k \geq 1$,

$$\|\nabla y_h\|_T \leq C\frac{1}{h} \|y_h\|_T , \tag{6.12}$$

with constant C independent of h.

Green's formula of integration by parts for the advection is

$$(y, \beta \cdot \nabla v)_T = -(\beta \cdot \nabla y, v)_T + (\beta \cdot ny, v)_{\partial T} . \tag{6.13}$$

Reformulation of the advective bilinear form For later use we rewrite $b_h(y_h, v_h)$. Recall $b_h(y_h, v_h)$ from equation (6.2b):

$$\begin{aligned} b_h(y_h,v_h) = &- \sum_{T\in\mathbb{T}_h} (y_h, \beta \cdot \nabla v_h)_T + \sum_{F\in\mathbb{F}_h^{int}} (\{\!\{y_h\}\!\}, 2\{\!\{v_h \beta \cdot n\}\!\})_F \\ &+ \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_\mu(\beta, n) [\![y_h]\!] , [\![v_h]\!]\right)_F , + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(\frac{1}{2}\beta \cdot ny_h + \frac{1}{2}\sigma_\mu(\beta, n)y_h, v_h\right)_F . \end{aligned}$$

6.1.5 Lemma. *For $y_h, v_h \in V_h$ it holds*

$$\begin{aligned} b_h(y_h, v_h) = &\sum_{T\in\mathbb{T}_h} (\beta \cdot \nabla y_h, v_h)_T - \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2} [\![\beta \cdot ny_h]\!] , [\![v_h]\!]\right)_F \\ &+ \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_\mu(\beta, n) [\![y_h]\!] , [\![v_h]\!]\right)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(-\frac{1}{2}\beta \cdot ny_h + \frac{1}{2}\sigma_\mu(\beta, n)y_h, v_h\right)_F . \end{aligned}$$

Proof. First, we use integration by parts (6.13)

$$\begin{aligned} b_h(y_h, v_h) \overset{(6.13)}{=} &\sum_{T\in\mathbb{T}_h} (\beta \cdot \nabla y_h, v_h)_T - \sum_{T\in\mathbb{T}_h} (\beta \cdot ny_h, v_h)_{\partial T} \\ &+ \sum_{F\in\mathbb{F}_h^{int}} (\{\!\{y_h\}\!\}, 2\{\!\{v_h\, \beta \cdot n\}\!\})_F + \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_\mu(\beta, n) [\![y_h]\!] , [\![v_h]\!]\right)_F \end{aligned}$$

$$+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left(\frac{1}{2}\beta\cdot ny_h+\frac{1}{2}\sigma_\mu(\beta,n)y_h,v_h\right)_F.$$

After that we reorder the sum from cell boundaries ∂T to faces F and summing up with the terms on interior and boundary faces:

$$-\sum_{T\in\mathbb{T}_h}(\beta\cdot ny_h,v_h)_{\partial T}+\sum_{F\in\mathbb{F}_h^{int}}(\{\!\{y_h\}\!\},2\{\!\{v_h\beta\cdot n\}\!\})_F+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left(\frac{1}{2}\beta\cdot ny_h,v_h\right)_F. \tag{6.14}$$

On the boundary $\mathbb{F}_h^{\Gamma}$ we get

$$-\sum_{F\in\mathbb{F}_h^{\Gamma}}(\beta\cdot ny_h,v_h)_F+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left(\frac{1}{2}\beta\cdot ny_h,v_h\right)_F=\sum_{F\in\mathbb{F}_h^{\Gamma}}\left(-\frac{1}{2}\beta\cdot ny_h,v_h\right)_F.$$

For the reordering of the first sum in (6.14) with respect to inner faces let us consider an inner face F of two adjacent cells T_1 and T_2 and let n_1 be the outward pointing normal vector of T_1 at F

$$-\Big((\beta\cdot n_1y_1,v_1)_{F(T_1)}+(\beta\cdot n_2y_2,v_2)_{F(T_2)}\Big)=-\int_F(\beta\cdot n_1y_1v_1+\beta\cdot n_2y_2v_2)ds.$$

The summand in the second sum of (6.14) is

$$\begin{aligned}(\{\!\{y_h\}\!\},2\{\!\{v_h\beta\cdot n\}\!\})_F&=\int_F\frac{1}{2}(y_1+y_2)(v_1\beta\cdot n_1+v_2\beta\cdot n_2)ds\\&=\int_F\frac{1}{2}\left(y_1v_1\beta\cdot n_1+y_1v_2\beta\cdot n_2+y_2v_1\beta\cdot n_1+y_2v_2\beta\cdot n_2\right)ds.\end{aligned}$$

Adding these two terms yields

$$\int_F\frac{1}{2}\left(-y_1v_1\beta\cdot n_1+y_1v_2\beta\cdot n_2+y_2v_1\beta\cdot n_1-y_2v_2\beta\cdot n_2\right)ds.$$

On the other hand, it holds

$$-\frac{1}{2}([\![\beta\cdot ny_h]\!],[\![v_h]\!])_F=-\frac{1}{2}\int_F(\beta\cdot n_1y_1-\beta\cdot n_2y_2)(v_1-v_2)ds$$

$$= -\frac{1}{2}\int\limits_F (\beta \cdot n_1 y_1 v_1 - \beta \cdot n_1 y_1 v_2 - \beta \cdot n_2 y_2 v_1 + \beta \cdot n_2 y_2 v_2) ds.$$

Thus, the equality in Lemma 6.1.5 holds true. □

6.2. Analysis of differentiable upwind method

For the analysis of the proposed differentiable upwind method we mainly follow the standard procedure for convergence analysis of finite element discretizations:

6.2.1 Consistency of the discretization

6.2.2 Coercivity of the bilinear form and definition of energy norm

6.2.3 Stability estimate for bilinear form

6.2.4 $\mathcal{L}^2$-projection error estimate

6.2.5 Estimate for bilinear form and error estimate in the energy norm

6.2.6 Superconvergence result

An analysis of the upwind discontinuous Galerkin method for advection problems without differentiable stabilization can be found in [45], [58], [59], [60].

6.2.1. Consistency of the discretization

6.2.1 Definition. ([38]) The discretization is called *consistent*, if $b_h(.,.)$ can be extended from $V_h \times V_h$ to $[V(\Omega) + V_h] \times V_h$ and if the exact weak solution $y \in V(\Omega)$ of problem (6.1) solves the discrete problem (6.2). That means

$$b_h(y, v_h) = f_h(v_h), \quad v_h \in V_h.$$

△

Assume $y \in V(\Omega)$ to be the exact weak solution of (6.1).
6.2.2 Lemma. *The discretization is consistent. That means the exact weak solution y of (6.1) satisfies the discrete problem (6.2) .*

Proof. Using the facts that all jumps vanish $[\![y]\!] = 0$, the averages are $\{\!\{y\}\!\} = y$ and

$y = y_D$ on Γ, the remaining parts of (6.2) are

$$\begin{aligned}
&- \sum_{T\in\mathbb{T}_h} (y, \beta \cdot \nabla v_h)_T + \sum_{F\in\mathbb{F}_h^{int}} (y, 2\{\!\{v_h \beta \cdot n\}\!\})_F \\
&+ \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(\frac{1}{2}\beta \cdot n y_D + \frac{1}{2}\sigma_\mu(\beta, n) y_D, v_h\right)_F && (6.15a)\\
&\quad = \sum_{T\in\mathbb{T}_h} (f, v_h)_T - \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(\frac{1}{2}\beta \cdot n y_D - \frac{1}{2}\sigma_\mu(\beta, n) y_D, v_h\right)_F \\
\Leftrightarrow \quad &- \sum_{T\in\mathbb{T}_h} (y, \beta \cdot \nabla v_h)_T + \sum_{F\in\mathbb{F}_h^{int}} (y, 2\{\!\{v_h \beta \cdot n\}\!\})_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} (\beta \cdot n y_D, v_h)_F \\
&\quad = \sum_{T\in\mathbb{T}_h} (f, v_h)_T && (6.15b)
\end{aligned}$$

for all $v_h \in V_h$. Using integration by parts (6.13), equation (6.15b) is equivalent to

$$\begin{aligned}
&\sum_{T\in\mathbb{T}_h} (\beta \cdot \nabla y, v_h)_T - \sum_{T\in\mathbb{T}_h} (\beta \cdot n y, v_h)_{\partial T} \\
&\quad + \sum_{F\in\mathbb{F}_h^{int}} (y, 2\{\!\{v_h \beta \cdot n\}\!\})_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} (\beta \cdot n y_D, v_h)_F = \sum_{T\in\mathbb{T}_h} (f, v_h)_T \\
\Leftrightarrow \quad &\sum_{T\in\mathbb{T}_h} (\beta \cdot \nabla y, v_h)_T = \sum_{T\in\mathbb{T}_h} (f, v_h)_T \\
\Leftrightarrow \quad &\sum_{T\in\mathbb{T}_h} (\beta \cdot \nabla y - f, v_h)_T = 0.
\end{aligned}$$

Hence, the exact solution of (6.1) satisfies the discretization (6.2). □

6.2.2. Coercivity

6.2.3 Lemma. *The bilinear form $b_h(.,.)$ of (6.2b) is positive definite.*

Proof. Let $v_h \in V_h$. According to Green's formula for integration by parts (6.13) we have

$$(v_h, \beta \cdot \nabla v_h)_T = -(\beta \cdot \nabla v_h, v_h)_T + (\beta \cdot n v_h, v_h)_{\partial T}.$$

Due to symmetry of the $\mathcal{L}^2$ scalar product this gives

$$2(v_h, \beta \cdot \nabla v_h)_T = (\beta \cdot n v_h, v_h)_{\partial T} \quad \Leftrightarrow \quad (v_h, \beta \cdot \nabla v_h)_T = \frac{1}{2}(\beta \cdot n v_h, v_h)_{\partial T}.$$

Now summing up over all cells yields

$$\sum_{T\in\mathbb{T}_h} (v_h, \beta\cdot\nabla v_h)_T = \frac{1}{2}\sum_{T\in\mathbb{T}_h} (\beta\cdot n v_h, v_h)_{\partial T}\,. \tag{6.16}$$

We now reorder the sum with respect to faces. Let us consider an inner face F of two adjacent cells T_1 and T_2 and let be n_1 the outward pointing normal vector of T_1 at F,

$$\frac{1}{2}(\beta\cdot n_1 v_1, v_1)_{F(T_1)} + \frac{1}{2}(\beta\cdot n_2 v_2, v_2)_{F(T_2)} = \frac{1}{2}\int\limits_F (\beta\cdot n_1 v_1^2 + \beta\cdot n_2 v_2^2)ds.$$

On the other hand, on an inner face F it holds

$$\begin{aligned}
(\{\!\{v_h\}\!\}, 2\{\!\{v_h\beta\cdot n\}\!\})_F &= \int\limits_F \frac{1}{2}(v_1+v_2)2\frac{1}{2}(v_1\beta\cdot n_1 + v_2\beta\cdot n_2)ds\\
&= \int\limits_F \frac{1}{2}(v_1^2\beta\cdot n_1 + v_1v_2\beta\cdot n_2 + v_2v_1\beta\cdot n_1 + v_2^2\beta\cdot n_2)ds\\
&= \int\limits_F \frac{1}{2}(v_1^2\beta\cdot n_1 + v_1v_2(\beta\cdot n_2 + \beta\cdot n_1) + v_2^2\beta\cdot n_2)ds\\
&= \int\limits_F \frac{1}{2}(v_1^2\beta\cdot n_1 + v_1v_2(\beta\cdot n_2 - \beta\cdot n_2) + v_2^2\beta\cdot n_2)ds\\
&= \int\limits_F \frac{1}{2}(v_1^2\beta\cdot n_1 + v_2^2\beta\cdot n_2)ds.
\end{aligned}$$

Hence, the right hand side of (6.16) is

$$\frac{1}{2}\sum_{T\in\mathbb{T}_h} (\beta\cdot n v_h, v_h)_{\partial T} = \sum_{F\in\mathbb{F}_h^{int}} (\{\!\{v_h\}\!\}, 2\{\!\{v_h\beta\cdot n\}\!\})_F + \frac{1}{2}\sum_{F\in\mathbb{F}_h^{\Gamma}} (\beta\cdot n v_h, v_h)_F\,. \tag{6.17}$$

Now we evaluate the bilinear form b_h (6.2b) in (v_h, v_h) and insert (6.16) and (6.17) to obtain

$$\begin{aligned}
&b_h(v_h, v_h) =\\
&\quad -\sum_{T\in\mathbb{T}_h} (v_h, \beta\cdot\nabla v_h)_T\\
&\quad + \sum_{F\in\mathbb{F}_h^{int}} (\{\!\{v_h\}\!\}, 2\{\!\{v_h\,\beta\cdot n\}\!\})_F + \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_\mu(\beta, n)\,[\![v_h]\!]\,, [\![v_h]\!]\right)_F
\end{aligned}$$

$$
\begin{aligned}
&+ \sum_{F\in\mathbb{F}_h^\Gamma} \left(\frac{1}{2}\beta\cdot n v_h + \frac{1}{2}\sigma_\mu(\beta,n)v_h, v_h\right)_F && \text{(6.18a)}\\
&\overset{(6.16),(6.17)}{=} - \sum_{F\in\mathbb{F}_h^{int}} (\{\!\{v_h\}\!\}, 2\{\!\{v_h\,\beta\cdot n\}\!\})_F - \frac{1}{2}\sum_{F\in\mathbb{F}_h^\Gamma}(\beta\cdot n v_h, v_h)_F\\
&+ \sum_{F\in\mathbb{F}_h^{int}} (\{\!\{v_h\}\!\}, 2\{\!\{v_h\,\beta\cdot n\}\!\})_F + \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_\mu(\beta,n)\,[\![v_h]\!]\,, [\![v_h]\!]\right)_F\\
&+ \sum_{F\in\mathbb{F}_h^\Gamma} \left(\frac{1}{2}\beta\cdot n v_h + \frac{1}{2}\sigma_\mu(\beta,n)v_h, v_h\right)_F && \text{(6.18b)}\\
&= \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_\mu(\beta,n)\,[\![v_h]\!]\,, [\![v_h]\!]\right)_F + \sum_{F\in\mathbb{F}_h^\Gamma}\left(\frac{1}{2}\sigma_\mu(\beta,n)v_h, v_h\right)_F && \text{(6.18c)}\\
&> 0, \qquad \forall v_h \in V_h. && \text{(6.18d)}
\end{aligned}
$$

The last inequality holds, if $\sigma_\mu(\beta,n) > 0$. This is the case for $\mu \neq 0$. □

Since the bilinear form is positive definite, the linear system (6.2) has a unique solution.

Energy norm Motivated by the coercivity we define the energy norm

$$
|||y|||^2 := b_h(y,y) + \sum_{T\in\mathbb{T}_h} h\,\|\beta\cdot\nabla y\|_T^2\,. \tag{6.19}
$$

Note that the norm depends on $\sigma_\mu(\beta,n)$ via $b_h(y,y)$.

6.2.3. Stability estimate or inf-sup condition

We have to prove that for all $y_h \in V_h$ there exists a $v_h \in V_h$ such that

$$
b_h(y_h,v_h) \geq C|||y_h|||\,|||v_h||| \tag{6.20}
$$

holds with $C > 0$ independent of h.

6.2.4 Lemma. *For $y_h \in V_h$ arbitrarily we set $v_h := C_2 y_h + h\beta\cdot\nabla y_h$ with $C_2 > 0$. It is*

$$
|||v_h||| = |||C_2 y_h + h\beta\cdot\nabla y_h||| \leq C|||y_h|||
$$

where C is independent of h.

Proof. We verify this by

$$\begin{aligned} |||C_2 y_h + h\beta \cdot \nabla y_h||| &\leq C_2 |||y_h||| + |||h\beta \cdot \nabla y_h||| \\ &\leq C_2 |||y_h||| + C|||y_h||| \\ &= (C_2 + C)|||y_h|||. \end{aligned}$$

Let us validate the second inequality. The energy norm with bilinear form b_h as in (6.18c) reads

$$\begin{aligned} |||h\beta \cdot \nabla y_h|||^2 = &\sum_{F \in \mathbb{F}_h^{int}} \left(\frac{1}{2} \sigma_\mu(\beta, n) \left[\!\left[h\beta \cdot \nabla y_h \right]\!\right], \left[\!\left[h\beta \cdot \nabla y_h \right]\!\right] \right)_F \\ &+ \sum_{F \in \mathbb{F}_h^{\Gamma}} \left(\frac{1}{2} \sigma_\mu(\beta, n) h\beta \cdot \nabla y_h, h\beta \cdot \nabla y_h \right)_F + \sum_{T \in \mathbb{T}_h} h \left\| \beta \cdot \nabla (h\beta \cdot \nabla y_h) \right\|_T^2 \end{aligned} \tag{6.21}$$

We first reformulate the face terms

$$\begin{aligned} &\sum_{F \in \mathbb{F}_h^{int}} \left(\frac{1}{2} \sigma_\mu(\beta, n) \left[\!\left[h\beta \cdot \nabla y_h \right]\!\right], \left[\!\left[h\beta \cdot \nabla y_h \right]\!\right] \right)_F + \sum_{F \in \mathbb{F}_h^{\Gamma}} \left(\frac{1}{2} \sigma_\mu(\beta, n) h\beta \cdot \nabla y_h, h\beta \cdot \nabla y_h \right)_F \\ &= \sum_{F \in \mathbb{F}_h^{int}} \frac{1}{2} \sigma_\mu(\beta, n) \left\| \left[\!\left[h\beta \cdot \nabla y_h \right]\!\right] \right\|_F^2 + \sum_{F \in \mathbb{F}_h^{\Gamma}} \frac{1}{2} \sigma_\mu(\beta, n) \left\| h\beta \cdot \nabla y_h \right\|_F^2 . \end{aligned}$$

We relate the faces to the boundary of one element T. Let F be the inner face between cells T_1 and T_2. By the triangle inequality it holds

$$\left\| \left[\!\left[h\beta \cdot \nabla y_h \right]\!\right] \right\|_F^2 = h^2 \left\| \beta \cdot \nabla y_1 - \beta \cdot \nabla y_2 \right\|_F^2 \leq h^2 (\|\beta \cdot \nabla y_1\|_F^2 + \|\beta \cdot \nabla y_2\|_F^2). \tag{6.22}$$

With Lemma 6.1.3 we have that $\frac{1}{2}\sigma_\mu(\beta, n)$ is bounded from above by some $\kappa > 0$, this yields

$$\sum_{F \in \mathbb{F}_h^{int}} \frac{1}{2} \sigma_\mu(\beta, n) \left\| \left[\!\left[h\beta \cdot \nabla y_h \right]\!\right] \right\|_F^2 + \sum_{F \in \mathbb{F}_h^{\Gamma}} \frac{1}{2} \sigma_\mu(\beta, n) \left\| h\beta \cdot \nabla y_h \right\|_F^2 \leq \kappa h^2 \sum_{T \in \mathbb{T}_h} \left\| \beta \cdot \nabla y_h \right\|_{\partial T}^2 .$$

Now we apply the trace inequality (6.11) to obtain for the face part

$$\kappa h^2 \sum_{T \in \mathbb{T}_h} \left\| \beta \cdot \nabla y_h \right\|_{\partial T}^2 \leq \kappa h C \sum_{T \in \mathbb{T}_h} \left\| \beta \cdot \nabla y_h \right\|_T^2 . \tag{6.23}$$

Second, we transform the cell part of equation (6.21) with the Cauchy-Schwarz inequality (6.6) and with the inverse estimate (6.12)

$$\begin{aligned}\sum_{T\in\mathbb{T}_h} h\left\|\beta\cdot\nabla(h\beta\cdot\nabla y_h)\right\|_T^2 &\overset{(6.6)}{\leq} \sum_{T\in\mathbb{T}_h} h\left\|\beta\right\|_T^2\left\|\nabla(h\beta\cdot\nabla y_h)\right\|_T^2\\ &\overset{(6.12)}{\leq} C\sum_{T\in\mathbb{T}_h} h\left\|\beta\cdot\nabla y_h\right\|_T^2 .\end{aligned}$$

Note that we used the Assumption 6.1.1 $\|\beta\| = 1$. Altogether we have

$$\begin{aligned}&\sum_{F\in\mathbb{F}_h^{int}}\left(\frac{1}{2}\sigma_\mu(\beta,n)\left[\!\left[h\beta\cdot\nabla y_h\right]\!\right],\left[\!\left[h\beta\cdot\nabla y_h\right]\!\right]\right)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}}\left(\frac{1}{2}\sigma_\mu(\beta,n)h\beta\cdot\nabla y_h, h\beta\cdot\nabla y_h\right)_F\\ &+\sum_{T\in\mathbb{T}_h} h\left\|\beta\cdot\nabla(h\beta\cdot\nabla y_h)\right\|_T^2\\ \leq&\kappa hC\sum_{T\in\mathbb{T}_h}\left\|\beta\cdot\nabla y_h\right\|_T^2 + C\sum_{T\in\mathbb{T}_h} h\left\|\beta\cdot\nabla y_h\right\|_T^2 .\end{aligned}$$

We add face terms to arrive at the energy norm. The inequality still holds with these additional face terms

$$\begin{aligned}|||h\beta\cdot\nabla y_h|||^2 \leq&C\Bigg[\sum_{F\in\mathbb{F}_h^{int}}\left(\frac{1}{2}\sigma_\mu(\beta,n)\left[\!\left[y_h\right]\!\right],\left[\!\left[y_h\right]\!\right]\right)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}}\left(\frac{1}{2}\sigma_\mu(\beta,n)y_h, y_h\right)_F\\ &+\sum_{T\in\mathbb{T}_h} h\left\|\beta\cdot\nabla y_h\right\|_T^2\Bigg]\\ =&C\Bigg[b_h(y_h,y_h)+\sum_{T\in\mathbb{T}_h} h\left\|\beta\cdot\nabla y_h\right\|_T^2\Bigg] = C|||y_h|||^2. \qquad (6.24)\end{aligned}$$

Thus it holds

$$|||h\beta\cdot\nabla y_h||| \leq C|||y_h|||.$$

□

Therefore it is enough to prove that

$$b_h(y_h, v_h) \geq C|||y_h|||^2, \quad \text{for } \ v_h := C_2 y_h + h\beta\cdot\nabla y_h,$$

which we do in the next lemma.

6.2.5 Lemma. *There exists a constant $C_2 > 0$ such that for all $y_h \in V_h$,*

$$b_h(y_h, C_2 y_h + h\beta \cdot \nabla y_h) \geq C|||y_h|||^2.$$

Proof. We follow the proof of Lemma A.1 in [45]. Since our discretization of the advection differs from [45] we now consider the advection parts of their proof. Hence we have to show that for any C_2 there exists a constant $c_2' > 0$ such that

$$b_h(y_h, C_2 y_h + h\beta \cdot \nabla y_h) \geq C_2 b_h(y_h, y_h) - c_2' b_h(y_h, y_h) + h \left\| \beta \cdot \nabla y_h \right\|_{\Omega}^2 / 2. \tag{6.25}$$

This is enough to prove, because if we choose $C_2 > c_2'$, the inequality in Lemma 6.2.5 holds. We mainly follow the proofs of [45, Lemma A.1] and [60, Lemma 1.4.11]. However, for completeness we sketch all details. We use the reformulation of the bilinear form b_h as in Lemma 6.1.5. Applying Cauchy-Schwarz inequality (6.6) and afterwards Young's inequality (6.8) to all products yields

$$\begin{aligned}
&b_h(y_h, C_2 y_h + h\beta \cdot \nabla y_h) = C_2 b_h(y_h, y_h) + h \left\| \beta \cdot \nabla y_h \right\|_{\mathbb{T}}^2 \\
&\quad - \sum_{F \in \mathbb{F}_h^{int}} \left(\frac{1}{2} [\![\beta \cdot n y_h]\!], [\![h\beta \cdot \nabla y_h]\!] \right)_F + \sum_{F \in \mathbb{F}_h^{int}} \left(\frac{1}{2} \sigma_\mu(\beta, n) [\![y_h]\!], [\![h\beta \cdot \nabla y_h]\!] \right)_F \\
&\quad - \sum_{F \in \mathbb{F}_h^{\Gamma}} \left(\frac{1}{2} \beta \cdot n y_h, h\beta \cdot \nabla y_h \right)_F + \sum_{F \in \mathbb{F}_h^{\Gamma}} \left(\frac{1}{2} \sigma_\mu(\beta, n) y_h, h\beta \cdot \nabla y_h \right)_F \\
&\quad \overset{(6.6),(6.8)}{\geq} C_2 b_h(y_h, y_h) + h \left\| \beta \cdot \nabla y_h \right\|_{\mathbb{T}}^2 - \frac{1}{2\nu} \sum_{F \in \mathbb{F}_h^{int}} \left\| \frac{1}{2} [\![\beta \cdot n y_h]\!] \right\|_F^2 - \frac{\nu}{2} \sum_{F \in \mathbb{F}_h^{int}} \left\| [\![h\beta \cdot \nabla y_h]\!] \right\|_F^2 \\
&\quad - \frac{1}{2\nu} \sum_{F \in \mathbb{F}_h^{int}} \left\| \frac{1}{2} \sigma_\mu(\beta, n) [\![y_h]\!] \right\|_F^2 - \frac{\nu}{2} \sum_{F \in \mathbb{F}_h^{int}} \left\| [\![h\beta \cdot \nabla y_h]\!] \right\|_F^2 \\
&\quad - \frac{1}{2\nu} \sum_{F \in \mathbb{F}_h^{\Gamma}} \left\| \frac{1}{2} \beta \cdot n y_h \right\|_F^2 - \frac{1}{2\nu} \sum_{F \in \mathbb{F}_h^{\Gamma}} \left\| \frac{1}{2} \sigma_\mu(\beta, n) y_h \right\|_F^2 - 2\frac{\nu}{2} \sum_{F \in \mathbb{F}_h^{\Gamma}} \left\| h\beta \cdot \nabla y_h \right\|_F^2 .
\end{aligned}$$

Let us sort the terms and bound them bit by bit afterwards

$$\begin{aligned}
&b_h(y_h, C_2 y_h + h\beta \cdot \nabla y_h) \\
&\geq C_2 b_h(y_h, y_h) + h \left\| \beta \cdot \nabla y_h \right\|_{\mathbb{T}}^2 && (T_0) \\
&\quad - \frac{1}{2\nu} \left(\sum_{F \in \mathbb{F}_h^{int}} \left\| \frac{1}{2} \sigma_\mu(\beta, n) [\![y_h]\!] \right\|_F^2 + \sum_{F \in \mathbb{F}_h^{\Gamma}} \left\| \frac{1}{2} \sigma_\mu(\beta, n) y_h \right\|_F^2 \right) && (T_1)
\end{aligned}$$

$$- \frac{\nu}{2}\left(2 \sum_{F\in\mathbb{F}_h^{int}} \left\|\llbracket h\beta\cdot\nabla y_h\rrbracket\right\|_F^2 + 2 \sum_{F\in\mathbb{F}_h^{\varGamma}} \left\|h\beta\cdot\nabla y_h\right\|_F^2 \right) \tag{T_2}$$

$$- \frac{1}{2\nu}\frac{1}{4}\left(\sum_{F\in\mathbb{F}_h^{int}} \left\|\llbracket\beta\cdot n y_h\rrbracket\right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\varGamma}} \left\|\beta\cdot n y_h\right\|_F^2 \right) \tag{T_3}$$

Let us begin with T_1. Assume that $\frac{1}{2}\sigma_\mu(\beta,n)$ is bounded from above by κ and use (6.18c) to obtain

$$T_1 = -\frac{1}{2\nu}\left(\sum_{F\in\mathbb{F}_h^{int}} \left\|\frac{1}{2}\sigma_\mu(\beta,n)\llbracket y_h\rrbracket\right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\varGamma}} \left\|\frac{1}{2}\sigma_\mu(\beta,n)y_h\right\|_F^2 \right) \geq -\frac{1}{2\nu}\kappa b_h(y_h,y_h). \tag{6.27}$$

For the second term T_2, we proceed in the same fashion as in the proof of the preceding Lemma 6.2.4. We relate the faces to the boundary of one element T. Let F be the inner face between cells T_1 and T_2. With the triangle inequality it holds

$$\left\|\llbracket h\beta\cdot\nabla y_h\rrbracket\right\|_F^2 = h^2\left\|\beta\cdot\nabla y_1 - \beta\cdot\nabla y_2\right\|_F^2 \leq h^2(\left\|\beta\cdot\nabla y_1\right\|_F^2 + \left\|\beta\cdot\nabla y_2\right\|_F^2). \tag{6.28}$$

This then yields

$$T_2 = -\nu\left(\sum_{F\in\mathbb{F}_h^{int}} \left\|\llbracket h\beta\cdot\nabla y_h\rrbracket\right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\varGamma}} \left\|h\beta\cdot\nabla y_h\right\|_F^2 \right) \geq -h^2\nu \sum_{T\in\mathbb{T}_h} \left\|\beta\cdot\nabla y_h\right\|_{\partial T}^2.$$

Now we apply the trace inequality (6.11) to obtain

$$T_2 \geq -h^2\nu \sum_{T\in\mathbb{T}_h} \left\|\beta\cdot\nabla y_h\right\|_{\partial T}^2 \geq -\nu h C \sum_{T\in\mathbb{T}_h} \left\|\beta\cdot\nabla y_h\right\|_T^2. \tag{6.29}$$

Let us finally bound the term T_3

$$\sum_{F\in\mathbb{F}_h^{int}} \left\|\llbracket\beta\cdot n y_h\rrbracket\right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\varGamma}} \left\|\beta\cdot n y_h\right\|_F^2 \tag{6.30a}$$

$$= \sum_{F\in\mathbb{F}_h^{int}} (\llbracket\beta\cdot n y_h\rrbracket, \llbracket\beta\cdot n y_h\rrbracket)_F + \sum_{F\in\mathbb{F}_h^{\varGamma}} (\beta\cdot n y_h, \beta\cdot n y_h)_F. \tag{6.30b}$$

We begin with the sum over the interior faces. We use the definition of the jumps

(4.2) and reformulate

$$\begin{aligned}(\llbracket \beta\cdot n y_h\rrbracket, \llbracket \beta\cdot n y_h\rrbracket)_F &= 4(\{\!\{\beta\cdot n y_h\}\!\}, \{\!\{\beta\cdot n y_h\}\!\})_F \\ &= \int\limits_F (\beta\cdot n_1 y_1 + \beta\cdot n_2 y_2)(\beta\cdot n_1 y_1 + \beta\cdot n_2 y_2) ds \\ &= (\beta\cdot n_1)^2 \left\|\llbracket y_h\rrbracket\right\|^2 .\end{aligned} \tag{6.31}$$

It holds $-1 \leq \beta\cdot n \leq 1$ since $|\beta\cdot n| \leq \|\beta\| \, \|n\| = 1$ due to Assumption 6.1.1. Thus we estimate

$$(\beta\cdot n_1)^2 \left\|\llbracket y_h\rrbracket\right\|^2 \leq |\beta\cdot n_1| \, (\llbracket y_h\rrbracket, \llbracket y_h\rrbracket)_F.$$

Furthermore, according to Lemma 6.1.2, the differentiable stabilization is greater or equal the absolute value function: $\beta\cdot n \leq |\beta\cdot n| \leq \sigma_\mu(\beta, n)$. That means,

$$|\beta\cdot n_1| \, (\llbracket y_h\rrbracket, \llbracket y_h\rrbracket)_F \leq (\sigma_\mu(\beta, n) \llbracket y_h\rrbracket, \llbracket y_h\rrbracket)_F.$$

In the same way we proceed for the term over the boundary faces

$$\begin{aligned}(\beta\cdot n y_h, \beta\cdot n y_h)_F &= (\beta\cdot n_1)^2 (y_h, y_h)_F \\ &\leq |\beta\cdot n_1| \, (y_h, y_h)_F \\ &\leq (\sigma_\mu(\beta, n) y_h, y_h)_F.\end{aligned}$$

Finally, the interior and boundary faces together with the formulation of $b_h(y_h, y_h)$ in (6.18c) read

$$\begin{aligned}&\sum_{F\in\mathbb{F}_h^{int}} (\llbracket \beta\cdot n y_h\rrbracket, \llbracket \beta\cdot n y_h\rrbracket)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} (\beta\cdot n y_h, \beta\cdot n y_h)_F \\ &\leq \sum_{F\in\mathbb{F}_h^{int}} (\sigma_\mu(\beta, n) \llbracket y_h\rrbracket, \llbracket y_h\rrbracket)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} (\sigma_\mu(\beta, n) y_h, y_h)_F \\ &\overset{(6.18c)}{=} 2 b_h(y_h, y_h).\end{aligned} \tag{6.32}$$

Putting all the pieces together, it holds

$$\begin{aligned}&T_0 + T_1 + T_2 + T_3 \\ &\quad \geq C_2 b_h(y_h, y_h) + h \left\|\beta\cdot\nabla y_h\right\|_{\mathbb{T}}^2 - \frac{1}{2\nu}\kappa b_h(y_h, y_h)\end{aligned}$$

$$
\begin{aligned}
&-\nu hC \sum_{T\in\mathbb{T}_h} \|\beta\cdot\nabla y_h\|_T^2 - \frac{1}{2\nu}\frac{1}{4}2b_h(y_h,y_h)\\
\geq & C_2 b_h(y_h,y_h) + (1-\nu C)h\,\|\beta\cdot\nabla y_h\|_{\mathbb{T}}^2 - \frac{1}{2\nu}(\kappa+2)b_h(y_h,y_h).
\end{aligned}
$$

We choose $\nu := \frac{1}{2C}$ and $C_2 > \frac{1}{2\nu}(\kappa+2) = C(\kappa+2)$ to complete the proof. □

The main assumption of Lemmas 6.2.4 and 6.2.5 is

$$
\frac{1}{2}\sigma_\mu(\beta,n) \leq \kappa \quad \forall n. \tag{6.33}
$$

With Lemma 6.1.3 such a κ is easily found.

6.2.4. Error estimate for the $\mathcal{L}^2$-projection

Now we derive an error estimate for the $\mathcal{L}^2$-projection (6.5). This will be used in the next step to get an estimate for the overall error.

6.2.6 Lemma. *For any function $v \in \mathcal{H}^{k+1}(\mathbb{T}_h)$, the error for the $\mathcal{L}^2$-projection in the energy norm can be bounded*

$$
|||v - \mathit{\Pi}_h v||| \leq Ch^{k+\frac{1}{2}}|v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}.
$$

Proof. Again we follow the beginning of the proofs in [45, Theorem 5.1] and [60, Theorem 1.4.13]. The standard estimates for the $\mathcal{L}^2$-projection are, see [35, Theorem 3.1.5],

$$
\sum_{T\in\mathbb{T}_h} \|v - \mathit{\Pi}_h v\|_T \leq Ch^{k+1}|v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}, \tag{6.34}
$$

and

$$
\sum_{T\in\mathbb{T}_h} \|\nabla(v - \mathit{\Pi}_h v)\|_T \leq Ch^{k}|v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}. \tag{6.35}
$$

Due to the definition of the energy norm, the error of the $\mathcal{L}^2$-projection in the energy norm reads

$$
|||(v - \mathit{\Pi}_h v)||| = \left[b_h(v - \mathit{\Pi}_h v, v - \mathit{\Pi}_h v) + \sum_{T\in\mathbb{T}_h} h\,\|\nabla(v - \mathit{\Pi}_h v)\|_T^2 \right]^{\frac{1}{2}}. \tag{6.36}
$$

We begin by estimating the first term. The bilinear form equation (6.18c) is

$$\begin{aligned} b_h(v - \Pi_h v, v - \Pi_h v) &\overset{(6.18c)}{=} \sum_{F \in \mathbb{F}_h^{int}} \left(\frac{1}{2} \sigma_\mu(\beta, n) \llbracket v - \Pi_h v \rrbracket , \llbracket v - \Pi_h v \rrbracket \right)_F \\ &\quad + \sum_{F \in \mathbb{F}_h^{\Gamma}} \left(\frac{1}{2} \sigma_\mu(\beta, n)(v - \Pi_h v), (v - \Pi_h v) \right)_F . \end{aligned}$$

For the interior faces, we get with Young's inequality (6.8) and with $\frac{1}{2}\sigma_\mu(\beta, n) \leq 1$ due to Lemma 6.1.3

$$\begin{aligned} &\left(\frac{1}{2} \sigma_\mu(\beta, n) \llbracket v - \Pi_h v \rrbracket , \llbracket v - \Pi_h v \rrbracket \right)_F \\ &= \int\limits_F \frac{1}{2} \sigma_\mu(\beta, n) \Big((v_1 - \Pi_h v_1)^2 - 2(v_1 - \Pi_h v_1)(v_2 - \Pi_h v_2) + (v_2 - \Pi_h v_2)^2 \Big) ds \\ &\overset{(6.8)}{\leq} \int\limits_F C \frac{1}{2} \sigma_\mu(\beta, n) \Big((v_1 - \Pi_h v_1)^2 + (v_2 - \Pi_h v_2)^2 \Big) ds \\ &\leq C \int\limits_F (v_1 - \Pi_h v_1)^2 + (v_2 - \Pi_h v_2)^2 ds \\ &= C \Big((v_1 - \Pi_h v_1, v_1 - \Pi_h v_1)_{F(T_1)} + (v_2 - \Pi_h v_2, v_2 - \Pi_h v_2)_{F(T_2)} \Big) . \end{aligned}$$

In the same way with $\frac{1}{2}\sigma_\mu(\beta, n) \leq 1$, Lemma 6.1.3, we get for the boundary faces

$$\left(\frac{1}{2} \sigma_\mu(\beta, n)(v - \Pi_h v), (v - \Pi_h v) \right)_F \leq \int\limits_F (v - \Pi_h v)^2 ds.$$

Altogether, we estimate the first term with bilinear form by writing the interior and boundary faces in terms of the boundaries of the the single elements T. Thereafter, we use the relation that the square root of a sum is less or equal a sum of square roots. The third inequality holds, because of the trace inequality (6.9). The last step follows directly form the standard estimates (6.34), (6.35).

$$\begin{aligned} |||(v - \Pi_h v)||| &= \left[b_h(v - \Pi_h v, v - \Pi_h v) + \sum_{T \in \mathbb{T}_h} h \left\| \nabla (v - \Pi_h v) \right\|_T^2 \right]^{\frac{1}{2}} \\ &\leq \left[C \sum_{T \in \mathbb{T}_h} \| v - \Pi_h v \|_{\partial T}^2 + \sum_{T \in \mathbb{T}_h} h \left\| \nabla (v - \Pi_h v) \right\|_T^2 \right]^{\frac{1}{2}} \end{aligned}$$

$$
\begin{aligned}
&\leq C\left[\sum_{T\in\mathbb{T}_h}\|v-\Pi_h v\|_{\partial T}+\sum_{T\in\mathbb{T}_h}h^{\frac{1}{2}}\left\|\nabla(v-\Pi_h v)\right\|_T\right]\\
&\overset{(6.9)}{\leq} C\left[\sum_{T\in\mathbb{T}_h}h^{-\frac{1}{2}}\|v-\Pi_h v\|_T+\sum_{T\in\mathbb{T}_h}h^{\frac{1}{2}}\left\|\nabla(v-\Pi_h v)\right\|_T\right]\\
&\overset{(6.34),(6.35)}{\leq} Ch^{k+\frac{1}{2}}|v|_{\mathcal{H}^{k+1}(\mathbb{T})}.
\end{aligned}
$$

□

6.2.5. Estimate for bilinear form and error estimate in the energy norm

6.2.7 Theorem. *Let $y\in\mathcal{H}^{k+1}(\mathbb{T}_h)$. Suppose y solves equation (6.1) and $y_h\in V_h$ solves equation (6.2). Then the error $y-y_h$ admits the following estimate*

$$|||y-y_h|||\leq Ch^{k+\frac{1}{2}}|y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)},$$

with C independent of the mesh size h.

Proof. As before we mainly follow the proofs in [45, Theorem 5.1] and [60, Theorem 1.4.13]. We begin the proof by applying the triangle inequality to the energy norm

$$|||y-y_h|||\leq|||y-\Pi_h y|||+|||y_h-\Pi_h y|||.$$

For the first term, we use the error estimate for the $\mathcal{L}^2$-projection of the previous paragraph Lemma 6.2.6, the second term is estimated now. The consistent discretization, see Lemma 6.2.2, implies Galerkin orthogonality

$$b_h(y-y_h,v_h)=0\quad\text{for all } v_h\in V_h. \tag{6.37}$$

With the stability result Lemma 6.2.5 applied to $y_h-\Pi_h y\in V_h$ and the Galerkin orthogonality (6.37) we have

$$
\begin{aligned}
C|||y_h-\Pi_h y|||^2 &\leq b_h(y_h-\Pi_h y, C_2(y_h-\Pi_h y)+h\beta\cdot\nabla(y_h-\Pi_h y)) &(6.38)\\
&\overset{(6.37)}{=} b_h(y-\Pi_h y, C_2(y_h-\Pi_h y)+h\beta\cdot\nabla(y_h-\Pi_h y)). &(6.39)
\end{aligned}
$$

For abbreviation, we set $z:=y-\Pi_h y$ and $w:=y_h-\Pi_h y$. We split equation (6.39) in two terms and estimate them separately. There are two formulations of the bilinear form $b_h(y_h, v_h)$. For the first term we use the formulation of equation (6.2b). In addition we use that $h^{-\frac{1}{2}} \cdot h^{\frac{1}{2}} = h^0 = 1$ and the symmetry of the $\mathcal{L}^2$-scalar product. The first term then reads

$$\begin{aligned} b_h(z, C_2w) = &\sum_{T\in\mathbb{T}_h} \left(h^{-\frac{1}{2}}z, -h^{\frac{1}{2}}\beta \cdot \nabla C_2w\right)_T \\ &+ \sum_{F\in\mathbb{F}_h^{int}} (\{\!\{z\}\!\}, 2\{\!\{C_2w\,\beta\cdot n\}\!\})_F + \sum_{F\in\mathbb{F}_h^{int}} \left(\llbracket z\rrbracket, \frac{1}{2}\sigma_\mu(\beta,n)\llbracket C_2w\rrbracket\right)_F \\ &+ \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(z, \frac{1}{2}\beta\cdot nC_2w\right)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(z, \frac{1}{2}\sigma_\mu(\beta,n)C_2w\right)_F. \end{aligned}$$

For the second term we use the reformulation of the bilinear form in Lemma 6.1.5 and apply the same idea of splitting h in two portions

$$\begin{aligned} b_h(z, h\beta\cdot\nabla w) = &\sum_{T\in\mathbb{T}_h} \left(h^{\frac{1}{2}}\beta\cdot\nabla z, h^{\frac{1}{2}}\beta\cdot\nabla w\right)_T \\ &+ \sum_{F\in\mathbb{F}_h^{int}} \left(-\frac{1}{2}\llbracket\beta\cdot nz\rrbracket, \llbracket h\beta\cdot\nabla w\rrbracket\right)_F + \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_\mu(\beta,n)\llbracket z\rrbracket, \llbracket h\beta\cdot\nabla w\rrbracket\right)_F \\ &+ \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(-\frac{1}{2}\beta\cdot nz, h\beta\cdot\nabla w\right)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(\frac{1}{2}\sigma_\mu(\beta,n)z, h\beta\cdot\nabla w\right)_F. \end{aligned}$$

First we apply the Cauchy-Schwarz inequality (6.6), and get

$$\begin{aligned} &b_h(z, C_2w + h\beta\cdot\nabla w) \\ =&b_h(z, C_2w) + b_h(z, h\beta\cdot\nabla w) \\ \overset{CS}{\leq}& \sum_{T\in\mathbb{T}_h} \left\|h^{-\frac{1}{2}}z\right\|_T \cdot \left\|-h^{\frac{1}{2}}\beta\cdot\nabla C_2w\right\|_T \\ &+ \sum_{F\in\mathbb{F}_h^{int}} \|\{\!\{z\}\!\}\|_F \cdot \|2\{\!\{C_2w\,\beta\cdot n\}\!\}\|_F + \sum_{F\in\mathbb{F}_h^{int}} \|\llbracket z\rrbracket\|_F \cdot \left\|\frac{1}{2}\sigma_\mu(\beta,n)\llbracket C_2w\rrbracket\right\|_F \\ &+ \sum_{F\in\mathbb{F}_h^{\Gamma}} \|z\|_F \cdot \left\|\frac{1}{2}\beta\cdot nC_2w\right\|_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} \|z\|_F \cdot \left\|\frac{1}{2}\sigma_\mu(\beta,n)C_2w\right\|_F \\ &+ \sum_{T\in\mathbb{T}_h} \left\|h^{\frac{1}{2}}\beta\cdot\nabla z\right\|_T \cdot \left\|h^{\frac{1}{2}}\beta\cdot\nabla w\right\|_T \\ &+ \sum_{F\in\mathbb{F}_h^{int}} \left\|-\frac{1}{2}\llbracket\beta\cdot nz\rrbracket\right\|_F \cdot \|\llbracket h\beta\cdot\nabla w\rrbracket\|_F + \sum_{F\in\mathbb{F}_h^{int}} \left\|\frac{1}{2}\sigma_\mu(\beta,n)\llbracket z\rrbracket\right\|_F \cdot \|\llbracket h\beta\cdot\nabla w\rrbracket\|_F \end{aligned}$$

$$+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left\|-\frac{1}{2}\beta\cdot nz\right\|_F\cdot\|h\beta\cdot\nabla w\|_F+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left\|\frac{1}{2}\sigma_\mu(\beta,n)z\right\|_F\cdot\|h\beta\cdot\nabla w\|_F\,.$$

In the next step, we use the inequality of Schwarz (6.7) and sort the single terms to estimate them step by step in the course of this proof

$$\begin{aligned}
&b_h(z,C_2w+h\beta\cdot\nabla w)\\
\leq{}&C\Bigg[\sum_{F\in\mathbb{F}_h^{int}}\left\|\frac{1}{2}\sigma_\mu(\beta,n)\,[\![z]\!]\right\|_F^2+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left\|\frac{1}{2}\sigma_\mu(\beta,n)z\right\|_F^2 && (T_5)\\
&+\sum_{F\in\mathbb{F}_h^{int}}\left\|-\frac{1}{2}\,[\![\beta\cdot nz]\!]\right\|_F^2+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left\|-\frac{1}{2}\beta\cdot nz\right\|_F^2 && (T_6)\\
&+\sum_{T\in\mathbb{T}_h}\left\|h^{\frac{1}{2}}\beta\cdot\nabla z\right\|_T^2 && (T_7)\\
&+\sum_{T\in\mathbb{T}_h}\left\|h^{-\frac{1}{2}}z\right\|_T^2+\sum_{F\in\mathbb{F}_h^{int}}\|\{\!\{z\}\!\}\|_F^2+\sum_{F\in\mathbb{F}_h^{int}}\|[\![z]\!]\|_F^2+2\sum_{F\in\mathbb{F}_h^{\Gamma}}\|z\|_F^2\Bigg]^{\frac{1}{2}} && (T_8)\\
&\cdot\Bigg[\sum_{F\in\mathbb{F}_h^{int}}\left\|\frac{1}{2}\sigma_\mu(\beta,n)\,[\![C_2w]\!]\right\|_F^2+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left\|\frac{1}{2}\sigma_\mu(\beta,n)C_2w\right\|_F^2 && (T_9)\\
&+\sum_{F\in\mathbb{F}_h^{int}}\|2\{\!\{C_2w\,\beta\cdot n\}\!\}\|_F^2+\sum_{F\in\mathbb{F}_h^{\Gamma}}\left\|\frac{1}{2}\beta\cdot nC_2w\right\|_F^2 && (T_{10})\\
&+\sum_{T\in\mathbb{T}_h}\left\|-h^{\frac{1}{2}}\beta\cdot\nabla C_2w\right\|_T^2+\sum_{T\in\mathbb{T}_h}\left\|h^{\frac{1}{2}}\beta\cdot\nabla w\right\|_T^2\\
&+2\sum_{F\in\mathbb{F}_h^{int}}\|[\![h\beta\cdot\nabla w]\!]\|_F^2+2\sum_{F\in\mathbb{F}_h^{\Gamma}}\|h\beta\cdot\nabla w\|_F^2\Bigg]^{\frac{1}{2}} && (T_{11})\\
=:{}&C\Big[T_5+T_6+T_7+T_8\Big]^{\frac{1}{2}}\cdot\Big[T_9+T_{10}+T_{11}\Big]^{\frac{1}{2}}.
\end{aligned}$$

One row corresponds to one variable $T_i, i=5,...,10$, the last two rows are summarized in T_{11} .

We first estimate the terms T_5 and T_9, which are the same terms only differing in the variable z and w, respectively. Assume that $\frac{1}{2}\sigma_\mu(\beta,n)$ is bounded from above by κ.

Then use (6.18c) to obtain

$$T_5 = \sum_{F\in\mathbb{F}_h^{int}} \left\| \frac{1}{2}\sigma_\mu(\beta,n) [\![z]\!] \right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\| \frac{1}{2}\sigma_\mu(\beta,n) z \right\|_F^2 \leq \kappa b_h(z,z) \tag{6.41}$$

$$T_9 = \sum_{F\in\mathbb{F}_h^{int}} \left\| \frac{1}{2}\sigma_\mu(\beta,n) [\![C_2 w]\!] \right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\| \frac{1}{2}\sigma_\mu(\beta,n) C_2 w \right\|_F^2 \leq \kappa C_2^2 b_h(w,w). \tag{6.42}$$

The next two terms are T_6 and T_{10}. The term T_6 reads

$$T_6 = \sum_{F\in\mathbb{F}_h^{int}} \left\| -\frac{1}{2} [\![\beta\cdot nz]\!] \right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\| -\frac{1}{2}\beta\cdot nz \right\|_F^2 .$$

For the sum over the interior faces we get with the same arguments as for the term T_3 in the proof of Lemma 6.2.5, see the equations (6.30) to (6.32),

$$\begin{aligned}\frac{1}{4}([\![\beta\cdot nz]\!], [\![\beta\cdot nz]\!])_F &= (\{\!\{\beta\cdot nz\}\!\}, \{\!\{\beta\cdot nz\}\!\})_F \\ &= \frac{1}{4}(\beta\cdot n)^2 \left\| [\![z]\!] \right\|^2 \\ &\leq \frac{1}{2}|\beta\cdot n| \left\| [\![z]\!] \right\|^2 \\ &\leq \left(\frac{1}{2}\sigma_\mu(\beta,n) [\![z]\!], [\![z]\!] \right).\end{aligned} \tag{6.43}$$

Applying the same arguments for the boundary face sum, it follows for T_6

$$\begin{aligned} T_6 &= \sum_{F\in\mathbb{F}_h^{int}} \left\| -\frac{1}{2} [\![\beta\cdot nz]\!] \right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\| -\frac{1}{2}\beta\cdot nz \right\|_F^2 \\ &\leq \sum_{F\in\mathbb{F}_h^{int}} \left(\frac{1}{2}\sigma_\mu(\beta,n) [\![z]\!], [\![z]\!] \right)_F + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left(\frac{1}{2}\sigma_\mu(\beta,n) z, z \right)_F \\ &\overset{(6.18c)}{=} b_h(z,z). \end{aligned} \tag{6.44}$$

Taking a look at T_{10}

$$T_{10} = \sum_{F\in\mathbb{F}_h^{int}} \left\| 2\{\!\{C_2 w\, \beta\cdot n\}\!\} \right\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\| \frac{1}{2}\beta\cdot nC_2 w \right\|_F^2 , \tag{6.45}$$

we notice, that we have an average term, which equals the jumps in (6.43) or (6.31),

$$(2\{\!\{C_2 w\beta \cdot n\}\!\}, 2\{\!\{C_2 w\beta \cdot n\}\!\})_F = ([\![C_2 w\beta \cdot n]\!], [\![C_2 w\beta \cdot n]\!])_F.$$

Thus again with the same arguments, for the term T_{10} it holds

$$\begin{aligned} T_{10} &= \sum_{F\in\mathbb{F}_h^{int}} \|2\{\!\{C_2 w\,\beta \cdot n\}\!\}\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\| \frac{1}{2}\beta \cdot n C_2 w \right\|_F^2 \\ &\leq \sum_{F\in\mathbb{F}_h^{int}} C_2^2 (\sigma_\mu(\beta, n)\, [\![w]\!], [\![w]\!])_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} C_2^2 \left(\sigma_\mu(\beta, n) w, w\right)_F \\ &\leq 2C_2^2 b_h(w, w). \end{aligned} \tag{6.46}$$

After this we estimate the terms T_7 and T_{11}

$$T_7 = \sum_{T\in\mathbb{T}_h} \left\| h^{\frac{1}{2}} \beta \cdot \nabla z \right\|_T^2, \tag{6.47}$$

$$T_{11} = \sum_{T\in\mathbb{T}_h} \left\| -h^{\frac{1}{2}} \beta \cdot \nabla C_2 w \right\|_T^2 + \sum_{T\in\mathbb{T}_h} \left\| h^{\frac{1}{2}} \beta \cdot \nabla w \right\|_T^2 \tag{6.48}$$

$$+ 2 \sum_{F\in\mathbb{F}_h^{int}} \|[\![h\beta \cdot \nabla w]\!]\|_F^2 + 2 \sum_{F\in\mathbb{F}_h^{\Gamma}} \|h\beta \cdot \nabla w\|_F^2 . \tag{6.49}$$

Similarly as in equations (6.27) -(6.29) we apply the trace inequality (6.11) to the face terms of T_{11}

$$\begin{aligned} 2 \sum_{F\in\mathbb{F}_h^{int}} \|[\![h\beta \cdot \nabla w]\!]\|_F^2 + 2 \sum_{F\in\mathbb{F}_h^{\Gamma}} \|h\beta \cdot \nabla w\|_F^2 &\leq h^2 \sum_{T\in\mathbb{T}_h} \|\beta \cdot \nabla w\|_{\partial T}^2 \\ &\overset{(6.11)}{\leq} hC \sum_{T\in\mathbb{T}_h} \|\beta \cdot \nabla w\|_T^2 . \end{aligned}$$

Thus, with this trace inequality we have for the entire term T_{11}

$$T_{11} = \sum_{T\in\mathbb{T}_h} \left\| -h^{\frac{1}{2}} \beta \cdot \nabla C_2 w \right\|_T^2 + \sum_{T\in\mathbb{T}_h} \left\| h^{\frac{1}{2}} \beta \cdot \nabla w \right\|_T^2 \tag{6.50}$$

$$+ 2 \sum_{F\in\mathbb{F}_h^{int}} \|[\![h\beta \cdot \nabla w]\!]\|_F^2 + 2 \sum_{F\in\mathbb{F}_h^{\Gamma}} \|h\beta \cdot \nabla w\|_F^2 \tag{6.51}$$

$$\leq \sum_{T\in\mathbb{T}_h} \left\| -h^{\frac{1}{2}} \beta \cdot \nabla C_2 w \right\|_T^2 + \sum_{T\in\mathbb{T}_h} \left\| h^{\frac{1}{2}} \beta \cdot \nabla w \right\|_T^2 \tag{6.52}$$

$$+ hC \sum_{T\in\mathbb{T}_h} \|\beta \cdot \nabla w\|_T^2 \tag{6.53}$$

$$\leq (C_2^2 + 1 + C) \sum_{T\in\mathbb{T}_h} h \|\beta \cdot \nabla w\|_T^2 . \tag{6.54}$$

The next step is summing up the estimates for T_5, equation (6.41), T_6, equation (6.44) and T_7, equation (6.47), which belong to the first pair of parenthesis of (6.40)

$$T_5 + T_6 + T_7 \leq \kappa b_h(z,z) + 2b_h(z,z) + \sum_{T\in\mathbb{T}_h} h \|\beta \cdot \nabla z\|_T^2 \tag{6.55}$$

$$\leq (\kappa C + 2C) \cdot \left(b_h(z,z) + \sum_{T\in\mathbb{T}_h} h \|\beta \cdot \nabla z\|_T^2 \right) \tag{6.56}$$

$$= C |||z|||^2. \tag{6.57}$$

For the energy norm of z we already have an estimate, because z was defined as the projection error $z = y - \Pi_h y$. With the error estimate for the $\mathcal{L}^2$-projection in Lemma (6.2.6) we have

$$T_5 + T_6 + T_7 \leq C |||z|||^2 \tag{6.58}$$

$$\leq C \left(h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \right)^2 . \tag{6.59}$$

For the second pair of parenthesis in (6.40) we sum up the estimates of T_9, equation (6.42), T_{10}, equation (6.46), and T_{11}, equation (6.54)

$$T_9 + T_{10} + T_{11} \leq \kappa C_2^2 b_h(w,w) + 2C_2^2 b_h(w,w) + (C_2^2 + 1 + C) \sum_{T\in\mathbb{T}_h} h \|\beta \cdot \nabla w\|_T^2 \tag{6.60}$$

$$\leq C \left(b_h(w,w) + \sum_{T\in\mathbb{T}_h} h \|\beta \cdot \nabla w\|_T^2 \right) \tag{6.61}$$

$$= C |||w|||^2. \tag{6.62}$$

Term T_8 is left

$$T_8 = \sum_{T\in\mathbb{T}_h} \left\| h^{-\frac{1}{2}} z \right\|_T^2 + \sum_{F\in\mathbb{F}_h^{int}} \|\{\!\{z\}\!\}\|_F^2 + \sum_{F\in\mathbb{F}_h^{int}} \|[\![z]\!]\|_F^2 + 2 \sum_{F\in\mathbb{F}_h^{\Gamma}} \|z\|_F^2 . \tag{6.63}$$

We apply the trace inequality (6.9) to relate the face terms of T_8 to the cell terms. Therefore we rewrite the face terms in terms of cell boundaries. With the triangle inequality we get for the jump term

$$\|[\![z]\!]\|_T^2 = \|z_1 - z_2\|^2 \leq \|z_1\|^2 + \|z_2\|^2 .$$

This yields

$$\sum_{F\in\mathbb{F}_h^{int}} \|[\![z]\!]\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \|z\|_F^2 = \sum_{F\in\mathbb{F}_h^{int}} \|z_1 - z_2\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \|z\|_F^2 \leq \sum_{F\in\mathbb{T}} \|z\|_{\partial T}^2 .$$

For the average term we get in the same way with triangle inequality

$$\|\{\!\{z\}\!\}\|_T^2 = \frac{1}{4}\|z_1 + z_2\|^2 \leq \|z_1\|^2 + \|z_2\|^2$$

and hence

$$\sum_{F\in\mathbb{F}_h^{int}} \|\{\!\{z\}\!\}\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \|z\|_F^2 = \sum_{F\in\mathbb{F}_h^{int}} \frac{1}{4}\|z_1 + z_2\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \|z\|_F^2 \leq \sum_{F\in\mathbb{T}_h} \|z\|_{\partial T}^2 ,$$

since $\frac{1}{4} \leq 1$. In total we get by applying the trace inequality (6.9) to the cell boundary term

$$\begin{aligned} T_8 &\leq \sum_{T\in\mathbb{T}_h} h^{-1}\|z\|_T^2 + 2\sum_{T\in\mathbb{T}_h} \|z\|_{\partial T}^2 && (6.64)\\ &\overset{(6.9)}{\leq} \sum_{T\in\mathbb{T}_h} h^{-1}\|z\|_T^2 + 2C\sum_{T\in\mathbb{T}_h} h\|\nabla z\|_T^2 + 2C\sum_{T\in\mathbb{T}_h} h^{-1}\|z\|_T^2 && (6.65)\\ &\leq C\Big(\sum_{T\in\mathbb{T}_h} h^{-1}\|z\|_T^2 + \sum_{T\in\mathbb{T}_h} h\|\nabla z\|_T^2\Big). && (6.66)\end{aligned}$$

Because z is defined by $z = y - \mathit{\Pi}_h y$, we use the standard estimates for the $\mathcal{L}^2$-projection (6.34) and (6.35) in the form

$$h^{-\frac{1}{2}}\sum_{T\in\mathbb{T}_h} \|z\|_T \leq Ch^{k+\frac{1}{2}}|y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}, \tag{6.67}$$

and

$$h^{\frac{1}{2}}\sum_{T\in\mathbb{T}_h} \|\nabla z\|_T \leq Ch^{k+\frac{1}{2}}|y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}. \tag{6.68}$$

Squaring and after that summing of (6.67) and (6.68) leads to the estimate

$$h^{-1} \sum_{T \in \mathbb{T}_h} \|z\|_T^2 + h \sum_{T \in \mathbb{T}_h} \|\nabla z\|_T^2 \leq C \left(h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \right)^2 .$$

Equation (6.66) thus reads

$$T_8 \leq C \Big(\sum_{T \in \mathbb{T}_h} h^{-1} \|z\|_T^2 + \sum_{T \in \mathbb{T}_h} h \|\nabla z\|_T^2 \Big) \leq C \left(h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \right)^2 . \tag{6.69}$$

Putting all pieces together, equation (**??**) reads with the estimates (6.59), (6.69), (6.62)

$$\begin{aligned} b_h(z, C_2 w + h\beta \cdot \nabla w) &\leq C \Big[T_5 + T_6 + T_7 + T_8 \Big]^{\frac{1}{2}} \cdot \Big[T_9 + T_{10} + T_{11} \Big]^{\frac{1}{2}} \\ &\leq C \Big[C \left(h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \right)^2 + C \left(h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \right)^2 \Big]^{\frac{1}{2}} \cdot \Big[C |||w|||^2 \Big]^{\frac{1}{2}} \\ &\leq C \Big[h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \Big] \cdot |||w|||. \end{aligned}$$

Altogether, with our abbreviations $z = y - \Pi_h y$ and $w = y_h - \Pi_h y$, the error estimate of the discretization error in the energy norm in Theorem 6.2.7 follows from

$$\begin{aligned} C |||y_h - \Pi_h y|||^2 &\overset{(6.39)}{\leq} b_h(y - \Pi_h y, C_2(y_h - \Pi_h y) + h\beta \cdot \nabla (y_h - \Pi_h y)) \\ &\leq C h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} |||y_h - \Pi_h y||| \\ \Leftrightarrow \quad C |||y_h - \Pi_h y||| &\leq C h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}. \end{aligned}$$

□

6.2.6. Superconvergence result

6.2.8 Corollary. *Let the grid be a Cartesian grid and the polynomial space $\mathcal{P}(T)$ the space of piecewise polynomials of degree k. Let $y \in \mathcal{H}^{k+2}(\mathbb{T}_h)$. Suppose y solves equation (6.1) and $y_h \in V_h$ solves equation (6.2). Then the error $y - y_h$ admits the following estimate in the $\mathcal{L}^2$ norm*

$$\|y - y_h\| \leq C h^{k+1} |y|_{\mathcal{H}^{k+2}(\mathbb{T}_h)},$$

with C independent of the mesh size h.

Proof. For a proof of this result see [74, Theorem 6]. □

6.3. Diffusion advection reaction

In this section, we show, that the results of the previous subsections also hold for the diffusion advection reaction problem, Example 2.1.5. In this case, we have to adjust the test functions [11], [45]. The diffusion advection reaction PDE reads

$$-\nabla \cdot (\alpha \nabla y) + \beta \cdot \nabla y + \rho y = f \qquad \text{on } \Omega \tag{6.70a}$$

$$y = y_D \qquad \text{on } \mathit{\Gamma}. \tag{6.70b}$$

The corresponding discrete problem is, see (4.7), find $y_h \in V_h$ such that

$$\alpha a_h(y_h, v_h) + b_h(y_h, v_h) + \rho c_h(y_h, v_h) = f_h(v_h), \quad \forall v_h \in V_h. \tag{6.71}$$

Here we use the same discretization as in the previous sections with the proposed differentiable stabilization $\sigma_\mu(\beta, n)$ for the advection and the standard discretization as depicted in Chapter 4 for the diffusion part $a_h(y_h, v_h)$ and the reaction part $c_h(y_h, v_h)$. The advection vector β is normalized, $\|\beta\| = 1$. We define a different energy norm for this diffusion advection reaction problem [11], [60]:

$$|||y|||_{dar}^2 := \alpha |||y|||_d^2 + |||y|||^2 + \rho |||y|||_r^2. \tag{6.72}$$

This energy norm consists of the energy norms of the three parts. The advection norm $|||y|||$ is the same as in (6.19). The diffusion norm is defined as [60]

$$|||y|||_d^2 := \sum_{T \in \mathbb{T}} \|\nabla y\|_T^2 + \sum_{F \in \mathbb{F}_h^{int}} \|\sqrt{\gamma}\, [\![y]\!]\|_F^2 + \sum_{F \in \mathbb{F}_h^{\mathit{\Gamma}}} \left\| \sqrt{2\gamma} y \right\|_F^2, \tag{6.73}$$

and the energy norm of the reaction part is

$$|||y|||_r^2 := \sum_{T \in \mathbb{T}} \|y\|_T^2 .$$

With these definitions we state an estimate for the $\mathcal{L}^2$-projection and a stability estimate. After that we derive an error estimate in the energy norm for the diffusion advection reaction problem.

6.3.1 Lemma. *The error of the $\mathcal{L}^2$-projection in the energy norm for any function $v \in \mathcal{H}^{k+1}(\mathbb{T}_h)$ can be estimated by*

$$|||v - \Pi_h v|||_{dar} \leq C \max(\sqrt{\alpha} h^k, h^{k+\frac{1}{2}}, \sqrt{\rho} h^{k+1}) |v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}.$$

Proof. To proof Lemma 6.3.1 we take a look at the separate parts. An estimate for the diffusion part is found in [9], [10]:

$$|||v - \Pi_h v|||_d \leq C h^k |v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}. \tag{6.74}$$

For the reaction part we use the standard estimate for $\mathcal{L}^2$-projection (6.34)

$$|||v - \Pi_h v|||_r = \sum_{T \in \mathbb{T}_h} \|v - \Pi_h v\|_T \leq C h^{k+1} |v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}. \tag{6.75}$$

Summing up the estimates for diffusion (6.74), advection in Lemma 6.2.6 and reaction (6.75), we arrive at the statement

$$\begin{aligned} |||v - \Pi_h v|||_{dar} &\leq \sqrt{\alpha} |||v - \Pi_h v|||_d + |||v - \Pi_h v||| + \sqrt{\rho} |||v - \Pi_h v|||_r \\ &\leq \sqrt{\alpha} C h^k |v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} + C h^{k+\frac{1}{2}} |v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} + \sqrt{\rho} C h^{k+1} |v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \\ &\leq C \max(\sqrt{\alpha} h^k, h^{k+\frac{1}{2}}, \sqrt{\rho} h^{k+1}) |v|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}, \end{aligned}$$

see also [45] for diffusion advection. □

In contrast to the standard approximation results for the diffusion and reaction, where for the stability $v := y$ is used as a test function, in the diffusion advection reaction case we have to use augmented test functions, which we used for the advection $v_h := C y_h + h\beta \cdot \nabla y_h$, see Lemma 6.2.5. This is because of the additional term in the energy norm of the advection, which is important when the advection term dominates. In the same manner as in the stability proof of the advection Lemma 6.2.5 we have to show that for all $y_h \in V_h$ there exists a $v_h \in V_h$ such that

$$\begin{aligned} &\alpha a_h(y_h, C y_h + h\beta \cdot \nabla y_h) \\ &+ b_h(y_h, C y_h + h\beta \cdot \nabla y_h) \\ &+ \rho c_h(y_h, C y_h + h\beta \cdot \nabla y_h) \geq C |||y_h|||_{dar} |||v_h|||_{dar} \end{aligned}$$

holds with $C > 0$ independent of h. We proceed in the same manner as in the stability proofs of the advection part.

6.3.2 Lemma. *For the diffusion advection reaction energy norm (6.72) with augmented test functions, it holds*

$$\begin{aligned}|||Cy_h + h\beta \cdot \nabla y_h|||_{dar} &\leq C|||y_h|||_{dar} + |||h\beta \cdot \nabla y_h|||_{dar} \\ &\leq C|||y_h|||_{dar} + C|||y_h|||_{dar} \\ &\leq C|||y_h|||_{dar}.\end{aligned}$$

Proof. The first inequality holds, because of triangle inequality. We now prove the second inequality, by showing

$$|||h\beta \cdot \nabla y_h|||_{dar} \leq C|||y_h|||_{dar}.$$

We begin with the reaction part of the diffusion advection reaction energy norm. With the Cauchy-Schwarz inequality, $\|\beta\|^2 = 1$ and the inverse estimate (6.12) it holds

$$\begin{aligned}|||h\beta \cdot \nabla y_h|||_r^2 &= \sum_{T\in\mathbb{T}} \|h\beta \cdot \nabla y_h\|_T^2 \\ &\overset{CS}{\leq} \sum_{T\in\mathbb{T}} \|\beta\|^2 \|h\nabla y_h\|_T^2 \\ &\overset{(6.12)}{\leq} C\sum_{T\in\mathbb{T}} \|y_h\|_T^2 = C|||y_h|||_r^2.\end{aligned}$$

Furthermore we estimate the diffusion part in the same manner as we did for the advection part in proof of Lemma 6.2.4. The diffusion part reads

$$\begin{aligned}&|||h\beta \cdot \nabla y_h|||_d^2 \\ &= \sum_{T\in\mathbb{T}} \|\nabla(h\beta \cdot \nabla y_h)\|_T^2 + \sum_{F\in\mathbb{F}_h^{int}} \|\sqrt{\gamma}\,[\![h\beta \cdot \nabla y_h]\!]\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\|\sqrt{2\gamma} h\beta \cdot \nabla y_h\right\|_F^2.\end{aligned}$$

We first reformulate the sums over the faces. In the same manner as in equation (6.22) we get with triangle inequality

$$\begin{aligned}\|\sqrt{\gamma}\,[\![h\beta \cdot \nabla y_h]\!]\|_F^2 &= h^2 \|\sqrt{\gamma} h\beta \cdot \nabla y_1 - \sqrt{\gamma} h\beta \cdot \nabla y_2\|^2 \\ &\leq h^2 \left(\|\sqrt{\gamma} h\beta \cdot \nabla y_1\|^2 + \|\sqrt{\gamma} h\beta \cdot \nabla y_2\|^2\right).\end{aligned}$$

Thus we relate the face terms to the boundary of the elements T

$$\begin{aligned}&\sum_{F\in\mathbb{F}_h^{int}} \|\sqrt{\gamma}\,[\![h\beta\cdot\nabla y_h]\!]\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\|\sqrt{2\gamma}h\beta\cdot\nabla y_h\right\|_F^2\\ &\leq Ch^2\sum_{T\in\mathbb{T}} \|\sqrt{\gamma}\beta\cdot\nabla y_h\|_{\partial T}^2 .\end{aligned}$$

Now we apply the trace inequality (6.10) and get

$$Ch^2\sum_{T\in\mathbb{T}} \|\sqrt{\gamma}\beta\cdot\nabla y_h\|_{\partial T}^2 \leq Ch\sum_{T\in\mathbb{T}} \|\sqrt{\gamma}\beta\cdot\nabla y_h\|_T^2 .$$

Secondly we reformulate the cell part. We use the inverse estimate (6.12) and the Cauchy-Schwarz inequality

$$\begin{aligned}\sum_{T\in\mathbb{T}} \|\nabla(h\beta\cdot\nabla y_h)\|_T^2 &\overset{(6.12)}{\leq} C\sum_{T\in\mathbb{T}} \|\beta\cdot\nabla y_h\|_T^2\\ &\overset{CS}{\leq} C\sum_{T\in\mathbb{T}} \|\beta\|^2\,\|\nabla y_h\|_T^2 .\end{aligned}$$

The estimates for cell and face part together thus read

$$\begin{aligned}&\sum_{T\in\mathbb{T}} \|\nabla(h\beta\cdot\nabla y_h)\|_T^2 + \sum_{F\in\mathbb{F}_h^{int}} \|\sqrt{\gamma}\,[\![h\beta\cdot\nabla y_h]\!]\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\|\sqrt{2\gamma}h\beta\cdot\nabla y_h\right\|_F^2\\ &\leq C\sum_{T\in\mathbb{T}} \|\nabla y_h\|_T^2 + Ch\sum_{T\in\mathbb{T}} \|\sqrt{\gamma}\beta\cdot\nabla y_h\|_T^2 .\end{aligned}$$

The inequality still holds, if we add face terms to arrive at the energy norm for the diffusion part

$$\begin{aligned}&C\sum_{T\in\mathbb{T}} \|\nabla y_h\|_T^2 + Ch\sum_{T\in\mathbb{T}} \|\sqrt{\gamma}\beta\cdot\nabla y_h\|_T^2\\ &\leq C\sum_{T\in\mathbb{T}} \|\nabla y_h\|_T^2 + Ch\sum_{T\in\mathbb{T}} \|\sqrt{\gamma}\beta\cdot\nabla y_h\|_T^2 + \sum_{F\in\mathbb{F}_h^{int}} \|\sqrt{\gamma}\,[\![y_h]\!]\|_F^2 + \sum_{F\in\mathbb{F}_h^{\Gamma}} \left\|\sqrt{2\gamma}y_h\right\|_F^2\\ &= Ch\sum_{T\in\mathbb{T}} \|\sqrt{\gamma}\beta\cdot\nabla y_h\|_T^2 + C|\!|\!|y|\!|\!|_d^2.\end{aligned}$$

For the advection part it holds with equation (6.24) in the course of the proof of

Lemma 6.2.4

$$\begin{aligned}&|||h\beta \cdot \nabla y_h|||^2\\ &\leq C\Big[b_h(y_h, y_h) + \sum_{T\in\mathbb{T}_h} h\,\|\beta \cdot \nabla y_h\|_T^2\Big].\end{aligned}$$

We get for the overall diffusion advection reaction energy norm, where the remaining term of the diffusion estimate enters the advection energy norm

$$\begin{aligned}&|||h\beta \cdot \nabla y_h|||_{dar}^2\\ =&\alpha|||h\beta \cdot \nabla y_h|||_d^2 + |||h\beta \cdot \nabla y_h|||^2 + \rho|||h\beta \cdot \nabla y_h|||_r^2\\ \leq&\alpha Ch\sum_{T\in\mathbb{T}} \|\sqrt{\gamma}\beta \cdot \nabla y_h\|_T^2 + \alpha C|||y|||_d^2\\ &+ Cb_h(y_h, y_h) + C\sum_{T\in\mathbb{T}_h} h\,\|\beta \cdot \nabla y_h\|_T^2 + \rho C|||y_h|||_r^2\\ \leq&\alpha C|||y_h|||_d^2 + C|||y_h|||^2 + \rho C|||y_h|||_r^2\\ =&C|||y_h|||_{dar}^2.\end{aligned}$$

Thus we have

$$|||h\beta \cdot \nabla y_h|||_{dar} \leq C|||y_h|||_{dar}.$$

□

6.3.3 Lemma. *There exists a constant $C > 0$ such that for all $y_h \in V_h$,*

$$\begin{aligned}&\alpha a_h(y_h, Cy_h + h\beta \cdot \nabla y_h)\\ &+ b_h(y_h, Cy_h + h\beta \cdot \nabla y_h)\\ &+ \rho c_h(y_h, Cy_h + h\beta \cdot \nabla y_h) \geq C|||y_h|||_{dar}^2.\end{aligned}$$

Proof. We investigate the three parts separately, because the energy norm $|||.|||_{dar}$ is a sum. For the advection part, the stability estimate is shown in Lemma 6.2.5, which states

$$b_h(y_h, Cy_h + h\beta \cdot \nabla y_h) \geq C|||y_h|||^2. \tag{6.76}$$

For the diffusion part, a proof is found in [45], proof of Lemma A.1. An important step is the use of the inverse inequality and the trace estimate. It concludes

$$\alpha a_h(y_h, Cy_h + h\beta \cdot \nabla y_h) \geq C\alpha |||y_h|||_d^2. \tag{6.77}$$

The stability estimate for the reaction part can be proven by applying the inverse estimate [60, Lemma 5.1.5]

$$\rho c_h(y_h, Cy_h + h\beta \cdot \nabla y_h) \geq C\rho |||y_h|||_r^2. \tag{6.78}$$

Altogether we get

$$\begin{aligned}
&\alpha a_h(y_h, Cy_h + h\beta \cdot \nabla y_h)\\
&\quad + b_h(y_h, Cy_h + h\beta \cdot \nabla y_h)\\
&\quad + \rho c_h(y_h, Cy_h + h\beta \cdot \nabla y_h)\\
\geq& C|||y_h|||^2 + C\alpha|||y_h|||_d^2 + C\rho|||y_h|||_r^2\\
=& C|||y_h|||_{dar}^2.
\end{aligned}$$

□

6.3.4 Theorem. *Let $y \in \mathcal{H}^{k+1}(\mathbb{T}_h)$. Suppose y solves equation (6.70) and $y_h \in V_h$ solves the discrete problem equation (6.71). Then the error $y - y_h$ admits the following estimate*

$$|||y - y_h|||_{dar} \leq C \max(\sqrt{\alpha} h^k, h^{k+\frac{1}{2}}, \sqrt{\rho} h^{k+1}) |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)},$$

with C independent of mesh size h.

Proof. The proof follows the same steps as the proof of Theorem 6.2.7. With the previous Lemmas 6.3.1 and 6.3.3 the estimate in Theorem 6.3.4 follows. First we apply the triangle inequality

$$|||y - y_h|||_{dar} \leq |||y - \Pi_h y|||_{dar} + |||y_h - \Pi_h y|||_{dar}. \tag{6.79}$$

The first part of (6.79) is estimated by the projection estimate in Lemma 6.3.1. The second part is estimated now. In a first step we use the stability estimate of Lemma 6.3.3 and, because the discretization is consistent, the Galerkin orthogonality:

$$\begin{aligned}
|||y_h - \Pi_h y|||_{dar}^2 \leq& \alpha a_h(y_h - \Pi_h y, C(y_h - \Pi_h y) + h\beta \cdot \nabla(y_h - \Pi_h y))\\
&+ b_h(y_h - \Pi_h y, C(y_h - \Pi_h y) + h\beta \cdot \nabla(y_h - \Pi_h y))
\end{aligned}$$

$$\begin{aligned} &+ \rho c_h(y_h - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ =&\alpha a_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ &+ b_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ &+ \rho c_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)). \end{aligned}$$

We need to use the augmented test functions here, too. Thus we have to show

$$\begin{aligned} &\alpha a_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ &+ b_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ &+ \rho c_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ \le& C \max(\sqrt{\alpha} h^k, h^{k+\frac{1}{2}}, \sqrt{\rho} h^{k+1}) |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} |||y_h - \mathit{\Pi}_h y|||_{dar}. \end{aligned} \tag{6.80}$$

Then the error estimate in Theorem 6.3.4 follows with:

$$\begin{aligned} |||y_h - \mathit{\Pi}_h y|||_{dar}^2 \le& \alpha a_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ &+ b_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ &+ \rho c_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \qquad (6.81) \\ \le& C \max(\sqrt{\alpha} h^k, h^{k+\frac{1}{2}}, \sqrt{\rho} h^{k+1}) |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} |||y_h - \mathit{\Pi}_h y|||_{dar} \\ \Leftrightarrow \quad |||y_h - \mathit{\Pi}_h y|||_{dar} \le& C \max(\sqrt{\alpha} h^k, h^{k+\frac{1}{2}}, \sqrt{\rho} h^{k+1}) |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}. \end{aligned}$$

Again, we investigate the three parts, advection, diffusion, reaction, separately. The advection part was proven in Theorem 6.2.7, see (**??**):

$$\begin{aligned} &b_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ &\le C h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} |||y_h - \mathit{\Pi}_h y|||, \\ &\le C h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} |||y_h - \mathit{\Pi}_h y|||_{dar}, \end{aligned} \tag{6.82}$$

the diffusion part is shown in [45], proof of Theorem 5.1,

$$\begin{aligned} &\alpha a_h(y - \mathit{\Pi}_h y, C(y_h - \mathit{\Pi}_h y) + h\beta \cdot \nabla(y_h - \mathit{\Pi}_h y)) \\ &\le C \alpha h^k |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} |||y_h - \mathit{\Pi}_h y|||_d, \\ &\le C \sqrt{\alpha} h^k |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \frac{\sqrt{\alpha} |||y_h - \mathit{\Pi}_h y|||_d}{|||y_h - \mathit{\Pi}_h y|||_{dar}} |||y_h - \mathit{\Pi}_h y|||_{dar}, \\ &\le C \sqrt{\alpha} h^k |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} |||y_h - \mathit{\Pi}_h y|||_{dar}. \end{aligned} \tag{6.83}$$

For the reaction part we have to proof:

$$\begin{aligned}\rho c_h(y - \Pi_h y, C(y_h - \Pi_h y) + h\beta \cdot \nabla(y_h - \Pi_h y)) \\ \leq C\sqrt{\rho}h^{k+1}|y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}|||y_h - \Pi_h y|||_{dar}.\end{aligned} \tag{6.84}$$

This follows by applying the Cauchy-Schwarz inequality (6.6), the inequality of Schwarz (6.7), the projection estimate (6.75) of Lemma 6.3.1 and the inverse estimate (6.12),

$$\begin{aligned}
&\rho c_h(y - \Pi_h y, C(y_h - \Pi_h y) + h\beta \cdot \nabla(y_h - \Pi_h y)) \\
&\quad = \rho \sum_{T\in\mathbb{T}} (y - \Pi_h y, C(y_h - \Pi_h y) + h\beta \cdot \nabla(y_h - \Pi_h y))_T \\
&\quad \overset{(6.6)}{\leq} \rho \sum_{T\in\mathbb{T}} \|y - \Pi_h y\|_T \left(\|C(y_h - \Pi_h y)\|_T + \|h\beta \cdot \nabla(y_h - \Pi_h y)\|_T\right) \\
&\quad \overset{(6.7)}{\leq} \rho \left(\sum_{T\in\mathbb{T}} \|y - \Pi_h y\|_T^2\right)^{\frac{1}{2}} \left(\sum_{T\in\mathbb{T}} \left(\|C(y_h - \Pi_h y)\|_T + \|h\beta \cdot \nabla(y_h - \Pi_h y)\|_T\right)^2\right)^{\frac{1}{2}} \\
&\quad \overset{(6.75)}{\leq} \rho \left(\left(Ch^{k+1}|y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}\right)^2\right)^{\frac{1}{2}} \left(\sum_{T\in\mathbb{T}} \left(\|C(y_h - \Pi_h y)\|_T + \|h\beta \cdot \nabla(y_h - \Pi_h y)\|_T\right)^2\right)^{\frac{1}{2}} \\
&\quad \overset{(6.12)}{\leq} \rho C h^{k+1}|y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \left(\sum_{T\in\mathbb{T}} C\,\|y_h - \Pi_h y\|_T^2\right)^{\frac{1}{2}} \\
&\quad \leq \sqrt{\rho}Ch^{k+1}|y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)} \frac{\sqrt{\rho}|||y_h - \Pi_h y|||_r}{|||y_h - \Pi_h y|||_{dar}} |||y_h - \Pi_h y|||_{dar} \\
&\quad \leq \sqrt{\rho}Ch^{k+1}|y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)}|||y_h - \Pi_h y|||_{dar}
\end{aligned}$$

Altogether, with the estimate for advection (6.82), for diffusion (6.83) and reaction (6.84), equation (6.80) follows and with (6.81) the estimate in Theorem 6.3.4. □

6.4. Non normalized advection coefficient

In the case that Assumption 6.1.1 does not hold, i. e. $\|\beta\| \neq 1$, we have to adjust several ingredients. The main assumption originating from $\|\beta\| = 1$, see for example equations (6.30)-(6.32), is

$$-1 \leq \beta \cdot n \leq 1$$

and thus $(\beta \cdot n)^2 \leq |\beta \cdot n|$. For $\|\beta\| \neq 1$ this equation changes

$$\begin{aligned} & -\|\beta\| \leq \beta \cdot n \leq \|\beta\|\,, \\ \Leftrightarrow \quad & -1 \leq \frac{\beta \cdot n}{\|\beta\|} \leq 1. \end{aligned}$$

With the weighting factor $\frac{1}{\|\beta\|}$, the term belongs always to the interval $[-1, 1]$. Therefore it holds

$$\begin{aligned} & \left(\frac{\beta \cdot n}{\|\beta\|}\right)^2 \leq \frac{|\beta \cdot n|}{\|\beta\|}, \\ \Leftrightarrow \quad & (\beta \cdot n)^2 \leq \|\beta\| |\beta \cdot n|\,. \end{aligned}$$

Check assumptions Let us examine the two remaining Lemmas 6.1.2 and 6.1.3.

Lemma 6.1.2 holds for a non-normalized advection coefficient
6.4.1 Lemma. *The following inequality holds*

$$|\beta \cdot n| \leq \sigma_\mu(\beta, n).$$

Proof. Because of $|\beta \cdot n| \leq \|\beta\|$ it holds $(\beta \cdot n)^2 \leq \|\beta\|^2$. Therefore, the inequality holds:

$$\begin{aligned} \sigma_\mu(\beta, n) &= \frac{\sqrt{(\beta \cdot n)^2 + \|\beta\|^2 \mu^2}}{\sqrt{1 + \mu^2}} \\ &= \sqrt{(\beta \cdot n)^2} \left(\sqrt{\frac{1 + \frac{\|\beta\|^2 \mu^2}{(\beta \cdot n)^2}}{1 + \mu^2}} \right) \geq |\beta \cdot n|\,. \end{aligned}$$

□

An adjusted variant of Lemma 6.1.3 holds as well:
6.4.2 Lemma. *The continuously differentiable stabilization $\sigma_\mu(\beta, n)$ is bounded from above*

$$\frac{1}{\|\beta\|} \sigma_\mu(\beta, n) \leq 1.$$

Proof. This holds true, because

$$\begin{aligned}
& |\beta \cdot n| && \leq \|\beta\| \, \|n\| = \|\beta\| \\
\Rightarrow \quad & (\beta \cdot n)^2 && \leq \|\beta\|^2 \\
\Rightarrow \quad & \frac{(\beta \cdot n)^2}{\|\beta\|^2} + \mu^2 && \leq 1 + \mu^2 \\
\Rightarrow \quad & \sqrt{\frac{(\beta \cdot n)^2}{\|\beta\|^2} + \mu^2} && \leq \sqrt{1 + \mu^2} \\
\Rightarrow \quad & \frac{1}{\|\beta\|} \sigma_\mu(\beta, n) = \frac{1}{\|\beta\|} \frac{\sqrt{(\beta \cdot n)^2 + \|\beta\|^2 \mu^2}}{\sqrt{1+\mu^2}} && \leq 1.
\end{aligned}$$

□

Error estimates By including an additional term $\frac{1}{\|\beta\|}$, we adjust the energy norm

$$|||y|||_\beta^2 := b_h(y, y) + \sum_{T \in \mathbb{T}_h} \frac{h}{\|\beta\|} \|\beta \cdot \nabla y\|_T^2 ,$$

and the augmented test functions $v_{h,\beta} := \frac{1}{\|\beta\|} v_h = \frac{1}{\|\beta\|} C y_h + \frac{h}{\|\beta\|} \beta \cdot \nabla y_h$.

With these adjusted energy norm and augmented test functions, the stability analysis and error estimates of the preceding Lemmas and Theorems hold for $\|\beta\| \neq 1$.

6.4.3 Theorem. *Let $y \in \mathcal{H}^{k+1}(\mathbb{T}_h)$. Suppose y solves equation (6.1) and $y_h \in V_h$ solves equation (6.2). Then the error $y - y_h$ admits the following estimate*

$$|||y - y_h|||_\beta \leq C_\beta h^{k+\frac{1}{2}} |y|_{\mathcal{H}^{k+1}(\mathbb{T}_h)},$$

with C_β independent of the mesh size h, but it depends on $\|\beta\|$.

Proof. We do not show all steps in detail, because the proof proceeds in the same way as in the preceding Lemmas and Theorems. As mentioned before, now the energy norm and the augmented test functions are weighted with $\frac{1}{\|\beta\|}$. At the points in the preceding proofs where we used $\|\beta\| = 1$ and therefore $(\beta \cdot n)^2 \leq |\beta \cdot n|$, now it holds $(\beta \cdot n)^2 \leq \|\beta\| |\beta \cdot n|$. Additionally, we use Lemma 6.4.1 and Lemma 6.4.2, which hold for non-normalized advection coefficient. With these changes, all other steps stay the same. In contrast to before, the constant C_β now depends on $\|\beta\|$. □

For a diffusion reaction advection problem, the authors in [11] investigate error estimates for a non normalized advection coefficient with a more sophisticated augmentation of the test function. They use the standard upwind discretization for the advection part.

6.5. Numerical results

To verify the theory developed above by numerical examples, we define two quantities: the error e_h and the experimental order of convergence (EOC). We use both to investigate the convergence of the discretization method. We perform successive global mesh refinements and compare the computed solutions.

6.5.1 Definition. The error e_h between a global refinement step is defined by

$$e_h := \|y(h) - y(h/2)\| .$$

△

6.5.2 Definition. The *experimental order of convergence (EOC)* is defined by

$$EOC := \log\left(\frac{\|y(h) - y(h/2)\|}{\|y(h/2) - y(h/4)\|}\right)\frac{1}{\log(2)}.$$

△

The finer the mesh refinement, which means the smaller h, the smaller this error e_h should become. The experimental order of convergence should be 2 with linear finite elements, and 3 with quadratic finite elements with respect to the $\mathcal{L}^2$-norm.

We test these theoretical findings for our newly developed differentiable discretization in comparison with the standard upwind discretization. First we test it with an example with a smooth solution, after that we show results for a nonsmooth solution.

6.5.3 Example. *Smooth solution.* As a first example, we take example 1 from [11]. They consider a diffusion advection problem with small diffusion. We change it to a pure advection problem, that means $\alpha = 0$ and $\rho = 0$, see Example 2.1.7. We take $\beta = (0.71, 0.71)$, instead of $\beta = (1, 1)$ in [11], because $\beta = (1, 1)$ would yield the same value for both $\sigma_\mu(\beta, n)$ and $\sigma_{upw}(\beta, n)$ and thus a comparison is not possible. The right-hand-side f and the inflow boundary condition are chosen such that the analytical solution is $y = \sin(2\pi x_1)\sin(2\pi x_2)$, that means $y_D = \sin(2\pi x_1)\sin(2\pi x_2)$. As domain we choose the rectangle $\Omega = [-1, 1]^2$.

Tables 6.1 and 6.2 show the computed errors e_h and experimental orders of convergence

(EOC) for three different settings: the differentiable discretization with $\sigma_\mu(\beta, n)$ from (6.3) with $\mu = 0.01$, the extreme case $\mu = 1.0$, and the standard upwind discretization with $\sigma_{upw}(\beta, n)$ from (4.5), which is the same as $\sigma_\mu(\beta, n)$ with $\mu = 0$. The first rows of Tables 6.1 and 6.2 show the number of degrees of freedom (DoFs) which illustrate the refinement of the mesh. Table 6.1 shows results for linear finite elements, while Table 6.2 shows results for quadratic finite elements. All errors are computed in the $\mathcal{L}^2$-norm.

	$\sigma_\mu(\beta, n)$, $\mu = 1.0$		$\sigma_\mu(\beta, n)$, $\mu = 0.01$		$\sigma_{upw}(\beta, n)$	
#DoFs	e_h	EOC	e_h	EOC	e_h	EOC
64	$1.5046 \cdot 10^{-1}$	-	$1.4357 \cdot 10^{-1}$	-	$1.4356 \cdot 10^{-1}$	-
256	$4.1625 \cdot 10^{-2}$	1.8539	$4.1502 \cdot 10^{-2}$	1.7905	$4.1502 \cdot 10^{-2}$	1.7904
1,024	$8.8261 \cdot 10^{-3}$	2.2376	$9.7460 \cdot 10^{-3}$	2.0903	$9.7463 \cdot 10^{-3}$	2.0903
4,096	$1.8906 \cdot 10^{-3}$	2.2229	$2.1208 \cdot 10^{-3}$	2.2002	$2.1209 \cdot 10^{-3}$	2.2002
16,384	$4.4762 \cdot 10^{-4}$	2.0785	$5.0049 \cdot 10^{-4}$	2.0832	$5.0051 \cdot 10^{-4}$	2.0832
65,536	$1.1024 \cdot 10^{-4}$	2.0216	$1.2308 \cdot 10^{-4}$	2.0237	$1.2309 \cdot 10^{-4}$	2.0237

Table 6.1.: Linear finite elements. Errors e_h and EOC for differentiable discretization $\sigma_\mu(\beta, n)$ and standard discretization $\sigma_{upw}(\beta, n)$.

	$\sigma_\mu(\beta, n)$, $\mu = 1.0$		$\sigma_\mu(\beta, n)$, $\mu = 0.01$		$\sigma_{upw}(\beta, n)$	
#DoFs	e_h	EOC	e_h	EOC	e_h	EOC
576	$9.7877 \cdot 10^{-3}$	-	$9.8133 \cdot 10^{-3}$	-	$9.8133 \cdot 10^{-3}$	-
2,304	$4.4884 \cdot 10^{-3}$	1.1248	$4.4401 \cdot 10^{-3}$	1.1441	$4.4401 \cdot 10^{-3}$	1.1441
9,216	$5.8977 \cdot 10^{-4}$	2.9280	$5.2576 \cdot 10^{-4}$	3.0781	$5.2575 \cdot 10^{-4}$	3.0782
36,864	$7.6045 \cdot 10^{-5}$	2.9552	$6.5406 \cdot 10^{-5}$	3.0069	$6.5405 \cdot 10^{-5}$	3.0069
147,456	$9.6173 \cdot 10^{-6}$	2.9832	$8.1726 \cdot 10^{-6}$	3.0006	$8.1723 \cdot 10^{-6}$	3.0006
589,824	$1.2072 \cdot 10^{-6}$	2.9940	$1.0215 \cdot 10^{-6}$	3.0001	$1.0215 \cdot 10^{-6}$	3.0001

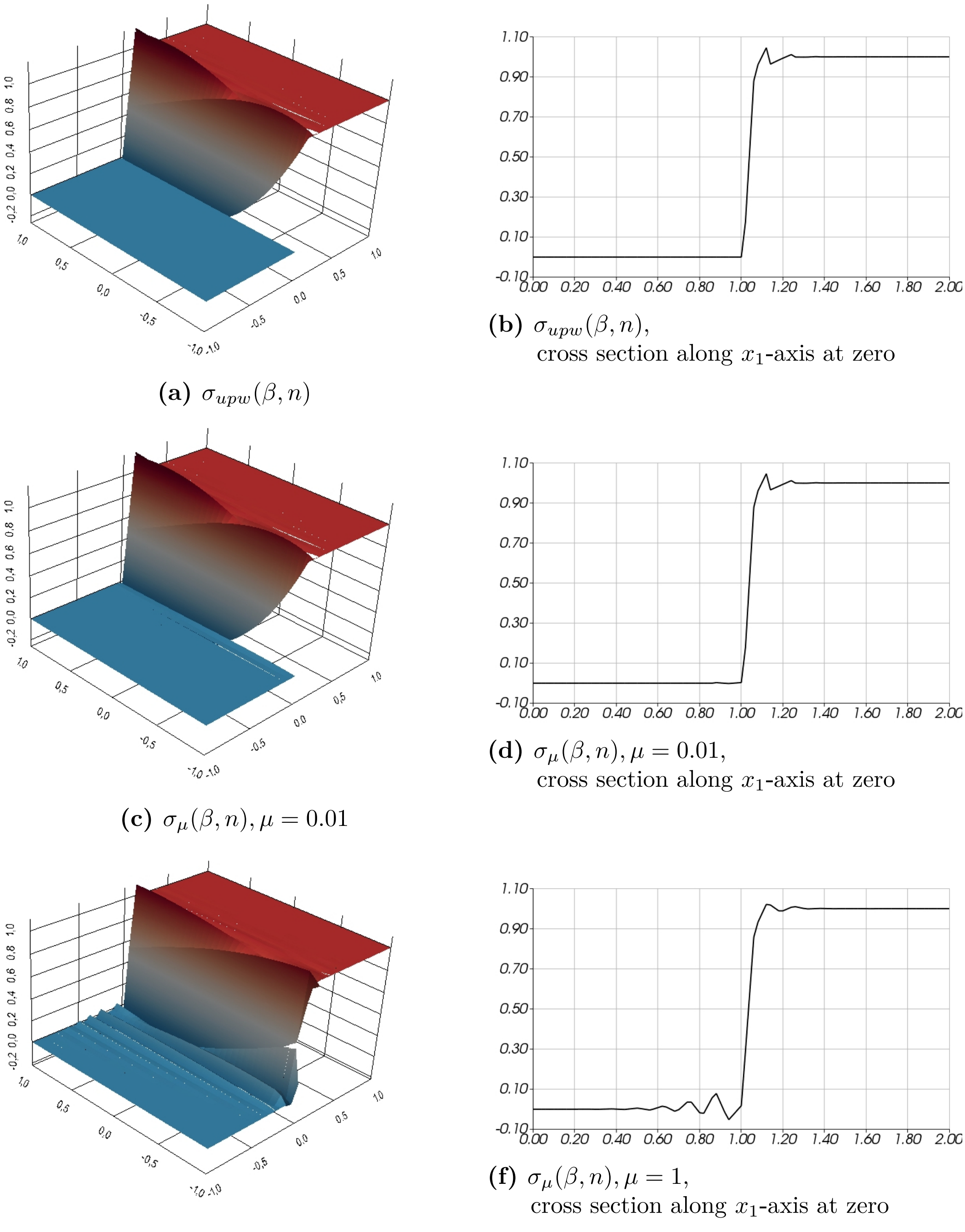
Table 6.2.: Quadratic finite elements. Errors e_h and EOC for differentiable discretization $\sigma_\mu(\beta, n)$ and standard discretization $\sigma_{upw}(\beta, n)$.

As expected from the theory developed above, the errors converge with order 2 for linear finite elements (Table 6.1) and order 3 for quadratic finite elements (Table 6.2). This is true for all examined discretizations. The developed differentiable discretization with $\sigma_\mu(\beta, n)$ shows a similar convergence behavior as the standard upwind discretization with $\sigma_{upw}(\beta, n)$. We notice that the choice of μ has a slight influence on the computed numbers. With $\mu = 1.0$ we get different errors, the order of convergence is still the same as for the upwind discretization. $\triangle$

6.5.4 Example. *Nonsmooth solution: Riemann problem.* Again we compute a pure advection problem, Example 2.1.7, to compare the two discretizations. The special feature of the Riemann problem are the boundary conditions, on the inflow boundary it is $y_D = 0$ for $x \leq 0$ and $y_D = 1$ for $x > 0$. The right hand side is $f = 0$. We choose advection direction $\beta = (0.03125, 1.0)$. This results in a discontinuity for the solution y, which is nearly along the x_2-axis but slightly rotated. The domain is the rectangle $\Omega = [-1, 1]^2$.

Figure 6.1 shows the computed results for a global refinement with mesh size $h = 0.0625$ that results in a rather coarse grid consisting of 2,304 DoFs. The left column depicts a three-dimensional view on the solution y_h, while the right column shows the corresponding cross section along the x_1-axis at zero, respectively. The first row shows the computed solution y_h for the standard upwind discretization $\sigma_{upw}(\beta, n)$ (4.5), the second row shows the computed solution y_h for the differentiable discretization $\sigma_\mu(\beta, n)$ (6.3) with $\mu = 0.01$ and the third row shows the computed solution y_h for $\mu = 1$.

The results of the first two rows are almost identical. The third row with $\mu = 1$ shows that the choice of the variable μ is important. If it is chosen too high, as in this example, the approximation of the solution is no longer as good as the approximation by the upwind discretization. The choice of μ has to be further investigated. A dependence on the mesh size h or the advection parameter β should be studied. $\triangle$



(a) $\sigma_{upw}(\beta, n)$

(b) $\sigma_{upw}(\beta, n)$, cross section along x_1-axis at zero

(c) $\sigma_\mu(\beta, n), \mu = 0.01$

(d) $\sigma_\mu(\beta, n), \mu = 0.01$, cross section along x_1-axis at zero

(e) $\sigma_\mu(\beta, n), \mu = 1$

(f) $\sigma_\mu(\beta, n), \mu = 1$, cross section along x_1-axis at zero

Figure 6.1.: Nonsmooth solution, Riemann problem. Computed solution y_h for upwind discretization with $\sigma_{upw}(\beta, n)$ and differentiable discretization with $\sigma_\mu(\beta, n)$ for $\mu = 0.01$ and $\mu = 1$.

Part IV.

Sensitivity generation: Transfer of the principle of IND to PDEs

7. Structure exploitation of primal and tangential discretization schemes

This chapter deals with novel sensitivity generation methods for parameter estimation and optimum experimental design with PDE models. We transfer the principle of IND to PDEs. Because of freezing the adaptive components and thus using the same discretization for primal and tangential problems, structure exploitation becomes possible. We exploit the common structure of primal and tangential discretizations and develop tailored methods for algorithmic differentiation. That leads to a significant saving of memory in comparison to differentiating the complete code as in black box AD. Thus we develop methods to efficiently and automatically generate sensitivities.

We proceed as follows: we investigate two possibilities for structure exploitation, first the problem structure, which means the structure of the primal and the tangential problems derived in Chapter 5. Second, we develop a method to exploit the structure of the finite element method. Finally, we demonstrate the efficiency of the developed methods on numerical examples.

7.1. Problem structure

As we have seen in Chapter 5, for algorithmic solutions of both optimization problems, the PE problem and the OED problem, the Jacobian of the PE problem is required. The Jacobian consists in particular of directional derivatives of the model response with respect to the parameters. To evaluate those directional derivatives, we select the sensitivity approach, it is depicted in Section 5.3. For that, we have to set up and solve tangential PDE problems. Following the principle of IND to generate consistent sensitivities, we solve these tangential PDE problems with the same discretization as the primal PDE problem. This creates possibilities to exploit the common structure of primal and tangential discretization schemes.

The primal PDE problem in short notation is

$$F(p; S(p), v) = 0, \quad \forall v \in V(\Omega), \tag{7.1}$$

with solution operator $S(p)$, see Section 2.2.

The corresponding tangential equation (5.4) revisited with $S_{p_j} = \frac{\partial S(p)}{\partial p_j}$ reads

$$\frac{dF(p;S(p),v)}{dp_j} = \frac{\partial F(p;S(p),v)}{\partial S} S_{p_j} + \frac{\partial F(p;S(p),v)}{\partial p_j} = 0, \quad \forall v \in V(\Omega). \tag{7.2}$$

Again, we omit the direction δp_j of the directional derivatives for clarity.

We solve the tangential PDE problem (7.2) and get as solution the derivative of the solution operator $S(p)$ with respect to the j-th parameter: S_{p_j}. We repeat this procedure for every $p_j, j = 1,..,n_p$. With these derivatives $S_{p_j}, j = 1,..,n_p$, we calculate the Jacobian, which we need in the course of the optimization algorithms for parameter estimation and optimum experimental design.

We compute the tangential solutions S_{p_j}, $j = 1,..,n_p$, in two steps:

1) set up tangential equations (7.2), $j = 1,..,n_p$,

2) solve primal and tangential PDE problems by FE method.

In the following we explain these two steps in more detail. For each step we examine the structure exploitation.

Tangential equations set up We begin with examining the structure exploitation for the tangential equations (7.2) set up for $j = 1,..,n_p$. The tangential equation (7.2) depends on the parameter p_j. We take the discretized version of the primal equation (4.9)

$$F_h(p;y_h,v_h) = 0, \quad \forall v_h \in V_h, \tag{7.3}$$

and use it to generate a discretized version (5.6) of the tangential equation

$$\frac{\partial F_h(p;S_h(p),v_h)}{\partial S_h} S_{h,p_j} = -\frac{\partial F_h(p;S_h(p),v_h)}{\partial p_j}, \quad \forall v_h \in V_h. \tag{7.4}$$

To generate the tangential equation, we need to calculate two partial derivatives: one with respect to the discrete solution operator S_h

$$\frac{\partial F_h(p;S_h(p),v_h)}{\partial S_h} S_{h,p_j} \tag{7.5}$$

and another one with respect to parameter p_j

$$\frac{\partial F_h(p; S_h(p), v_h)}{\partial p_j}. \tag{7.6}$$

For derivative (7.5) we exploit the problem structure, which means the linearity of the model problem. The derivative (7.6) is dependent on parameter p_j. For the generation of this derivative we cannot reuse parts of the primal problem. Instead we have to differentiate the primal discretization with respect to every parameter $p_j, j = 1, .., n_p$. How we do this efficiently by exploiting the structure of the finite element (FE) method is depicted in the next Section 7.2.

7.1.1 Example. *Diffusion advection reaction model problem.* We illustrate the structure exploitation due to the linearity of the model problem by the diffusion advection reaction model problem, Example 2.1.5. The discrete PDE model problem, the primal problem, in short notation reads

$$F_h(p; S_h(p), v_h) = \alpha a_h(S_h(p), v_h) + b_h(p; S_h(p), v_h) + \rho(p) c_h(p; S_h(p), v_h) - f_h(p; v_h).$$

We calculate the derivative of the discrete bilinear form $F_h(p; S_h(p), v_h)$ with respect to the second argument (7.5) for the primal problem

$$\begin{aligned}\frac{\partial F_h(p; S_h(p), v_h)}{\partial S_h} S_{h,p_j} =& \alpha \frac{\partial}{\partial S_h} a_h(S_h(p), v_h) S_{h,p_j} + \frac{\partial}{\partial S_h} b_h(p; S_h(p), v_h) S_{h,p_j} \\ &+ \rho(p) \frac{\partial}{\partial S_h} c_h(p; S_h(p), v_h) S_{h,p_j} - \frac{\partial}{\partial S_h} f_h(p; v_h) S_{h,p_j}, \\ =& \alpha a_h(S_{h,p_j}, v_h) + b_h(p; S_{h,p_j}, v_h) + \rho(p) c_h(p; S_{h,p_j}, v_h) \\ =& F_h(p; S_{h,p_j}, v_h) + f_h(p; v_h).\end{aligned}$$

We observe that the diffusion advection reaction structure remains unchanged. Except for the right hand side function $f_h(p; v_h)$, which is not dependent on the solution operator S_h and thus cancels out, we can reuse the primal bilinear form $F_h(p; ., v_h)$ to compute this derivative. △

As we have seen in the example, the derivative (7.5) is the same as the left hand side of the primal problem (7.3). Indeed, the only difference lies in the solution operator: in the primal problem $S_h(p)$ solves the equation, in the tangential problem S_{h,p_j} solves the equation. Thus if we add the right hand side of the primal problem $f_h(p; v_h)$ to

$F_h(p; ., v_h)$, the right hand side cancels out and we can write for the derivative (7.5)

$$\frac{\partial F_h(p; S_h(p), v_h)}{\partial S_h} S_{h,p_j} = F_h(p; S_{h,p_j}, v_h) + f_h(p; v_h).$$

In other words, the left hand side of the tangential problem (7.4) is identical with left hand side of the primal problem (7.3). We exploit this linearity of the model problem by reusing the already existing discretization of the left hand side of the primal problem (7.3) to generate the left hand side part (7.5) of the sensitivity equation.

Solve primal and tangential PDE problems by FE method After setting up the tangential problems, we solve primal and tangential PDE problems with the FE method. In the first step of the FE algorithm we assemble stiffness matrix and load vector, see Section 4.4. For the computation of the left hand side the stiffness matrix is assembled. We have seen in the last paragraph that the left hand sides of primal and tangential problems are identical due to our linear problem setting. Therefore the stiffness matrix is identical for primal and tangential equations. Thus it does not need to be assembled newly for every tangential PDE problem. We can built it once for the primal problem and reuse the stiffness matrix for the solution of the tangential problems.

7.1.2 Remark. *Matrix-free simulation.* This approach of storing requires memory space. Recent approaches in the simulation of PDE problems [71], [78] propose to not store this matrix and to not even built it up completely. Instead, recomputing single values is cheaper than storing the whole matrix. The developed methods for structure exploitation of primal and tangential PDE problems can be directly transferred to matrix-free simulation methods. △

7.2. Structure of finite element method

In this section we exploit the structure of the FE method and therefore apply algorithmic differentiation to core parts of the discretized primal PDE problem. We begin with describing our general approach. After that we go into more detail and investigate the structures induced by the FE discretization. Finally we illustrate the structure exploitation on the algorithm level.

General approach The derivative of the discrete bilinear form $F_h(p; S_h(p), v_h)$ with respect to the parameter p_j (7.6) has to be set up newly for every parameter $p_j, j = 1, .., n_p$:

$$\frac{\partial F_h(p; S_h(p), v_h)}{\partial p_j}. \tag{7.7}$$

It corresponds to the right hand side of the tangential equation (7.4), because it is independent of the tangential solution operator S_{h,p_j}. We set up and compute the derivative (7.7) by exploiting the structure of the FE method. We apply algorithmic differentiation to the discretized primal PDE problem (7.3). That means we process the discretized primal PDE (7.3) by an automatic differentiation tool to generate the discretization of the tangential problem (7.4). We only process that part of programming code, which implements core parts of the discretization of the PDE problem.

Structure exploitation FE discretization The discontinuous Galerkin FE discretization comprises of three major sums: one sum over the cells $T \in \mathbb{T}_h$, another sum over the interior faces $F \in \mathbb{F}_h^{int}$ and a third one over the boundary faces $F \in \mathbb{F}_h^{\Gamma}$. The summands of these sums consist in the integrals over each cell, interior face and boundary face. We define a short notation for the terms belonging to these integrals: $g_T(p; y_h, v_h)$ for the cell integral, $g_{int}(p; y_h, v_h)$ for the interior face integral and $g_\Gamma(p; y_h, v_h)$ for the boundary face integral. The discretization of the primal problem (7.3) in this short notation reads

$$\begin{aligned} F_h(p; y_h, v_h) = & \sum_{T \in \mathbb{T}_h} \int_T g_T(p; y_h, v_h) dx + \sum_{F \in \mathbb{F}_h^{int}} \int_F g_{int}(p; y_h, v_h) ds \\ & + \sum_{F \in \mathbb{F}_h^{\Gamma}} \int_F g_\Gamma(p; y_h, v_h) ds. \end{aligned} \tag{7.8}$$

Our goal is to differentiate this discretization (7.8) with respect to the parameter p_j

$$\begin{aligned} \frac{\partial}{\partial p_j} F_h(p; y_h, v_h) = & \sum_{T \in \mathbb{T}_h} \int_T \frac{\partial}{\partial p_j} g_T(p; y_h, v_h) dx + \sum_{F \in \mathbb{F}_h^{int}} \int_F \frac{\partial}{\partial p_j} g_{int}(p; y_h, v_h) ds \\ & + \sum_{F \in \mathbb{F}_h^{\Gamma}} \int_F \frac{\partial}{\partial p_j} g_\Gamma(p; y_h, v_h) ds. \end{aligned} \tag{7.9a}$$

Notice that a generation of the tangential discretizations from the primal discretization is only possible, if we are operating on the identical sets $\mathbb{T}_h, \mathbb{F}_h^{int}$ and $\mathbb{F}_h^{\Gamma}$. That means the triangulation of primal and tangential problems has to be identical. We use the same triangulation with frozen adaptive components for primal and tangential discretizations, due to the transfer of the principle of IND, see Section 5.2.

Thus the structure of the FE method consists in summing over cells, interior faces and boundary faces and integrating over each single cell, interior face and boundary face. We exploit this structure by differentiating the single integrands by automatic differentiation. An automatic differentiation tool processes these parts of code and generates via code transformation the derivatives of the integrands with respect to parameter p_j. Details about this procedure are depicted after the subsequent example in the next paragraph.

7.2.1 Example. *Structure exploitation for the diffusion advection reaction model problem.* We calculate the derivative of the discrete bilinear form with respect to parameter p_j (7.7) by the presented procedure. We use the discrete primal problem of our diffusion advection reaction model problem Example 2.1.5 to set up the discrete tangential problem. Thereby we exploit the structure of the FE method by only deriving the single integrands in the discretization.

The discrete problem for the diffusion advection reaction model problem reads, see Section 4.2 and Section 6.3: Find $y_h \in V_h$ such that

$$F_h(p; y_h, v_h) = \alpha a_h(y_h, v_h) + b_h(p; y_h, v_h) + \rho(p) c_h(p; y_h, v_h) - f_h(p; v_h), \tag{7.10a}$$

the diffusion part reads

$$\begin{aligned} a_h(y_h, v_h) = &\sum_{T \in \mathbb{T}_h} \int_T \nabla y_h \nabla v_h dx \\ &+ \sum_{F \in \mathbb{F}_h^{int}} \int_F \Big[-2\{\!\{\nabla y_h\}\!\}\{\!\{v_h n\}\!\} - 2\{\!\{y_h n\}\!\}\{\!\{\nabla v_h\}\!\} + \gamma [\![y_h]\!] [\![v_h]\!] \Big] ds \\ &+ \sum_{F \in \mathbb{F}_h^{\Gamma}} \int_F \Big[-v_h \partial_n y_h - y_h \partial_n v_h + 2\gamma y_h v_h \Big] ds, \end{aligned} \tag{7.10b}$$

the advection part reads

$$b_h(p; y_h, v_h) = -\sum_{T \in \mathbb{T}_h} \int_T y_h \beta(p) \cdot \nabla v_h dx + \sum_{F \in \mathbb{F}_h^{int}} \int_F \{\!\{y_h\}\!\} 2 \{\!\{v_h\, \beta(p) \cdot n\}\!\} ds \tag{7.10c}$$

$$+ \sum_{F\in\mathbb{F}_h^{int}} \int_F \frac{1}{2}\sigma_\mu(\beta(p),n) \llbracket y_h \rrbracket \llbracket v_h \rrbracket \, ds + \sum_{F\in\mathbb{F}_h^{\mathit{\Gamma}}} \int_F \Big[\Big(\frac{1}{2}\beta(p)\cdot n y_h + \frac{1}{2}\sigma_\mu(\beta(p),n) y_h\Big) v_h\Big] ds,$$

the reaction part reads

$$c_h(p; y_h, v_h) = \sum_{T\in\mathbb{T}_h} \int_T y_h v_h dx, \tag{7.10d}$$

and the right hand side is

$$\begin{aligned} f_h(p; v_h) = \sum_{T\in\mathbb{T}_h} \int_T f(p) v_h dx + \sum_{F\in\mathbb{F}_h^{\mathit{\Gamma}}} \int_F \Big[2\gamma y_D(p) v_h - \alpha y_D(p)\partial_n v_h\Big] ds \\ - \sum_{F\in\mathbb{F}_h^{\mathit{\Gamma}}} \int_F \Big[\Big(\frac{1}{2}\beta(p)\cdot n y_D(p) - \frac{1}{2}\sigma_\mu(\beta(p),n) y_D(p)\Big) v_h\Big] ds. \end{aligned} \tag{7.10e}$$

We regroup the terms in a cell integral, a interior face integral and a boundary face integral. Therefore we define a function for all cell terms

$$g_T(p; y_h, v_h) := \alpha\nabla y_h \nabla v_h - y_h\beta(p)\cdot\nabla v_h + \rho(p) y_h v_h - f(p) v_h,$$

a function for all terms corresponding to interior faces

$$\begin{aligned} g_{int}(p; y_h, v_h) :=& \alpha\Big(- 2\{\!\{\nabla y_h\}\!\}\{\!\{v_h n\}\!\} - 2\{\!\{y_h n\}\!\}\{\!\{\nabla v_h\}\!\} + \gamma \llbracket y_h \rrbracket \llbracket v_h \rrbracket \Big) \\ &+ \{\!\{y_h\}\!\} 2\{\!\{v_h \, \beta(p)\cdot n\}\!\} + \frac{1}{2}\sigma_\mu(\beta(p),n) \llbracket y_h \rrbracket \llbracket v_h \rrbracket \end{aligned}$$

and a function for all boundary face terms

$$\begin{aligned} g_{\mathit{\Gamma}}(p; y_h, v_h) := \alpha\Big(- v_h\partial_n y_h - y_h\partial_n v_h + 2\gamma y_h v_h\Big) + \Big(\frac{1}{2}\beta(p)\cdot n y_h + \frac{1}{2}\sigma_\mu(\beta(p),n) y_h\Big) v_h \\ - \Big(2\gamma y_D(p) v_h - \alpha y_D(p)\partial_n v_h\Big) + \Big(\frac{1}{2}\beta(p)\cdot n y_D(p) - \frac{1}{2}\sigma_\mu(\beta(p),n) y_D(p)\Big) v_h. \end{aligned}$$

With these definitions, we rewrite the discretization of the primal problem (7.10a) of the diffusion advection reaction model problem as

$$\begin{aligned} &F_h(p; y_h, v_h) = \\ &\sum_{T\in\mathbb{T}_h} \int_T g_T(p; y_h, v_h) dx + \sum_{F\in\mathbb{F}_h^{int}} \int_F g_{int}(p; y_h, v_h) ds + \sum_{F\in\mathbb{F}_h^{\mathit{\Gamma}}} \int_F g_{\mathit{\Gamma}}(p; y_h, v_h) ds = 0. \end{aligned}$$

In the next step we derive the single integrands according to equation (7.9). We begin with $g_T(p; y_h, v_h)$:

$$\frac{\partial}{\partial p_j} g_T(p; y_h, v_h) = \frac{\partial}{\partial p_j}\left(\alpha \nabla y_h \nabla v_h\right) - \frac{\partial}{\partial p_j}\left(y_h \beta(p) \cdot \nabla v_h\right) \tag{7.11a}$$

$$+ \frac{\partial}{\partial p_j}\left(\rho(p) y_h v_h\right) - \frac{\partial}{\partial p_j}\left(f(p) v_h\right). \tag{7.11b}$$

Furthermore we derive $g_{int}(p; y_h, v_h)$

$$\frac{\partial}{\partial p_j} g_{int}(p; y_h, v_h) = \tag{7.12a}$$

$$\frac{\partial}{\partial p_j}\alpha\Big(- 2\{\!\{\nabla y_h\}\!\}\{\!\{v_h n\}\!\} - 2\{\!\{y_h n\}\!\}\{\!\{\nabla v_h\}\!\} + \gamma \, [\![y_h]\!] \, [\![v_h]\!] \Big) \tag{7.12b}$$

$$+ \frac{\partial}{\partial p_j}\Big(\{\!\{y_h\}\!\} 2 \{\!\{v_h \, \beta(p) \cdot n\}\!\}\Big) + \frac{\partial}{\partial p_j}\left(\frac{1}{2}\sigma_\mu(\beta(p), n) \, [\![y_h]\!] \, [\![v_h]\!]\right) \tag{7.12c}$$

and $g_\Gamma(p; y_h, v_h)$

$$\frac{\partial}{\partial p_j} g_\Gamma(p; y_h, v_h) = \frac{\partial}{\partial p_j}\alpha\Big(- \partial_n y_h v_h - y_h \partial_n v_h + 2\gamma y_h v_h\Big) \tag{7.13a}$$

$$+ \frac{\partial}{\partial p_j}\Big(\frac{1}{2}\beta(p) \cdot n y_h + \frac{1}{2}\sigma_\mu(\beta(p), n) y_h\Big) v_h \tag{7.13b}$$

$$- \frac{\partial}{\partial p_j}\Big(2\gamma y_D(p) v_h - \alpha y_D(p) \partial_n v_h\Big) \tag{7.13c}$$

$$+ \frac{\partial}{\partial p_j}\Big(\frac{1}{2}\beta(p) \cdot n y_D(p) - \frac{1}{2}\sigma_\mu(\beta(p), n) y_D(p)\Big) v_h. \tag{7.13d}$$

Thus by summing up the derivatives of the single integrands, we arrive at the derivative of the primal discretized bilinear form with respect to parameter p_j. Note that some terms are independent of p_j and thus cancel out. We get

$$\frac{\partial F_h(p; y_h, v_h)}{\partial p_j} \tag{7.14a}$$

$$= \sum_{T \in \mathbb{T}_h} \int_T \frac{\partial}{\partial p_j} g_T(p; y_h, v_h) dx + \sum_{F \in \mathbb{F}_h^{int}} \int_F \frac{\partial}{\partial p_j} g_{int}(p; y_h, v_h) ds \tag{7.14b}$$

$$+ \sum_{F \in \mathbb{F}_h^\Gamma} \int_F \frac{\partial}{\partial p_j} g_\Gamma(p; y_h, v_h) ds \tag{7.14c}$$

$$= \sum_{T \in \mathbb{T}_h} \int_T \Big[- \frac{\partial}{\partial p_j} \left(y_h \beta(p) \cdot \nabla v_h \right) + \frac{\partial}{\partial p_j} \left(\rho(p) y_h v_h \right) - \frac{\partial}{\partial p_j} \left(f(p) v_h \right) \Big] dx \tag{7.14d}$$

$$+ \sum_{F \in \mathbb{F}_h^{int}} \int_F \Big[\frac{\partial}{\partial p_j} \Big(\{\!\{ y_h \}\!\} 2 \{\!\{ v_h \, \beta(p) \cdot n \}\!\} \Big) + \frac{\partial}{\partial p_j} \Big(\frac{1}{2} \sigma_\mu(\beta(p), n) \, [\![y_h]\!] \, [\![v_h]\!] \Big) \Big] ds \tag{7.14e}$$

$$+ \sum_{F \in \mathbb{F}_h^{\Gamma}} \int_F \Big[\frac{\partial}{\partial p_j} \Big(\frac{1}{2} \beta(p) \cdot n y_h + \frac{1}{2} \sigma_\mu(\beta(p), n) y_h \Big) v_h \tag{7.14f}$$

$$- \frac{\partial}{\partial p_j} \Big(2 \gamma y_D(p) v_h - \alpha y_D(p) \partial_n v_h \Big) \tag{7.14g}$$

$$+ \frac{\partial}{\partial p_j} \Big(\frac{1}{2} \beta(p) \cdot n y_D(p) - \frac{1}{2} \sigma_\mu(\beta(p), n) y_D(p) \Big) v_h \Big] ds. \tag{7.14h}$$

As mentioned above, the derivation of the single integrands, performed in equations (7.11), (7.12) and (7.13), is done by an automatic differentiation tool in our algorithm.

The calculated derivative of the discrete bilinear form with respect to parameter p_j (7.14) represents the right hand side of the tangential problem belonging to p_j. Thus with equation (7.14) and the problem structure exploitation presented in the previous Section 7.1 we set up the full discrete tangential problem for p_j for the diffusion advection reaction model problem:

$$\begin{aligned}
&F_h(p; y_h, v_h) - f(p; v_h) \\
&\quad = \sum_{T \in \mathbb{T}_h} \int_T \frac{\partial}{\partial p_j} g_T(p; y_h, v_h) dx + \sum_{F \in \mathbb{F}_h^{int}} \int_F \frac{\partial}{\partial p_j} g_{int}(p; y_h, v_h) ds \\
&\qquad + \sum_{F \in \mathbb{F}_h^{\Gamma}} \int_F \frac{\partial}{\partial p_j} g_\Gamma(p; y_h, v_h) ds, \quad \forall v_h \in V_h \\
&\Leftrightarrow \\
&\alpha a_h(S_{h,p_j}, v_h) + b_h(p; S_{h,p_j}, v_h) + \rho(p) c_h(p; S_{h,p_j}, v_h) \\
&\quad = - \sum_{T \in \mathbb{T}_h} \int_T \Big[- \frac{\partial}{\partial p_j} \left(y_h \beta(p) \cdot \nabla v_h \right) + \frac{\partial}{\partial p_j} \rho(p) \left(y_h v_h \right) - \frac{\partial}{\partial p_j} \left(f(p) v_h \right) \Big] dx \\
&\qquad - \sum_{F \in \mathbb{F}_h^{int}} \int_F \Big[\frac{\partial}{\partial p_j} \Big(\{\!\{ y_h \}\!\} 2 \{\!\{ v_h \, \beta(p) \cdot n \}\!\} \Big) + \frac{\partial}{\partial p_j} \Big(\frac{1}{2} \sigma_\mu(\beta(p), n) \, [\![y_h]\!] \, [\![v_h]\!] \Big) \Big] ds \\
&\qquad - \sum_{F \in \mathbb{F}_h^{\Gamma}} \int_F \Big[\frac{\partial}{\partial p_j} \Big(\frac{1}{2} \beta(p) \cdot n y_h + \frac{1}{2} \sigma_\mu(\beta(p), n) y_h \Big) v_h \qquad\qquad (7.15) \\
&\qquad\qquad - \frac{\partial}{\partial p_j} \Big(2 \gamma y_D(p) v_h - \alpha y_D(p) \partial_n v_h \Big)
\end{aligned}$$

$$+ \frac{\partial}{\partial p_j}\Big(\frac{1}{2}\beta(p)\cdot n y_D(p) - \frac{1}{2}\sigma_\mu(\beta(p),n)y_D(p)\Big)v_h\Big]ds, \quad \forall v_h \in V_h.$$

$\triangle$

Structure exploitation FE algorithm Let us examine this structure exploitation on algorithm level. Before solving the system of equations by an iterative solver, we assemble the stiffness matrix to compute the left hand side and the load vector to compute the right hand side of the PDE problem. This assembling is done by computing the local matrices over cells, interior faces and boundary faces and after that summing the local contributions to the global stiffness matrix. Therefore the FE algorithm loops over all cells, interior faces and boundary faces of the grid elements. We do not have to differentiate all these loops, but rather the inner parts, where the mathematical formulation is encoded. For each cell, respective interior face or boundary face, there holds the same analytical formula. That is $g_T(p;\varphi_i,\varphi_j)$ on cells, $g_{int}(p;\varphi_i,\varphi_j)$ on interior faces and $g_\Gamma(p;\varphi_i,\varphi_j)$ on boundary faces.

We split the cell term into a matrix contribution and a right hand side contribution, the matrix contribution depends on both basis functions φ_i and φ_j, the right hand side contribution only depends on φ_j,

$$g_T(p;\varphi_i,\varphi_j) =: g_{T,m}(p;\varphi_i,\varphi_j) + g_{T,r}(p;\varphi_j).$$

In the same manner, we split the interior face term $g_{int}(p;\varphi_i,\varphi_j) =: g_{int,m}(p;\varphi_i,\varphi_j) + g_{int,r}(p;\varphi_j)$ and the boundary face term $g_\Gamma(p;\varphi_i,\varphi_j) =: g_{\Gamma,m}(p;\varphi_i,\varphi_j) + g_{\Gamma,r}(p;\varphi_j)$.

Algorithm 2 shows three loops for assembling of local stiffness matrices A_T for cells, $A_{F^{int}}$ for interior faces and A_{F^Γ} for boundary faces. We loop over all cells $T \in \mathbb{T}_h$, interior faces $F \in \mathbb{F}_h^{int}$ and boundary faces $F \in \mathbb{F}_h^\Gamma$. Each of these loops contains three additional loops: a loop over quadrature points $x_q, q = 1, ..., n_q$, where n_q is the number of quadrature points per element, and two loops over degrees of freedom (DoFs) of a cell, n_{dc} is the number of DoFs per cell. We evaluate the test functions φ_i and φ_j at quadrature point x_q and multiply the evaluated integrand with quadrature weight w_q.

Lines 7, 18 and 29 contain the inner part, where the mathematical formulation is encoded. Only these lines are parameter dependent. Thus we only differentiate these lines. The automatic differentiation tool takes the computational formulation of $g_{T,m}(p;\varphi_i,\varphi_j)$ and generates a code that contains the computational formulation of the derivative $\frac{\partial}{\partial p_j}g_{T,m}(p;\varphi_i,\varphi_j)$. We use this generated derivative in the assembling

Algorithm 2 Primal problem: assembling of local stiffness matrices. Loops over cells, interior faces and boundary faces.

```
Initialize matrices $A_T$, $A_{F^{int}}$, $A_{F^{\Gamma}} \in \mathbb{R}^{n_{dc} x n_{dc}}$ to zero.
                                                                  ▷ cells
for $T \in \mathbb{T}_h$ do
    for $q = 1, ..., n_q$ do
        for $i = 1, ..., n_{dc}$ do
            for $j = 1, ..., n_{dc}$ do
                compute $g_{T,m}(p; \varphi_i(x_q), \varphi_j(x_q))$
                $A_T(i,j) = A_T(i,j) + g_{T,m}(p; \varphi_i(x_q), \varphi_j(x_q)) w_q$
            end for
        end for
    end for
end for
                                                                  ▷ interior faces
for $F \in \mathbb{F}_h^{int}$ do
    for $q = 1, ..., n_q$ do
        for $i = 1, ..., n_{dc}$ do
            for $j = 1, ..., n_{dc}$ do
                compute $g_{int,m}(p; \varphi_i(x_q), \varphi_j(x_q))$
                $A_{F^{int}}(i,j) = A_{F^{int}}(i,j) + g_{int,m}(p; \varphi_i(x_q), \varphi_j(x_q)) w_q$
            end for
        end for
    end for
end for
                                                                  ▷ boundary faces
for $F \in \mathbb{F}_h^{\Gamma}$ do
    for $q = 1, ..., n_q$ do
        for $i = 1, ..., n_{dc}$ do
            for $j = 1, ..., n_{dc}$ do
                compute $g_{\Gamma,m}(p; \varphi_i(x_q), \varphi_j(x_q))$
                $A_{F^{\Gamma}}(i,j) = A_{F^{\Gamma}}(i,j) + g_{\Gamma,m}(p; \varphi_i(x_q), \varphi_j(x_q)) w_q$
            end for
        end for
    end for
end for
```

of the right hand side load vector of the j-th tangential problem. We proceed just the same for the two remaining derivatives $\frac{\partial}{\partial p_j} g_{int,m}(p; \varphi_i, \varphi_j)$ and $\frac{\partial}{\partial p_j} g_{\Gamma,m}(p; \varphi_i, \varphi_j)$.

In the same manner we generate the remaining derivatives from the assembling of the right hand side local load vectors of the primal problem. Algorithm 3 shows the corresponding loops for a local vector $\tilde{f}_T$ for the terms over cell sums, $\tilde{f}_{F^{int}}$ for the terms over interior face sums and $\tilde{f}_{F^\Gamma}$ for the terms over boundary face sums for the right hand side vector. As before in the matrix assembling loops, lines 6, 15 and 24 contain the inner part, which is parameter dependent. We take the computational formulation of $g_{T,r}(p; \varphi_j)$, $g_{int,r}(p; \varphi_j)$ and $g_{\Gamma,r}(p; \varphi_j)$ and process these lines by an automatic differentiation tool. We get code for the derivatives $\frac{\partial}{\partial p_j} g_{T,r}(p; \varphi_j)$, $\frac{\partial}{\partial p_j} g_{int,r}(p; \varphi_j)$ and $\frac{\partial}{\partial p_j} g_{\Gamma,r}(p; \varphi_j)$. We use the generated code in the assembling of the tangential right hand side load vector of the j-th tangential problem.

Algorithm 4 depicts the assembling of the right hand side load vector for tangential problem of parameter p_j. Again, we see the loops over cells, interior faces and boundary faces. In lines 5 and 6 the generated derivatives of the cell terms from local stiffness matrix assembling and local right hand side load vector assembling of the primal problem are evaluated. Similarly in lines 16, 17, 26 and 27 the derivatives of the interior face terms and the boundary face terms are evaluated, respectively. Note that depending on the problem setting the solution of the primal problem y_h can enter the tangential problem through derivatives of stiffness matrix contributions of the primal problem.

Thus, the structure of the FE method is exploited by differentiating the single integrands of the cell sums, interior face sums and boundary face sums. In this way, we efficiently and automatically generate the derivatives to set up the tangential problems.

Algorithm 3 Primal problem: assembling of local load vectors. Loops over cells, interior faces and boundary faces.

```
Initialize vectors $\tilde{f}_T$, $\tilde{f}_{F^{int}}$, $\tilde{f}_{F^{\Gamma}} \in \mathbb{R}^{n_{dc}}$ to zero.
                                                            ▷ cells
for $T \in \mathbb{T}_h$ do
    for $q = 1, ..., n_q$ do
        for $j = 1, ..., n_{dc}$ do
            compute $g_{T,r}(p; \varphi_j(x_q))$
            $\tilde{f}_T(j) = \tilde{f}_T(j) + g_{T,r}(p; \varphi_j(x_q))w_q$
        end for
    end for
end for
                                                            ▷ interior faces
for $F \in \mathbb{F}_h^{int}$ do
    for $q = 1, ..., n_q$ do
        for $j = 1, ..., n_{dc}$ do
            compute $g_{int,r}(p; \varphi_j(x_q))$
            $\tilde{f}_{F^{int}}(j) = \tilde{f}_{F^{int}}(j) + g_{int,r}(p; \varphi_j(x_q))w_q$
        end for
    end for
end for
                                                            ▷ boundary faces
for $F \in \mathbb{F}_h^{\Gamma}$ do
    for $q = 1, ..., n_q$ do
        for $j = 1, ..., n_{dc}$ do
            compute $g_{\Gamma,r}(p; \varphi_j(x_q))$
            $\tilde{f}_{F^{\Gamma}}(j) = \tilde{f}_{F^{\Gamma}}(j) + g_{\Gamma,r}(p; \varphi_j(x_q))w_q$
        end for
    end for
end for
```

Algorithm 4 Tangential problem: assembling of local load vectors. Loops over cells, interior faces and boundary faces.

```
Initialize vectors $\tilde{f}_T$, $\tilde{f}_{F^{int}}$, $\tilde{f}_{F^\Gamma} \in \mathbb{R}^{n_{dc}}$ to zero.
                                                                    ▷ cells
for $T \in \mathbb{T}_h$ do
    for $q = 1, ..., n_q$ do
        for $j = 1, ..., n_{dc}$ do
            compute $\frac{\partial}{\partial p_j} g_{T,m}(p; y_h(x_q), \varphi_j(x_q))$
            compute $\frac{\partial}{\partial p_j} g_{T,r}(p; \varphi_j(x_q))$
            $\tilde{f}_T(j) = \tilde{f}_T(j) - \frac{\partial}{\partial p_j} g_{T,m} w_q + \frac{\partial}{\partial p_j} g_{T,r} w_q$
        end for
    end for
end for
                                                                    ▷ interior faces
for $F \in \mathbb{F}_h^{int}$ do
    for $q = 1, ..., n_q$ do
        for $j = 1, ..., n_{dc}$ do
            compute $\frac{\partial}{\partial p_j} g_{int,m}(p; y_h(x_q), \varphi_j(x_q))$
            compute $\frac{\partial}{\partial p_j} g_{int,r}(p; \varphi_j(x_q))$
            $\tilde{f}_{F^{int}}(j) = \tilde{f}_{F^{int}}(j) - \frac{\partial}{\partial p_j} g_{int,m} w_q + \frac{\partial}{\partial p_j} g_{int.r} w_q$
        end for
    end for
end for
                                                                    ▷ boundary faces
for $F \in \mathbb{F}_h^\Gamma$ do
    for $q = 1, ..., n_q$ do
        for $j = 1, ..., n_{dc}$ do
            compute $\frac{\partial}{\partial p_j} g_{\Gamma,m}(p; y_h(x_q), \varphi_j(x_q))$
            compute $\frac{\partial}{\partial p_j} g_{\Gamma,r}(p; \varphi_j(x_q))$
            $\tilde{f}_{F^\Gamma}(j) = \tilde{f}_{F^\Gamma}(j) - \frac{\partial}{\partial p_j} g_{\Gamma,m} w_q + \frac{\partial}{\partial p_j} g_{\Gamma,r} w_q$
        end for
    end for
end for
```

7.3. Numerical results

7.3.1 Example. *Structure exploiting sensitivity evaluation for advection dominated diffusion advection PDE problem.* We illustrate the structure exploiting sensitivity evaluation with the unconstrained parameter estimation optimization Problem 2.2.2, respectively the discrete version Problem 4.3.2. The underlying PDE model problem possesses a small diffusion coefficient $\alpha = 0.001$, a constant right hand side $f = 1$ and a constant Dirichlet boundary condition $y_D = 1$. The domain Ω consists of the rectangle $[-1, 1]^2$. The strong form reads

$$\begin{aligned} -0.001\mathit{\Delta} y + \beta(p) \cdot \nabla y &= 1 \quad \text{on } \Omega, \\ y &= 1 \quad \text{on } \mathit{\Gamma} = \partial\Omega. \end{aligned}$$

The unknown parameter vector $p \in \mathbb{R}^2$ consists of the two components of the advection direction $\beta(p) = (p_1, p_2)^T$. Because of the small diffusion coefficient, the PDE model problem is advection dominated as long as the parameters do not get as small as the diffusion coefficient.

The corresponding discrete PDE problem reads: Find $y_h \in V_h$ such that

$$\alpha a_h(y_h, v_h) + b_h(p; y_h, v_h) = f_h(v_h), \quad \forall v_h \in V_h,$$

with $a_h(y_h, v_h)$, $b_h(p; y_h, v_h)$ and $f_h(v_h)$ defined as above, equation (7.10). We use the differentiable discretization $\sigma_\mu(\beta, n)$ (6.3) with $\mu = 0.1$. In this example, only the advection part $b_h(p; y_h, v_h)$ depends on the parameters.

The model response consists of point measurements given by the value of the discrete PDE solution operator at the measurement points $h_{i,h}(p) := S_h(p)\Big|_{x=x_i^m}$. Thus the discrete parameter estimation problem reads

$$\min_{p \in P} \frac{1}{2} \sum_{i=1}^{M} \left(\frac{\eta_i - h_{i,h}(p)}{\sigma_i} \right)^2 .$$

We generate the two tangential problems, one for the first parameter and one for the second parameter, with the structure exploiting technique depicted in the previous sections. Thus, we derive small parts of the primal discretization by an AD tool and generate the discrete tangential problems. With this generated code, we compute the tangential solutions as depicted above in Algorithm 4. We perform all computations on an uniformly refined grid with 147,456 DoFs. We use quadratic discontinuous finite elements for all computations. We compute on a desktop computer with a Pentium® Dual-Core CPU E5400 with 2.70GHz × 2 processor on Ubuntu 14.04 LTS with a memory of 12GB RAM.

Figure 7.1 shows the simulation of the primal problem (7.10) for the parameter values $p_1 = -0.2$ and $p_2 = 0.3$. Figure 7.2 shows on the left the simulation of the tangential problem (7.15) for the first parameter p_1 and on the right the solution of the tangential problem for the second parameter p_2. We see the influence of the advection direction on the PDE solution in all three figures. The advection direction points from the lower right corner to the left half of the upper boundary. If we compare the two tangential problems, we see that the solutions of them are mirrored at the advection direction: while the solution of the tangential problem for p_1 is zero on the left side and has very high values on the right side, the solution of the tangential problem for p_2 has very low values on the left side and is zero on the right side.

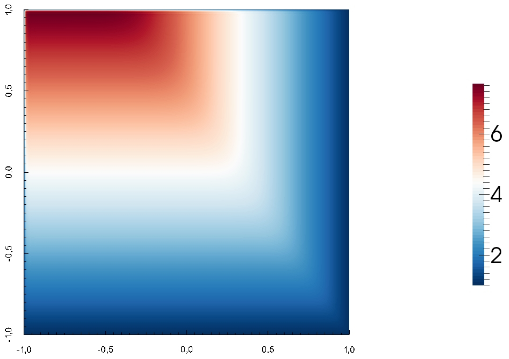


Figure 7.1.: Simulation of the primal PDE problem.

With this structure exploiting approach, the tangential problems are generated efficiently. No memory issues appear while generating the tangential problems. In contrast to using black box AD, we exploit the problem dependent structure and the structure of the FE method and therefore only differentiate small parts of the code with AD. This leads to a considerable saving of memory space, no memory issues occur. In comparison to finite differences, we obtain a much higher accuracy for the sensitivities. Furthermore, the user does not have to provide sensitivities. All sensitivities are generated by the program. △

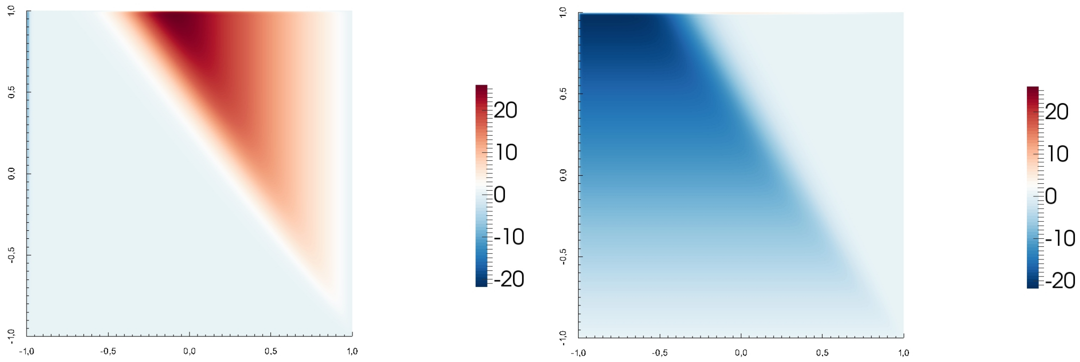


Figure 7.2.: Simulation of tangential PDE problems for p_1 (left) and p_2 (right).

8. Freezing of adaptive components

The main part of the principle of IND is frozen adaptivity. By freezing the adaptive components, for example grid, order, step size, and using the same discretization scheme for the primal and the tangential problems, the consistent derivatives of the model response are computed. Possible adaptive components are frozen, so that no problems arise with discontinuities from the adaptivity.

In our setting, potential adaptive components are the adaptive grid refinement with an error indicator of the spatial finite element grid and the adaptive step number of an iterative solver of the linear system. Both have to be frozen to obtain consistent sensitivities. How to treat these adaptive components in a suitable way is discussed in this chapter. We start with the adaptive FE grid and develop a heuristic, the *error sum strategy* for grid refinement. It generates one adaptively refined grid for primal and tangential PDE problems per Gauss-Newton iteration. After that we consider the iterative solver of the linear system. We analyze two possible options to solve the linear system of all PDE models. Finally, we present numerical results obtained with the developed methods.

8.1. Adaptive finite element grid

We begin this section with a general overview of grid refinement for a finite element grid. After that we consider the case of optimization: here we need multiple simulations of different PDE problems. Does that mean we need multiple grids? We give a short literature overview. Finally we propose the *error sum strategy* for grid refinement, which generates one adaptively refined grid for all PDE problems, that means for primal and tangential PDE problems, per Gauss-Newton iteration. The error sum strategy is in accordance with the principle of IND.

Grid refinement We perform the simulation of a PDE model problem on a grid, which is composed of closed quadrilateral grid cells. One possibility to generate such a grid is to start with a coarse grid and uniformly refine each grid cell to arrive at a finer grid. On the one hand, the finer the grid, the more accurate is the simulation. This is expressed by a small discretization error. On the other hand, the finer the grid, the more computationally expensive is the simulation.

To circumvent this problem, we use adaptive grid refinement [83], [94]. Instead of refining the grid uniformly everywhere, we only refine a subset of grid cells. We use an error indicator to select the grid cells, which we refine afterwards. For each element we compute an a posteriori error indicator for the discretization error. The cells with a high error are refined, while the cells with a low error are coarsened. This strategy is called *fixed fraction strategy.* With this procedure, we need less computational effort while the discretization error remains small. Figure 8.1 shows on the left hand an uniformly refined grid and on the right hand an adaptively refined grid.

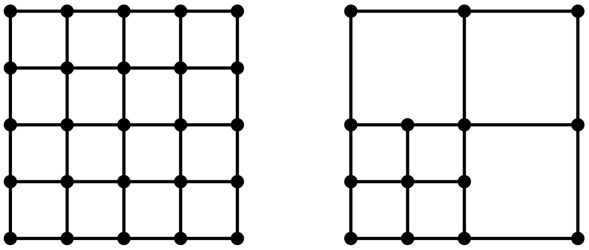


Figure 8.1.: Left hand: uniformly refined grid, right hand: adaptively refined grid.

Optimization: multiple simulations and therefore multiple grids? In our setting of derivative based optimization methods we simulate more than one PDE problem. We simulate primal and tangential PDE problems. To solve the optimization problems with derivative based algorithms, we need accurate and consistent sensitivities. Otherwise the optimization algorithm could not converge or it could converge to a false parameter value. Thus we need to solve all PDE problems, primal and tangential problems, as accurate as possible. That means the discretization error of primal and tangential problems should be low. If we use adaptive grid refinement straight forward we get for each PDE problem one individual adaptively refined grid. Each grid is refined such that the discretization error for the corresponding PDE problem is small. Figure 8.2 shows two different adaptively refined grids.

With this straightforward application of adaptive grid refinement, we observe two main problems that arise with different grids:

1) Practical problem: we set up the tangential problems with the help of the primal problem, because we exploit the common structure of primal and tangential problems. This is difficult with different grids. We need the identical sets $\mathbb{T}_h, \mathbb{F}_h^{int}$ and $\mathbb{F}_h^{\Gamma}$, otherwise our structure exploitation techniques cannot be applied. Additionally, in some cases we need the solution of the primal problem

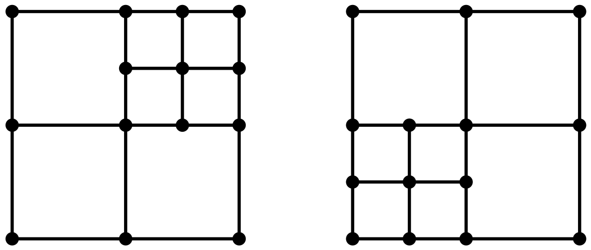


Figure 8.2.: Two different adaptively refined grids.

in the right hand side of the tangential problems.

2) Quality problem: the principle of IND shows that the sensitivity calculated with different grids does not correspond to the computed solution of the simulation of the primal problem. It is not clear, if this sensitivity is consistent, which introduces an error, which can become arbitrary large. We need accurate and consistent sensitivities for the optimization algorithms. Otherwise the optimization algorithms could converge to a false value or converge slowly.

We explain briefly two possible ways to solve practical problem 1). First we could use the smallest common grid. Figure 8.3 shows the building of the smallest common grid from two different adaptively refined grids. On the smallest common grid of a primal and a tangential problem, an exact integration is possible. A drawback of this procedure is the costly implementation. The second way is to interpolate between

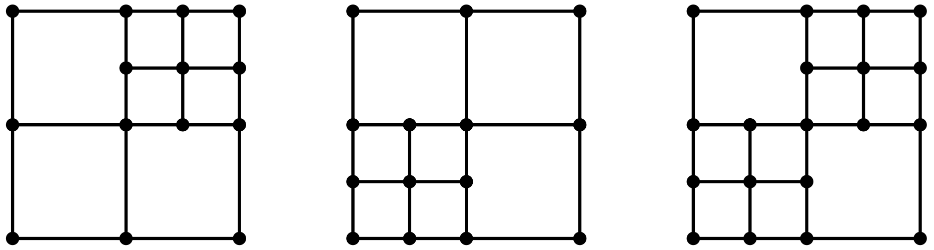


Figure 8.3.: Smallest common grid. Left: grid of primal problem, centre: grid of tangential problem, right: smallest common grid for primal and tangential problem.

grids. By interpolating we introduce an additional error.

Both ways do not solve the quality problem 2). To solve the quality problem 2) and thus transfer the principle of IND to PDEs, we use one grid for all PDE problems. That means we develop a strategy to generate one frozen adaptive grid per optimization iteration. In this way we ensure to compute tangential solutions, which correspond to the primal solution. Thus the computed sensitivities are consistent.

8.1.1 Remark. *Freeze grid for more than one optimization iteration* It is also possible to generate a new adaptive grid not in every optimization iteration, but after several iterations. As long as the discretization errors do not get too large, we can use the same grid. That way we freeze the grid for more than one iteration and thus reduce computational effort [17], [53]. △

Literature overview A sophisticated and evidenced solution are tailored error estimators. For a survey on adaptive mesh refinement with tailored error estimators with the dual-weighted residual method for optimal control problems see [83]. For parameter estimation problems, tailored error estimators, which include the primal and tangential or adjoint solution, are proposed in [17], [21], [22], [34]. They all use continuous finite elements. To our knowledge, for discontinuous finite elements tailored error estimators for parameter estimation problems are not yet developed.

First steps for an a posteriori error estimator for optimum experimental design optimization problems are undertaken in [33]. The author also uses continuous finite elements.

Error sum strategy for grid refinement How do we obtain one grid, which is suitable for primal and tangential PDE problems? We present a possibility to generate one common grid for primal and tangential PDE problems by an a posteriori error indicator.

We first solve the primal and tangential problems on the same coarse mesh. After that, we estimate the errors per cell individually for primal and tangential problems with the help of an error indicator. Before, the individual errors per cell were used to generate individual grids. Instead we now sum up the individual errors cell by cell to obtain the error sums per cell. We use these sums as an error indicator. We refine the 30% cells with the largest values of error sum and coarsen the 3% cells with the lowest values of error sum. Of course, we could choose other percentages to refine and coarsen. With the chosen percentages we approximately double the number of cells in two dimensions. That way, we obtain one adaptively refined grid that is suitable for primal and tangential problems. Algorithm 5 depicts the procedure.

We illustrate the error sum strategy by a small example.

Algorithm 5 Error sum strategy to obtain one common adaptively refined grid.

```
1: Calculate error per cell individually for primal and for tangential problems.
2: Sum up errors per cell of primal and tangential problems.
3: Refine grid according to this error sum indicator.
```

8.1.2 Example. *Error sum strategy for grid refinement.* After computing the primal and tangential solutions on a coarse mesh, we get errors per cell for primal and tangential problems from an error indicator. The errors of the four cells of the coarse grid sorted by size for the primal problem are $e_3^{(1)} > e_2^{(1)} > e_1^{(1)} > e_4^{(1)}$ and for the tangential problem are $e_2^{(2)} > e_1^{(2)} > e_3^{(2)} > e_4^{(2)}$. In the top line of Figure 8.4 we see the primal grid with corresponding errors, the middle line shows the tangential errors. If we refine the cell with the largest error in each case, we would get the upper right grid for the primal problem and the middle right grid for the tangential problem. Thus we would get two differently refined grids.

Instead, we sum up the errors per cell and get the sum of errors per cell. We sort the sums by size and get for this example $\sum_{i=1}^{2} e_2^{(i)} > \sum_{i=1}^{2} e_3^{(i)} > \sum_{i=1}^{2} e_1^{(i)} > \sum_{i=1}^{2} e_4^{(i)}$. In Figure 8.4, the bottom line depicts on the left the sums of errors and on the right the corresponding grid, which results in refining the cell with the highest sum of errors. △

The proposed error sum strategy results in one adaptively refined grid. The refinement is based on the discretization errors of primal and tangential solutions. Different orders of magnitude of the primal and tangential errors could lead to an emphasis on primal or on a tangential problem. This means on the adaptively refined grid based on the error sum strategy, the discretization error of this emphasized PDE problem is much lower than the discretization errors of the remaining PDE problems. A solution for this issue is scaling or weighting the individual discretization errors such that every error contributes the same proportion to the overall error.

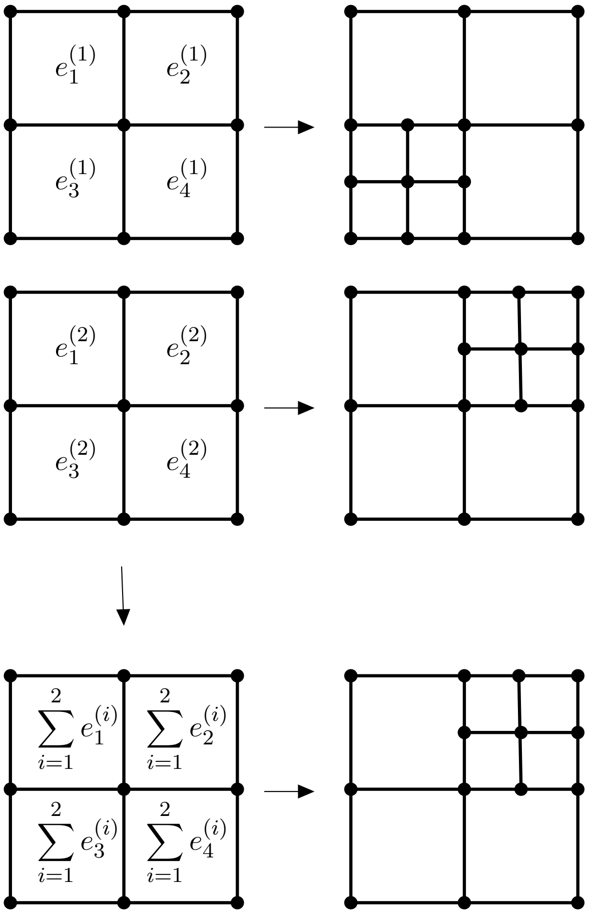

$e_1^{(1)}$
$e_2^{(1)}$
$e_3^{(1)}$
$e_4^{(1)}$
$e_1^{(2)}$
$e_2^{(2)}$
$e_3^{(2)}$
$e_4^{(2)}$
$\sum_{i=1}^{2} e_1^{(i)}$
$\sum_{i=1}^{2} e_2^{(i)}$
$\sum_{i=1}^{2} e_3^{(i)}$
$\sum_{i=1}^{2} e_4^{(i)}$

Figure 8.4.: Error sum strategy for grid refinement. It shows errors per cell and corresponding adaptively refined grids for the primal problem (top), the tangential problem (middle) and the sums of errors (bottom).

8.2. Iterative solver of linear system

To solve the linear system of equations in the course of the finite element algorithm we utilize an iterative solver, because the dimension of the linear system of equations is usually very large. Furthermore, iterative methods take advantage of the sparsity of the stiffness matrix. In our examples we use `GMRES` [86]. The considerations in this section also hold for other iterative solvers.

In the context of sensitivity generation with the principle of IND, the adaptive number of steps taken by the iterative solver is of interest. Depending on the stopping criterion, the iterative solver takes a different number of steps for primal and tangential problems.

For the sensitivity generation, we have two possibilities to handle the iterative solver [46, Chapter 15]:

1) *Piggyback approach*: we apply automatic differentiation to the iterative solver in every iteration. That means, we compute the solution of the primal problem and of the tangential problems in every iteration step. This leads to the use of the same iterative solver and the same grid for all PDE problems.

2) *Two-phase approach*: we assume a negligible residual of the linear system and use the implicit function theorem. We solve the primal and the tangential problems independently by the same iterative solver on the same grid.

Let us investigate these two approaches under the principle of IND. How does the adaptive number of iterations behave?

Piggyback approach The piggyback approach leads to the same number of iterations for primal and tangential problems, because we solve all problems together. The stopping criterion for the iterative solver depends on the accuracy of all solutions. This corresponds to the principle of IND and leads to consistent derivatives.

A practical observation is the following [46]: in the early phase of an iteration process, the convergence behavior of the primal problem is not smooth. Varying step lengths could lead to severe non-differentiablities. In [46], the authors propose a "delayed piggyback" approach. There the derivative evaluation starts later in the iteration process, when the convergence of the primal problem is smooth.

Furthermore, in the piggyback approach, we have to consider the differentiation of a preconditioner and of other sophisticated solution techniques. On the one hand, this introduces difficulties, which have to be investigated in detail to arrive at the accurate sensitivities. A short overview of possible difficulties and solution ways is presented in [46]. On the other hand, the piggyback approach offers possibilities to exploit the structure of the iterative solver and the preconditioner. In the context of DAEs, the author in [15] investigated ways to exploit the structure of an iterative solver under the principle of IND.

Two phase approach The two phase approach leads not necessarily to the same number of iteration steps. Therefore, we iterate until the residual error of the linear equation system is very small. Additional iterations would lead to nearly the same computed solution. So there is no influence of the adaptive number of iteration steps in this very accurate computation setting. Thus, the sensitivities are consistent.

In [46, Lemma 15.1], the authors propose a "consistency check" for the computed sensitivities. It depends on the residuals of primal and tangential problems. Thus if we solve the primal and tangential equation systems until the errors and thus the residuals are very small, we ensure the consistency of the computed sensitivities.

Conclusion We conclude that the piggyback approach is preferable if we desire low accuracies. With the piggyback approach, the adaptive number of iteration steps influences the computed solutions. To get consistent sensitivities in this setting, it is important to use the same number of iteration steps for the primal and the tangential problems. Contrary, if we need high accuracies, the two-step approach computes consistent sensitivities, too. With the two-phase approach, we do not have to consider the differentiation of a preconditioner or other solver-dependent specialities. Thus, we will utilize the two-phase approach with a very accurate stopping criterion of 10^{-10}.

8.2.1 Remark. In the model problems treated in this thesis, high accuracy does not lead to high computing times. This can be the case for three dimensional problems or other large-scale PDE problems. If high accuracy is a major reason for slow convergence, it is advisable to use the piggyback approach with lower accuracies. △

8.2.2 Remark. *Additional adaptive components* There are additional adaptive components, which should be frozen in the sensitivity generation with the principle of IND. Possible additional adaptive components are the order of the finite element and the choice of the finite element. Following the principle of IND, those adaptive components must be chosen the same in primal and tangential problems. Sometimes in the simulation of the primal PDE, to estimate the error by a tailored error estimator, the dual problem is computed on a finer grid or with a more accurate finite element order [30], [31], [33]. In the context of optimization, those are adaptive components, which must be frozen, too. That means, the same finite element with the same order as used for the solution of the primal problem must be used for the tangential problems. △

8.3. Numerical results

8.3.1 Example. *Sensitivity evaluation with frozen adaptivity for advection dominated diffusion advection PDE problem.* We utilize the same example as in Chapter 7 on structure exploitation, Example 7.3.1, to illustrate the frozen adaptivity for sensitivity evaluation. The underlying PDE model problem is an advection dominated diffusion advection problem. The domain Ω consists of the rectangle $[-1,1]^2$. The strong form reads

$$\begin{aligned} -0.001\varDelta y + \beta(p)\cdot\nabla y &= 1 \quad \text{on } \Omega, \\ y &= 1 \quad \text{on } \varGamma = \partial\Omega. \end{aligned}$$

As before the unknown parameters are the two components of the advection direction $\beta(p) = (p_1, p_2)^T$. The corresponding discrete PDE problem reads: Find $y_h \in V_h$ such that

$$\alpha a_h(y_h, v_h) + b_h(p; y_h, v_h) = f_h(v_h), \quad \forall v_h \in V_h,$$

with $a_h(y_h, v_h)$, $b_h(p; y_h, v_h)$ and $f_h(v_h)$ defined in equation (7.10). We use the differentiable discretization $\sigma_\mu(\beta, n)$ defined in equation (6.3) with $\mu = 0.1$. The model response consists of point measurements given by the value of the discrete PDE solution operator at the measurement points $h_{i,h}(p) = S_h(p)\Big|_{x=x_i^m}$. Thus the discrete parameter estimation problem reads

$$\min_{p\in P} \frac{1}{2}\sum_{i=1}^{M}\left(\frac{\eta_i - h_{i,h}(p)}{\sigma_i}\right)^2.$$

First we present the non-frozen setting, afterwards we apply the developed error sum strategy, Algorithm 5, to realize the frozen adaptivity and get one common grid. For the simulations in this example, we choose $p_1 = -0.2$ and $p_2 = 0.3$. As before, we use the same quadratic discontinuous finite elements for all computations.

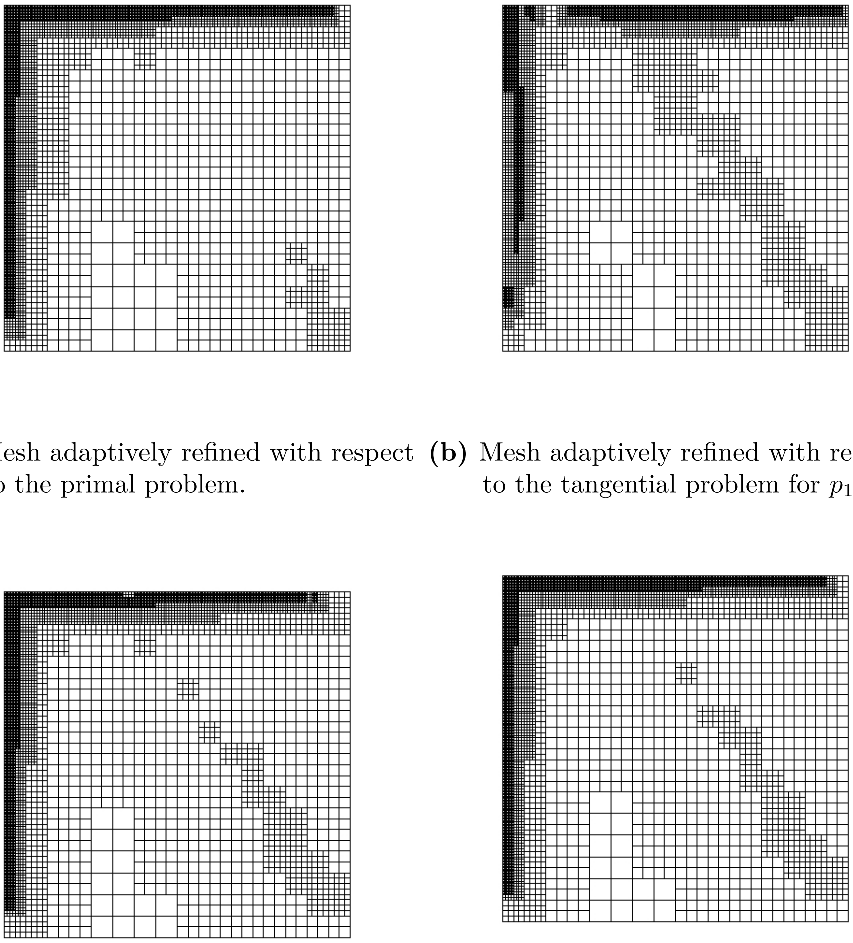
In the non-frozen case we get three different grids, one for each PDE problem: the primal problem, the tangential problem with respect to parameter p_1 and the tangential problem with respect to parameter p_2. Each grid is refined with respect to the discretization error of one of the just listed PDE problems. For the discretization error indication we use the Kelly refinement indicator [37], [62] which approximates the discretization error per cell of a PDE solution by the integration of the jump of a gradient between cells. For further details on the choice of error indicator see Remark 8.3.2. We start on a uniform refined grid with 2,304 DoFs and perform 3 refinement cycles. We arrive at adaptively refined grids with approximately 15,800 DoFs. Figure 8.5a shows the resulting grid, which is refined with respect to the primal problem. Figures 8.5b and 8.5c show the grids, which are refined with respect to the tangential problems. Figure 8.5b shows the grid for the first parameter p_1 and Figure 8.5c shows the grid for the second parameter p_2.

In contrast to that, our aim in the frozen adaptivity case is to generate one grid for all three PDE problems per Gauss-Newton iteration. Therefore we apply the developed error sum strategy, depicted in Section 8.1. It includes the discretization errors of all three PDE problems to refine the grid. The error sum strategy leads to the grid in Figure 8.5d.

We compute the solutions of all three PDE problems on this common grid with 15,885 DoFs. Furthermore, we use the two-phase approach and solve the three PDE problems independently with a very accurate stopping criterion of 10^{-10} for the iterative solver. Figure 8.6 shows the simulation of the primal problem on the common grid generated by the developed error sum strategy. Figure 8.7 shows the simulation of the first tangential problem (left) and the simulation of the second tangential problem (right). Compared to the simulations on the uniform grid in the previous chapter, these simulations on the adaptive grid look the same. Thus we successfully simulated the primal and tangential PDE problems on one adaptive grid. △

8.3.2 Remark. *Choice of error indicator.* The Kelly refinement indicator approximates the discretization error per cell of a PDE solution by the integration of the jump of a gradient between cells. For an advection dominated problem, sharp edges are part of the solution. Thus this error indicator is not the best choice in this setting. A residuum based error estimator, see for example [94], or a duality based error estimator [20] are better suited. Nonetheless our aim to show the generation of one common grid is illustrated well with this error indicator. △

By utilizing the frozen adaptivity case and computing all PDE problems on one common grid, we get accurate sensitivities. The computed sensitivities correspond to the solution of the primal problem and are thus consistent. This consistency of the sensitivities is important for the optimization algorithms to converge and to converge to the ”true“ optimum. Thus, with our developed error sum strategy for grid refinement to generate one common grid which is suitable for primal and tangential problems and the selection of the two-phase approach for the adaptive step number of the iterative solver, we evaluated the consistent sensitivities. With these sensitivity generation techniques we numerically solve the PE and OED optimization problems in the next chapters.



(a) Mesh adaptively refined with respect to the primal problem.

(b) Mesh adaptively refined with respect to the tangential problem for p_1.

(c) Mesh adaptively refined with respect to the tangential problem for p_2.

(d) Mesh adaptively refined with the error sum strategy. It includes the discretization errors of primal problem, tangential problem for p_1 and tangential problem for p_2 for the grid refinement.

Figure 8.5.: Different adaptively refined grids. Three grids (a), (b), (c) for the non-frozen adaptivity case and one grid (d) for the frozen adaptivity case.

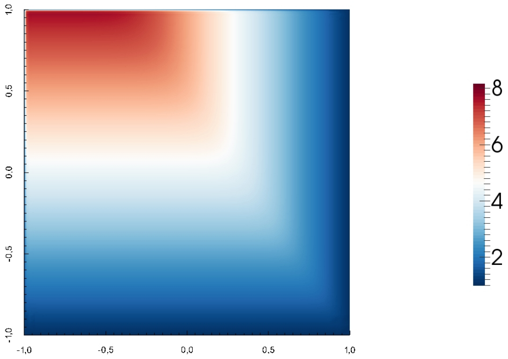


Figure 8.6.: Simulation of the primal problem on the common grid generated by the developed error sum strategy with 15,885 DoFs, see Figure 8.5d.

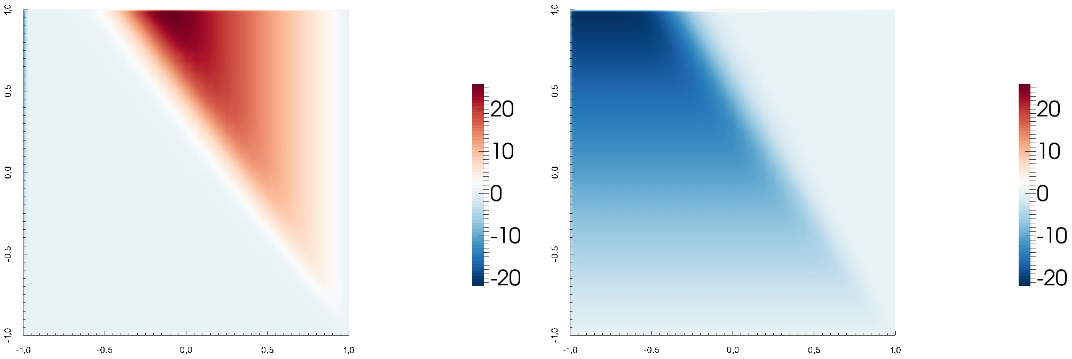


Figure 8.7.: Simulation of the tangential problems with respect to p_1 (left) and p_2 (right) on the common grid generated by the developed error sum strategy with 15,885 DoFs, see Figure 8.5d.

Part V.

Implemented software and numerical results for PE and OED

9. The software SeafaND-Optimizer

In this chapter the developed software SeafaND-Optimizer, short for *structure exploiting and frozen adaptivity numerical differentiation optimizer*, is introduced. First we give an overview over the software SeafaND-Optimizer. After that, we explain the program structure. Finally, we describe the workflow of the software SeafaND-Optimizer.

9.1. Overview

The SeafaND-Optimizer is a software for simulation, parameter estimation and optimum experimental design with diffusion advection reaction PDE models. For simulation of stationary 2D diffusion advection reaction PDE models, the discontinuous Galerkin methods depicted in Chapter 4 are implemented. Due to the modular architecture, an extension to other PDE model problems is possible. The core part of the software SeafaND-Optimizer consists of the efficient implementation of the developed techniques for sensitivity generation for PE and OED depicted in the three previous chapters on differentiable stabilization Chapter 6, on structure exploitation Chapter 7 and on frozen adaptivity Chapter 8. With these techniques, the solution of the PE and OED problems is possible with low memory usage and low computational effort.

A benefit of the software SeafaND-Optimizer is that the user does not have to set up the sensitivities, that means the tangential problems. Only the primal problem has to be characterized by domain, boundary conditions, right hand side and parameter values. Consistent tangential equations are generated by the program.

In addition, further options of optimization software and finite element library are available in the program. For example globalization strategies in optimization or different finite element mesh designs can be selected by the user.

9.2. Program structure

The SeafaND-Optimizer consists of a modular structure. For the simulation of the PDE models, the functionalities of the finite element library dealii [8], [12] and the experimentation suite Amandus [61] are available in the SeafaND-Optimizer. For the optimization problems, we use the data interface and optimization algorithms of VPLAN [66]. We follow an object-oriented approach by establishing a new class in VPLAN, wherein the simulating and differentiating of the PDE problem takes place. The advantage of this approach is that in every optimization step the whole information about the structure of solution and sensitivities is available. We implemented the developed techniques for sensitivity generation in this new class.

VPLAN [66] is a software for simulation, parameter estimation and optimum experimental design with ordinary differential equations (ODE) and differential algebraic equations (DAE). It consists of a modular structure with individual modules for simulation of ODEs and DAEs, sensitivity generation, parameter estimation and optimum experimental design. In the software SeafaND-Optimizer we utilize the VPLAN modules for optimization, that means for parameter estimation and optimum experimental design.

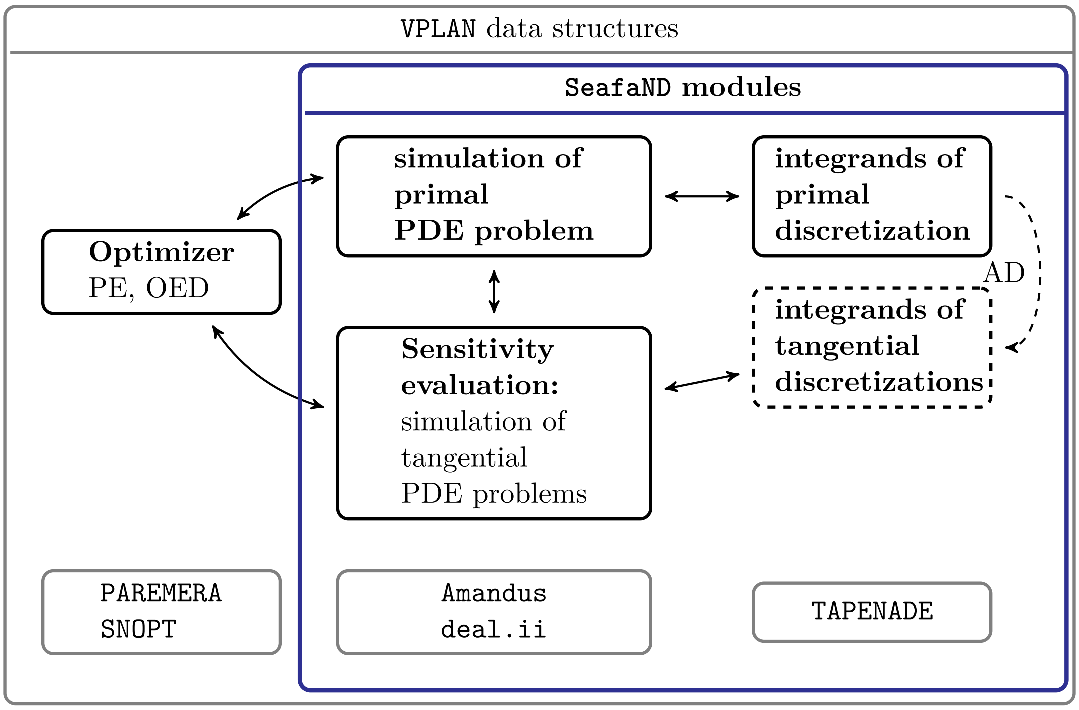

VPLAN data structures
SeafaND modules
simulation of primal PDE problem
integrands of primal discretization
Optimizer
PE, OED
AD
integrands of tangential discretizations
Sensitivity evaluation:
simulation of tangential PDE problems
PAREMERA
SNOPT
Amandus
deal.ii
TAPENADE

Figure 9.1.: Structure of the SeafaND-Optimizer.

Figure 9.1 shows the modular structure of the software SeafaND-Optimizer. As mentioned before, it is embedded in the data structures of VPLAN. Depending on the task, simulation, PE or OED, the optimizer provides all necessary information for the SeafaND modules. Here, the simulation and differentiation of the primal PDE problem takes place.

For simulation of stationary 2D diffusion advection reaction PDE models, the interior penalty discontinuous Galerkin method and the standard upwind discontinuous Galerkin method are implemented. Furthermore, we implemented the developed differentiable stabilization for the upwind discontinuous Galerkin method from Chapter 6 for the advection part of the PDE problem. The simulation relies on the finite element library `dealii` and the experimentation suite `Amandus`.

For sensitivity generation, we transferred the principle of IND to PDEs. One main aspect of this transfer are the structure exploitation techniques developed in Chapter 7. In particular, we exploit the structure of the FE method by only deriving single integrands. We do not derive the complete FE code. To realize this structure exploitation, we outsource the single integrands of the FE discretization to a separate file. Only this file is differentiated by an AD tool. In Figure 9.1, the box "integrands of primal discretization" represents those inner parts. The dashed line shows the generation of the sensitivity-files by an AD tool. For each tangential problem, the AD tool generates the "integrands of tangential discretization". We discretize the tangential problems by taking the generated sensitivity files ("integrands of tangential discretization" in the figure) for the discretization of the right hand side of the tangential problem. For the left hand side of the tangential problems we reuse the discretization of the left hand side of the primal problem. That way, we efficiently and automatically generate the tangential PDE problems. With the generated code, we simulate the tangential PDE problems and set up the required sensitivities for the optimization algorithms.

Figure 9.1 further depicts the state-of-the-art software utilized in the single steps of the procedure. In the optimizing steps, we use `PAREMERA` [63] to solve the PE problem or `SNOPT` [42], [43] to solve the OED problem. The computational solution of the PDE model problems is implemented with the help of `Amandus` [61] and `deal.ii` [8], [12]. The inner part files are automatically differentiated by the AD tool `TAPENADE` [48]. The packages `TAPENADE`, `PAREMERA` and `SNOPT` can be interchanged with other packages that are interfaced with `VPLAN`. For example, we could also use `PARFIT` [25], [88] for the PE problem.

Another main aspect of the transfer of the principle of IND to PDEs for sensitivity generation is the frozen adaptivity depicted in Chapter 8. We implemented the developed methods. For the iterative solver of the linear system we choose the two-phase approach, that means we solve each primal and tangential PDE problem independently. One adaptive grid is generated for all PDE problems via the developed error sum strategy in Section 8.1.

Algorithm 6 Frozen adaptivity in the software `SeafaND-Optimizer`

```
for $n = 1$ to $n_c$ do
    if $n = 1$ then
        generate coarse start grid
    else
        refine grid adaptively by error sum strategy
    end if
    Assemble and solve primal and tangential equation systems independently.
end for
Evaluate measurement points for primal and tangential problems and transfer
values to optimizer.
```

Algorithm 6 depicts the setting. In the first step, we generate a coarse start grid. After that, we assemble and solve primal and tangential problems independently but on the same coarse grid. In the next iteration, we refine the grid adaptively with the developed error sum strategy. Thereafter, primal and tangential problems are again assembled and solved independently on the same adaptively refined grid. We repeat this procedure, until we reach the number of refinement cycles n_c prescribed by the user. Finally we evaluate the solutions of the primal and tangential problems in the measurement points and transfer the values to the optimizer.

9.3. Workflow of the `SeafaND-Optimizer`.

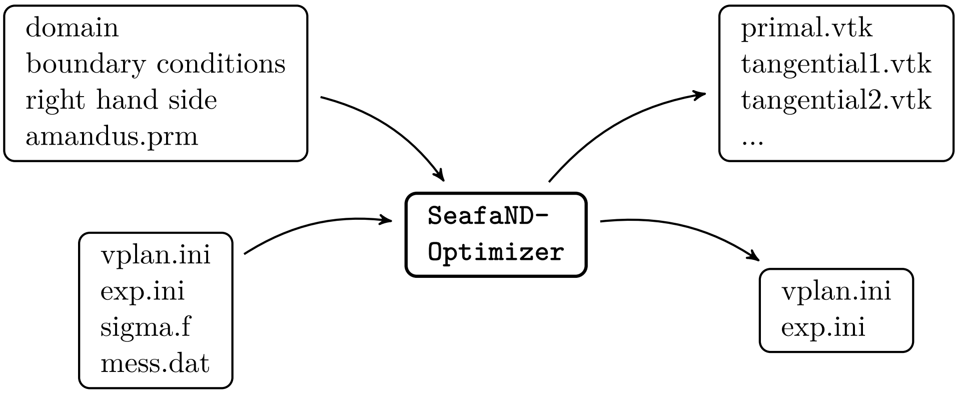


Figure 9.2.: Input and output of the `SeafaND-Optimizer`.

Figure 9.2 shows the workflow of the `SeafaND-Optimizer`. As input, the user has to provide two kinds of information, that is information about the PDE model problem and about the optimization problem. First, for the PDE model problem, we need the domain, the boundary conditions, the right hand side and the initial parameter values. We define the parameter values concerning the PDE model problem in a prm-file. With that we specify a diffusion advection reaction PDE model problem. As default the domain is the unit rectangle, the boundary conditions are constant Dirichlet boundary conditions along the whole boundary and the right hand side is a constant function. Many examples of more complicated domains, boundary conditions and right hand sides can be found in the `deal.ii`-library [8], [12] and can be applied within the `SeafaND-Optimizer`.

Second, for the optimization problems, the user has to specify ini-files, fortran-files and mess-files. There is one main input file, vplan.ini, where we specify which action the program performs: (S)imulation, (P)E or (V) for OED. Furthermore, for every experiment we create a separate exp.ini file. The most important specification for PE is, which parameters to estimate. For OED, an important specification is the placement of the possible measurement points with the corresponding sampling decisions. We use the main fortran-file differently than in the ODE or DAE case. In our PDE setting, the small inner parts with the integrands of the primal discretization are contained in the ffcnode.f file. That means, the user does not have to specify the dynamic model. Finally, the mess-files contain the measurement values. There are many more options, which can be set for the optimization, see [56], [63], [66] for more details.

After the run of the `SeafaND-Optimizer`, we get as output files vtk-files and ini-files. The vtk-files contain the simulation of the primal and tangential PDE model problems. They can be visualized by for example the software `ParaView` [2]. The ini-files contain all information regarding the optimization run: estimated parameters, optimized sampling decisions and values of the least squares functional.

10. Numerical results for PE and OED

This chapter presents the numerical results obtained with the developed methods implemented in the software `SeafaND-Optimizer` depicted in Chapter 9. We investigate the parameter estimation problem and the optimum experimental design problem constrained by a stationary 2D advection dominated diffusion advection PDE boundary value problem. The main challenge in the simulation of this PDE model is the small diffusion factor, which leads to advection domination. Furthermore, for the optimization problems, we are concerned with the correctness of the computed sensitivities. Without consistent sensitivities, the optimization algorithms are likely to not converge or they could converge to a wrong parameter value.

We begin this chapter with a section about the parameter estimation problem. First, we describe the problem formulation. After that, we show results obtained using three different sets of measurement data perturbed by different amounts of noise. The subsequent section shows results for the optimum experimental design problem. We begin with the problem formulation for sequential OED. After that, we show computational results for sequential OED. Finally, we perform a numerical study for different diffusion coefficients. This study demonstrates the applicability of the developed methods not only for one specific setting, but for a class of problems.

Throughout this chapter, we generate measurement data η_i by simulating the PDE model problem on a very fine grid. The grid is uniformly refined with 589,824 degrees of freedom (DoFs). For the simulation of measurement data, we use the standard upwind discretization (4.5) and quadratic discontinuous finite elements. For the optimization algorithms for PE and OED, we use for the simulation of the PDE problems the differentiable discretization $\sigma_\mu(\beta, n)$ defined in equation (6.3) with $\mu = 0.1$. We select for the iterative solver `GMRES` of the PDE simulation an accurate stopping criterion of 10^{-10}, due to the considerations on the handling of an iterative solver in Section 8.2.

10.1. Parameter estimation with different noise levels

In this section we perform parameter estimation with a stationary 2D advection dominated diffusion advection PDE problem. We select the same PDE model problem as in Examples 7.3.1 and 8.3.1. The sensitivities computed there are utilized here in the Gauss-Newton algorithm for parameter estimation.

10.1.1. Problem formulation: Parameter estimation with 2D advection dominated diffusion advection PDE problem

The strong form of the underlying PDE model problem reads

$$\begin{aligned} -0.001\Delta y + \beta(p)\cdot\nabla y &= 1 \quad \text{on } \Omega, \\ y &= 1 \quad \text{on } \Gamma = \partial\Omega. \end{aligned}$$

The domain Ω consists of the rectangle $[-1,1]^2$. The corresponding discrete PDE problem reads: Find $y_h \in V_h$ such that

$$\alpha a_h(y_h, v_h) + b_h(p; y_h, v_h) = f_h(v_h), \quad \forall v_h \in V_h,$$

with $a_h(y_h, v_h)$, $b_h(p; y_h, v_h)$ and $f_h(v_h)$ defined in equation (7.10).

The discrete parameter estimation problem reads

$$\min_{p\in P} \frac{1}{2}\sum_{i=1}^{m}\left(\frac{\eta_i - h_{i,h}(p)}{\sigma_i}\right)^2 .$$

We estimate two parameters, which are the components of the advection direction $\beta(p) = (p_1, p_2)^T$. The model response η_i consists of point measurements given by the value of the discrete PDE solution operator at the measurement points $h_{i,h}(p) := S_h(p)\Big|_{x=x_i^m}$. There are eight measurement points with coordinates

$$\begin{aligned} x_1^m &= (-0.5, 0.8), & x_2^m &= (0.5, 0.8), \\ x_3^m &= (0.8, 0.5), & x_4^m &= (0.8, -0.5), \\ x_5^m &= (0.5, -0.8), & x_6^m &= (-0.5, -0.8), \\ x_7^m &= (-0.8, -0.5), & x_8^m &= (-0.8, 0.5). \end{aligned}$$

Figure 10.1 shows the placement of the measurement points in the domain $\Omega = [-1,1]^2$.

The “true” parameter values we use for the generation of the measurement data are

$$p_1^* = -0.2, \quad p_2^* = 0.3.$$

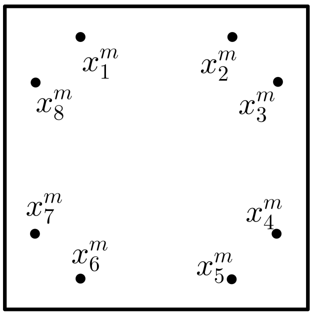


Figure 10.1.: Placement of the measurement points $x_i^m, i = 1, .., 8$, in the domain $\Omega = [-1, 1]^2$.

We generate three different sets of measurement data. We disturb the values of the measurement functions by an additive normally distributed error with zero mean and three different standard deviations, $\sigma_i = 0.1, \sigma_i = 0.2$, and $\sigma_i = 0.3, i = 1, .., 8$. This results in variances of $\sigma_i^2 = 0.01, \sigma_i^2 = 0.04$, and $\sigma_i^2 = 0.09, i = 1, .., 8$, respectively. Hence, we get three sets of measurement data with 1%, 4% and 9% noise and corresponding perturbations:

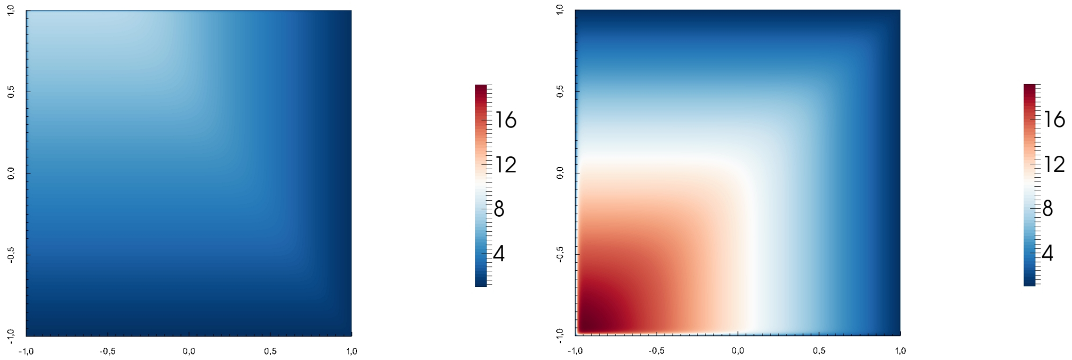
$$\begin{aligned} 1\% : \quad & \varepsilon = (0.040, -0.066, -0.019, -0.004, 0.023, -0.009, 0.019, -0.008), \\ 4\% : \quad & \varepsilon = (0.048, 0.197, -0.150, 0.121, 0.157, 0.048, 0.312, 0.095), \\ 9\% : \quad & \varepsilon = (-0.390, 0.393, 0.251, 0.488, 0.355, 0.352, -0.536, 0.034). \end{aligned}$$

We choose the start values of the parameters for the parameter estimation as

$$p_1^0 = -0.1, \quad p_2^0 = -0.1.$$

Note that the start value of the second parameter p_2^0 has a different sign than the ”true“ parameter p_2^*. This introduces a difficulty for the optimization algorithm, because the outcome of the PDE simulation changes.

Figure 10.2a shows a simulation of the primal PDE model problem for the “true” parameter values on the very fine uniform refined grid with 589,824 DoFs, with the standard upwind discretization. Figure 10.2b shows a simulation with the start values on a grid, which is adaptively refined by the developed error sum strategy. The grid has 131,148 DoFs and is simulated with the developed differentiable discretization. We see the influence of the parameters on the state: the “true” advection direction points from the lower right corner to the upper left corner, while the start advection direction points from the upper right corner to the lower left corner. Moreover the numerical values of the states differ: for the “true” parameters they are much lower than for the start parameters.



(a) Simulation primal problem with ”true“ parameter values, $p_1^* = -0.2$ and $p_2^* = 0.3$, on very fine uniform refined grid with 589,824 DoFs, with standard stabilization $\sigma_{upw}(\beta, n)$.

(b) Simulation primal problem with start values, $p_1^0 = -0.1$ and $p_2^0 = -0.1$, with developed error sum strategy adaptively refined grid with 131,148 DoFs, with developed differentiable stabilization $\sigma_\mu(\beta, n)$.

Figure 10.2.: Simulations of primal PDE problem with different parameter values, grid refinements and stabilizations.

For the sensitivity generation, we use the structure exploiting techniques depicted in Chapter 7. Furthermore we realize the frozen adaptivity from Chapter 8. We simulate the primal PDE problem and the two tangential PDE problems on one common grid. In each Gauss-Newton step, the error sum strategy depicted in Section 8.1 generates one new common grid. We start on a uniform refined grid with 36,864 DoFs and perform three refinement cycles. The generated grids consist of approximately 130,000 DoFs. Figure 10.3 shows the generated grid for the first optimization step with start values $p_1^0 = -0.1$ and $p_2^0 = -0.1$ for the parameters.

We choose for the Gauss-Newton algorithm as step size strategy a sophisticated globalization strategy: the restricted monotonicity test (RMT) developed by [29], implemented in the software `PAREMERA` by [63].

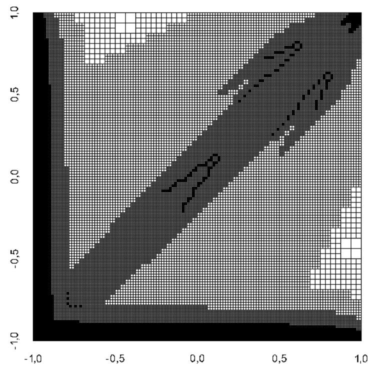


Figure 10.3.: Mesh adaptively refined with error sum strategy with 131,148 DoFs, for start values $p_1^0 = -0.1$ and $p_2^0 = -0.1$.

10.1.2. Comparison of noise levels

We perform parameter estimation with three sets of measurement data, each one disturbed by a different amount of noise with a different perturbation: $1\%, 4\%$ and 9% noise. We perform all computations with the developed software `SeafaND-Optimizer`, depicted in Chapter 9.

Table 10.1 shows results of the parameter estimation with the Gauss-Newton algorithm including our developed sensitivity generation methods for these three different noise levels. In the case of a rather high amount of 9% noise, the algorithm needs 4 (respectively, 3) iteration steps more to reach the stopping criterion in equation (2.13) with $tol = 10^{-6}$ compared to the lower amounts of 1% (respectively, 4%) noise. Similarly, the estimated values for $\hat{p}_1$ and $\hat{p}_2$ are the more far off the higher the noise level gets. That is what we expect, with a higher amount of noise it is more difficult to estimate the parameters correctly.

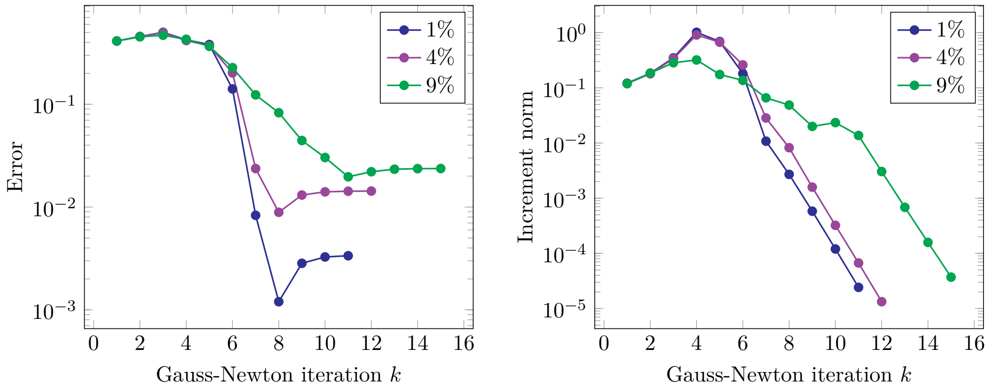
Figure 10.4 depicts for the three noise levels in Figure 10.4a the error in the Euclidian norm between estimated parameters $\hat{p}$ and "true" parameters p^* for every iteration k of the Gauss-Newton algorithm. In Figure 10.4b the increment $\|\delta p_k\|_2$ for every Gauss-Newton iteration k for the three noise levels is shown. Note that we use a semi logarithmic scale.

Let us begin with the errors in Figure 10.4a. In the first iterations, the errors increase, after that they decrease rapidly. We see linear convergence for all three noise levels

noise level	$\hat{p}_1$	$\hat{p}_2$	# iteration
1%	-0.2033	0.3004	11
4%	-0.1857	0.2991	12
9%	-0.1862	0.2807	15

Table 10.1.: Results of the parameter estimation with the Gauss-Newton algorithm for different noise levels, $1\%, 4\%$ and 9%. The first column depicts the noise level, the second and third column depict the estimated parameter values $\hat{p}_1$ and $\hat{p}_2$ and the fourth column depicts the number of iterations.

after iteration $k = 5$. This is what we expected from the convergence theory of the Gauss-Newton algorithm. For a lower noise level of 1% the errors get much smaller than for the higher noise levels of 4% and 9%. For a lower noise level, the measurements are more exact and thus the parameters can be estimated more precisely. After iteration 10 or 12, respectively, we see an asymptotic behavior of the error, the values do not decrease any more. Why does the Gauss-Newton algorithm not stop, if no improvement is made? An explanation for this question can be found in Figure 10.4b. The increment decreases until the last iteration step. The Gauss-Newton algorithm will only stop if the stopping criterion in equation (2.13) is fulfilled, which corresponds to a small increment $\|\delta p_k\|_2$. Here, we set $tol = 10^{-5}$.



(a) Error $\|\hat{p}_k - p^*\|_2$ in the Euclidian norm between estimated parameters $\hat{p}_k$ and "true" parameters p^* for every iteration k of the Gauss-Newton algorithm.

(b) Norm of the increment $\|\delta p_k\|_2$ for every Gauss-Newton iteration k is shown.

Figure 10.4.: Results of the parameter estimation with the Gauss-Newton algorithm for different noise levels, $1\%, 4\%$ and 9%.

Tables 10.2, 10.3 and 10.4 report the results of the parameter estimations corresponding to Figure 10.4 for 1%, 4% and 9% noise, respectively. The objective functional values (LS) and the increment values are reduced until the increment $\|\delta p_k\|_2$ fulfills the stopping criterion in equation (2.13).

Finally, let us take a closer look at the individual parameters. Figure 10.5 shows the relative errors in parameters p_1 and p_2 during the course of the Gauss-Newton iterations k for the three noise levels. We see in all three noise levels a similar behaviour: the relative errors in both parameters at first increase, after the third and fifth iteration, respectively, they decrease rapidly. In the fifth iteration the signs of the second parameter estimate $\hat{p}_2$ change, see Tables 10.2, 10.3 and 10.4. That leads to different outcomes of the PDE simulations. After this difficulty of changing the sign is solved, all relative errors decrease rapidly. After the tenth and twelfth iteration, respectively, we see for all noise levels an asymptotic behavior of both parameters. In Figure 10.5c for 9% noise, in iteration 11 the relative error of parameter $\hat{p}_1$ decreases rapidly, but after that iteration, the error increases again. This behavior can be explained by the values in Table 10.4: the estimate of $\hat{p}_{1,k}$ surpasses the true value $p_1^* = -0.2$:

$$\hat{p}_{1,10} = -0.2185, \; \hat{p}_{1,11} = -0.2003, \; \hat{p}_{1,12} = -0.1893,$$

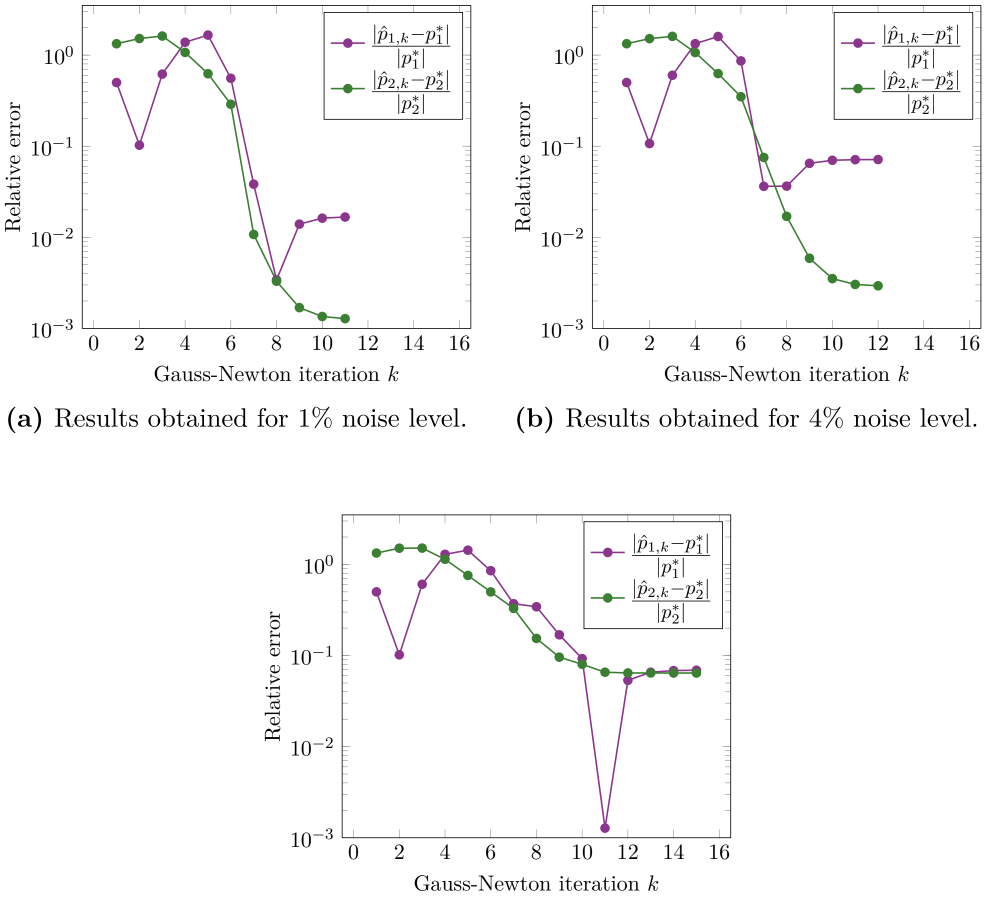
and converges to the value -0.1893. Due to the high noise level, only this less accurate value of -0.1893 is achievable.

Taking a closer look at the asymptotic behavior, we see that the parameters p_1 slightly increase, while the parameters p_2 slightly decrease. This leads to the decrease of the norm of the increment as we have seen in Figure 10.4b. But the overall error $\|\hat{p} - p^*\|_2$, in Figure 10.4a, does not improve. The algorithm reached the asymptotic area where no overall improvement is possible any more. A stopping criterion with $tol = 10^{-4}$ would lead to a similarly good estimation.

Altogether, the parameter estimation algorithm converges linearly for different sets of measurement values disturbed with different sizes of standard deviations. This is a hint, that the developed methods for sensitivity generation produce consistent sensitivities.

Iteration k	LS	Increment $\|\delta p_k\|_2$	Step size t_k	$\hat{p}_1$	$\hat{p}_2$
1	$4.1104 \cdot 10^4$	$1.2138 \cdot 10^{-1}$	1.0000	-0.1000	-0.1000
2	$1.4856 \cdot 10^4$	$1.8401 \cdot 10^{-1}$	1.0000	-0.1795	-0.1558
3	$6.3022 \cdot 10^3$	$3.4892 \cdot 10^{-1}$	0.8048	-0.3237	-0.1855
4	$2.3067 \cdot 10^3$	$1.0149 \cdot 10^0$	0.6299	-0.4767	-0.0211
5	$1.7297 \cdot 10^3$	$6.9071 \cdot 10^{-1}$	0.4386	-0.5317	0.4874
6	$4.4066 \cdot 10^2$	$1.8156 \cdot 10^{-1}$	1.0000	-0.3113	0.3866
7	$3.6135 \cdot 10^0$	$1.0795 \cdot 10^{-2}$	1.0000	-0.1923	0.3032
8	$6.6654 \cdot 10^{-1}$	$2.7054 \cdot 10^{-3}$	1.0000	-0.2007	0.3010
9	$5.1025 \cdot 10^{-1}$	$5.8114 \cdot 10^{-4}$	1.0000	-0.2028	0.3005
10	$5.0331 \cdot 10^{-1}$	$1.2005 \cdot 10^{-4}$	1.0000	-0.2032	0.3004
11	$5.0302 \cdot 10^{-1}$	$2.4177 \cdot 10^{-5}$	1.0000	-0.2033	0.3004

Table 10.2.: Results of the parameter estimation with the Gauss-Newton algorithm for 1% noise level. The first column depicts the iteration number k, the second column the least squares objective value (LS), the third column the norm of the increment $\|\delta p_k\|_2$, the fourth column the step size t_k, the fifth and sixth column depict the estimated parameter values $\hat{p}_1$ and $\hat{p}_2$.



(a) Results obtained for 1% noise level.

(b) Results obtained for 4% noise level.

(c) Results obtained for 9% noise level.

Figure 10.5.: Results of the parameter estimation with the Gauss-Newton algorithm for different noise levels, 1% **(a)**, 4% **(b)** and 9% **(c)**. Relative errors in the single parameters p_1 (purple) and p_2 (green) for every Gauss-Newton iteration k.

Iteration	LS	Increment $\|\|\delta p_k\|\|_2$	Step size	$\hat{p}_1$	$\hat{p}_2$
1	$1.0064 \cdot 10^4$	$1.1987 \cdot 10^{-1}$	1.0000	-0.1000	-0.1000
2	$3.6418 \cdot 10^3$	$1.7990 \cdot 10^{-1}$	1.0000	-0.1786	-0.1549
3	$1.5419 \cdot 10^3$	$3.3766 \cdot 10^{-1}$	0.8082	-0.3200	-0.1816
4	$5.6578 \cdot 10^2$	$9.0849 \cdot 10^{-1}$	0.7029	-0.4669	-0.0201
5	$4.4952 \cdot 10^2$	$6.7136 \cdot 10^{-1}$	0.3150	-0.5199	0.4880
6	$2.1038 \cdot 10^2$	$2.6090 \cdot 10^{-1}$	1.0000	-0.3727	0.4047
7	$1.5069 \cdot 10^1$	$2.8411 \cdot 10^{-2}$	1.0000	-0.2073	0.2774
8	$4.7236 \cdot 10^0$	$8.2261 \cdot 10^{-3}$	1.0000	-0.1927	0.2949
9	$4.1204 \cdot 10^0$	$1.5849 \cdot 10^{-3}$	1.0000	-0.1870	0.2982
10	$4.0956 \cdot 10^0$	$3.2211 \cdot 10^{-4}$	1.0000	-0.1860	0.2989
11	$4.0946 \cdot 10^0$	$6.6987 \cdot 10^{-5}$	1.0000	-0.1858	0.2991
12	$4.0945 \cdot 10^0$	$1.3293 \cdot 10^{-5}$	1.0000	-0.1857	0.2991

Table 10.3.: Results of the parameter estimation with the Gauss-Newton algorithm for 4% noise level. The first column depicts the iteration number k, the second column the least squares objective value (LS), the third column the norm of the increment $\|\delta p_k\|_2$, the fourth column the step size t_k, the fifth and sixth column depict the estimated parameter values $\hat{p}_1$ and $\hat{p}_2$.

Iteration	LS	Increment $\|\|\delta p_k\|\|_2$	Step size	$\hat{p}_1$	$\hat{p}_2$
1	$4.4598 \cdot 10^3$	$1.1904 \cdot 10^{-1}$	1.0000	-0.1000	-0.1000
2	$1.6433 \cdot 10^3$	$1.8507 \cdot 10^{-1}$	0.9554	-0.1796	-0.1522
3	$6.7221 \cdot 10^2$	$2.8486 \cdot 10^{-1}$	0.7764	-0.3211	-0.1533
4	$2.8738 \cdot 10^2$	$3.2037 \cdot 10^{-1}$	0.4600	-0.4583	-0.0416
5	$2.5397 \cdot 10^2$	$1.7441 \cdot 10^{-1}$	1.0000	-0.4868	0.0728
6	$1.2009 \cdot 10^2$	$1.3663 \cdot 10^{-1}$	1.0000	-0.3707	0.1502
7	$4.3211 \cdot 10^1$	$6.5917 \cdot 10^{-2}$	1.0000	-0.2741	0.2013
8	$2.5033 \cdot 10^1$	$4.8742 \cdot 10^{-2}$	1.0000	-0.2687	0.2538
9	$1.2678 \cdot 10^1$	$1.9997 \cdot 10^{-2}$	1.0000	-0.2338	0.2712
10	$1.0881 \cdot 10^1$	$2.3448 \cdot 10^{-2}$	1.0000	-0.2185	0.2759
11	$9.3447 \cdot 10^0$	$1.3687 \cdot 10^{-2}$	1.0000	-0.2003	0.2803
12	$8.8875 \cdot 10^0$	$3.0424 \cdot 10^{-3}$	1.0000	-0.1893	0.2807
13	$8.8613 \cdot 10^0$	$6.8301 \cdot 10^{-4}$	1.0000	-0.1869	0.2807
14	$8.8599 \cdot 10^0$	$1.5801 \cdot 10^{-4}$	1.0000	-0.1863	0.2807
15	$8.8598 \cdot 10^0$	$3.7061 \cdot 10^{-5}$	1.0000	-0.1862	0.2807

Table 10.4.: Results of the parameter estimation with the Gauss-Newton algorithm for 9% noise level. The first column depicts the iteration number k, the second column the least squares objective value (LS), the third column the norm of the increment $\|\delta p_k\|_2$, the fourth column the step size t_k, the fifth and sixth column depict the estimated parameter values $\hat{p}_1$ and $\hat{p}_2$.

10.2. Sequential optimum experimental design: Case study with different diffusion coefficients

In this section we show an optimum experimental design problem. We perform sequential optimum experimental design, that means we first perform parameter estimation, after that we perform optimum experimental design and finally a second parameter estimation with the optimized sampling points. We generate the sensitivities by the strategies developed in Chapter 6 on the differentiable upwind discontinuous Galerkin discretization, Chapter 7 on structure exploitation and Chapter 8 on frozen adaptivity.

We begin this section with the problem formulation of sequential OED. After that, we show computational results. In the last subsection we show results for a numerical study with different diffusion coefficients. This study confirms that the developed methods are suitable for a whole class of problems.

10.2.1. Problem formulation: Sequential OED with 2D advection dominated diffusion advection PDE problem

We begin the sequential optimum experimental design with a first parameter estimation. The setting is basically the same as in the preceding Subsection 10.1.1. The true parameter values are again $p_1^* = -0.2$ and $p_2^* = 0.3$. We choose for the standard deviation of the measurement errors $\sigma_i = 0.1, i = 1, .., 8$ and get the perturbation

$$\varepsilon = (-0.123, -0.204, 0.006, -0.088, -0.033, 0.140, -0.017, -0.012).$$

We have eight measurements at the same points as in Figure 10.1.

The start values are p^0= $(-0.1, -0.1)$. We adaptively refine the grid with the error sum strategy from Section 8.1, starting on a grid with 36,864 DoFs we perform 3 refinement cycles. This procedure leads to a grid with approximately 130,000 DoFs.

For the OED run, we define 81 possible measurement points, which are equidistantly placed in the domain $\Omega = [-1, 1]^2$,

$$\begin{array}{llll} x_1^m = (-0.8, 0.8), & x_2^m = (-0.6, 0.8), & \ldots, & x_9^m = (0.8, 0.8), \\ x_{10}^m = (-0.8, 0.6), & x_{11}^m = (-0.6, 0.6), & \ldots, & x_{18}^m = (0.8, 0.6), \\ \vdots & \vdots & & \vdots \\ x_{73}^m = (-0.8, -0.8), & x_{74}^m = (-0.6, -0.8), & \ldots, & x_{81}^m = (0.8, -0.8). \end{array}$$

Thereof, we choose a maximum number of 8 points, which are selected by the optimization algorithm. We determine the optimum placement of the measurement points by optimizing the sampling decisions w_i. Every sampling decision w_i corresponds to one possible spatial measurement point $x_i^m, i = 1, .., 81$. Initially, we weight all possible points uniformly. Therefore we choose the start values for the sampling decisions as $w_i^0 = 0.098765, i = 1, .., 81$, because the sum of all sampling decisions equals the number of points to select: $\sum_i w_i = 8$. We scale all parameters to 1, see Remark 3.2.1. That means $\beta(p) = (\hat{p}_1 p_1, \hat{p}_2 p_2)$ and $p_1 = p_2 = 1$, $\hat{p}_1, \hat{p}_2$ are the estimated parameters from the preceding PE. As objective function we choose the A-criterion $\Phi_A(C) = \frac{1}{n_p} \operatorname{tr}(C(w))$, that means we minimize the average half-axis length of the confidence ellipsoid of the parameters.

As the last step of the sequential OED, we perform a second parameter estimation. Now we apply the optimized measurement point setting. We generate measurement values for the optimized measurement point setting. As before, we choose for the standard deviation of the measurement errors $\sigma_i = 0.1$ and get a new perturbation

$$\varepsilon = (0.073, 0.290, 0.195, -0.090, -0.006, 0.014, -0.072, 0.028).$$

As start value for the parameter estimation, we take the estimated value from the previous parameter estimation, $p_1^0 = \hat{p}_1$ and $p_2^0 = \hat{p}_1$.

10.2.2. Computational results for sequential OED

In this subsection, we show results for the diffusion coefficient $\alpha = 0.001$ as depicted in the preceding problem formulation. Figure 10.6 shows the error between estimated parameters $\hat{p}$ of the PE before OED and "true" parameters p^* for every iteration k of the Gauss-Newton algorithm. The parameter estimation convergences after 10 iterations. We choose the stopping criterion in equation (2.13) with $tol = 10^{-4}$. Table 10.5 depicts the estimated parameter values and the corresponding standard deviations. The goal of optimum experimental design is to reduce the standard deviations of the parameters to enhance the significance of the parameter estimates.

We scale all parameters to 1, see Remark 3.2.1. That means $\beta(p) = (-0.2164 p_1, 0.3014 p_2)$ and $p_1 = p_2 = 1$. The A-optimal design algorithm converges after 20 major iterations. The determined optimal values for the sampling decisions $\hat{w}_i$ sorted by size are

$$\begin{aligned}
&\hat{w}_4 = \hat{w}_5 = \hat{w}_8 = \hat{w}_{14} = \hat{w}_{60} = 1, \\
&\hat{w}_{10} = 0.8569, \hat{w}_6 = 0.7426, \hat{w}_9 = 0.7056, \hat{w}_{15} = 0.6949.
\end{aligned}$$

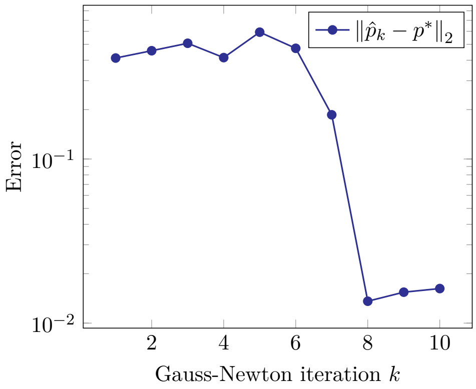


Figure 10.6.: Results for first parameter estimation in sequential optimum experimental design. Error in the Euclidian norm between estimated parameters $\hat{p}_k$ and "true" parameters p^* for every iteration k of the Gauss-Newton algorithm.

$\hat{p}$	standard deviation
-0.2164	$\pm 6.86\%$
0.3014	$\pm 2.40\%$

Table 10.5.: Estimated parameter values and corresponding standard deviations after the first parameter estimation, before optimum experimental design.

All remaining sampling decisions are $\hat{w}_i = 10^{-6}, i \notin \{4, 5, 6, 8, 9, 10, 14, 15, 60\}$, which is a lower bound to prevent numerical difficulties for $w_i = 0$. The optimization algorithm computes the derivative of $\sqrt{w_i}$, close to $w_i = 0$ this term becomes very large. To prevent numerical difficulties arising from this, we set a lower bound $w_i \geq 10^{-6}$.

We observe a fractional solution, not all sampling decisions are integers. Therefore we use a heuristic: the round up and off strategy. We round up the largest sampling decisions and round off the smallest ones, keeping the sum of all sampling decisions equal or below the maximum number, see Section 3.3. Thus the selected measurement points x_i^m are $i \in \{4, 5, 6, 8, 9, 10, 14, 60\}$.

Figure 10.7 shows the result of the OED. Three of the four measurement points with a fractional solution are selected. We notice that two of the measurement points with a fractional solution are neighbors. One of those neighbors is selected, the other one is not selected. That is a hint, that a point in between these two points is optimal, which is not part of our fixed grid.

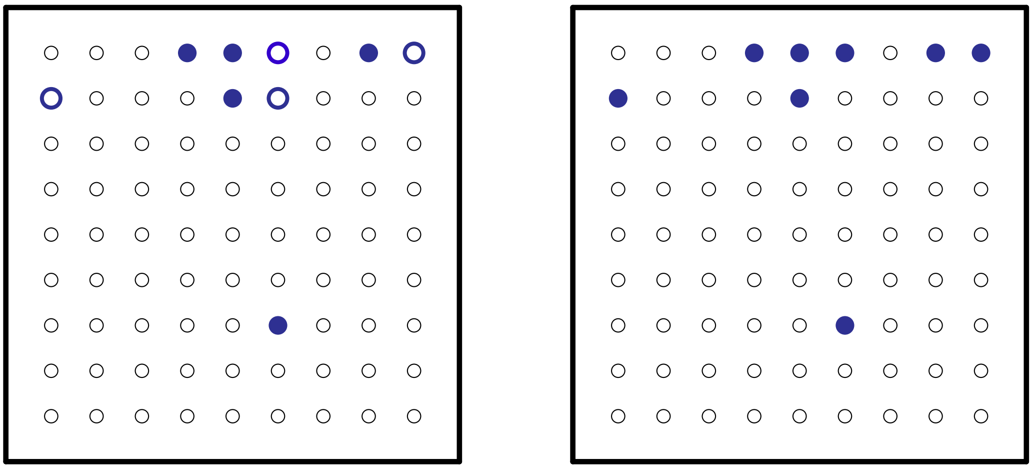


Figure 10.7.: Result of OED. The left picture shows the fractional solution with blue points for $w_i = 1$, blue circles for $w_i \in (1, 10^{-6})$ and black circles for $w_i = 10^{-6}$. The right picture shows the selected measurement points (blue points) and the not selected points (black circles).

With these optimized measurement points, we perform a second PE. Starting values for the parameters are the estimated $\hat{p}$ from the preceding PE, $p_1^0 = -0.2164$ and $p_2^0 = 0.3014$. The Gauss-Newton algorithm converges after 4 iterations with the same tolerance of $tol = 10^{-4}$. The estimated parameter values are $\hat{p}_1 = -0.1934$ and $\hat{p}_2 = 0.3011$. Again we examine the standard deviations of the parameters. Table 10.6 shows the estimated parameters and the corresponding standard deviations. We managed to reduce the standard deviation by 5.45 percentage points for the first parameter and by 0.66 percentage points for the second parameter. That means the uncertainty of the parameters is reduced by applying sequential OED.

Furthermore, Table 10.7 depicts the four OED criteria after the first PE (before OED) and after the second PE (after OED). All criteria are reduced. That means the linearized confidence regions of the parameters are reduced.

In summary, we solved the OED problem with an advection dominated diffusion advection PDE constraint by utilizing the developed sensitivity evaluation techniques, the differentiable upwind discontinuous Galerkin discretization from Chapter 6, the structure exploitation from Chapter 7 and the frozen adaptivity from Chapter 8.

before OED		after OED	
$\hat{p}$	std. dev.	$\hat{p}$	std. dev.
-0.2164	6.86%	-0.1934	1.41%
0.3014	2.40%	0.3011	1.74%

Table 10.6.: Estimated parameters and corresponding standard deviations of estimated parameters after the first PE (before OED) and after the second PE (after OED).

Criterion	A	D	E	M
before OED	$1.362 \cdot 10^{-4}$	$5.829 \cdot 10^{-5}$	$2.594 \cdot 10^{-4}$	$1.484 \cdot 10^{-2}$
after OED	$1.739 \cdot 10^{-5}$	$1.364 \cdot 10^{-5}$	$2.818 \cdot 10^{-5}$	$5.228 \cdot 10^{-3}$

Table 10.7.: Values of the OED criteria after the first PE (before OED) and after the second PE (after OED).

We reduced the linearized confidence regions of the parameters by selecting optimal measurement points. That way we increased the significance of the estimates considerably.

10.2.3. Comparison of diffusion coefficients

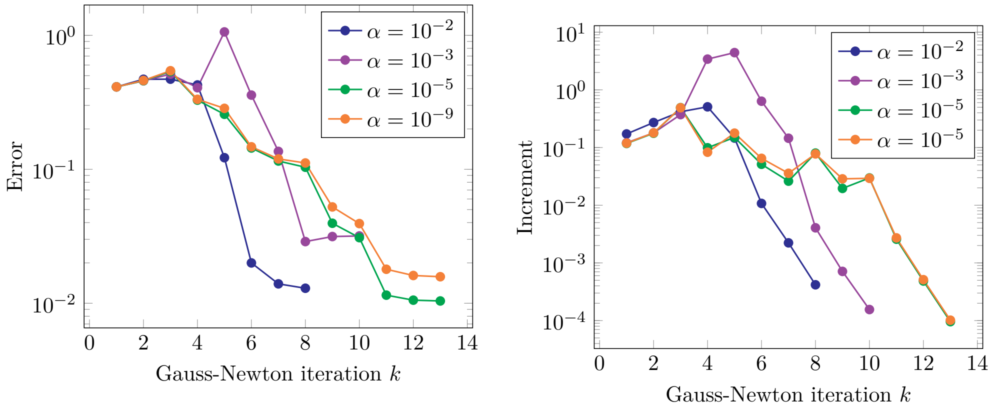
In this subsection, we test the algorithms with different diffusion coefficients: $\alpha = 10^{-2}, 10^{-3}, 10^{-5}, 10^{-9}$. That way we investigate if the developed methods are suitable not only for one specific setting but for a class of problems.

The setting is the same as in the preceding subsection, we perform sequential OED with a diffusion advection PDE model problem. We vary the diffusion coefficient α. The smaller the diffusion coefficient, the more advection dominated the PDE model becomes.

For the four choices of the diffusion coefficient α, we generate measurement data with 4% noise, that means a standard deviation of $\sigma_i = 0.2, i = 1,..,8$. We get four sets of measurement data, with four different perturbations:

$$\begin{aligned}
\alpha = 10^{-2}: &\quad \varepsilon = (-0.163, 0.014, -0.183, -0.158, -0.225, 0.208, -0.118, 0.184),\\
\alpha = 10^{-3}: &\quad \varepsilon = (-0.246, -0.407, 0.012, -0.175, -0.066, 0.281, -0.034, -0.024),\\
\alpha = 10^{-5}: &\quad \varepsilon = (-0.340, 0.031, 0.276, -0.001, 0.326, 0.025, 0.056, 0.166),\\
\alpha = 10^{-9}: &\quad \varepsilon = (0.075, -0.192, -0.201, -0.245, 0.045, 0.106, -0.122, -0.016).
\end{aligned}$$

Figure 10.8 shows the results for the PE before OED. For all four tested diffusion coefficients, the Gauss-Newton algorithm converges linearly. The increment value gets smaller than the stopping criterion in equation (2.13) with $tol = 10^{-4}$. The larger the diffusion coefficient, the less iterations are needed to reach the stopping criterion. That means the more advection dominated the PDE model gets, the more difficult the solution of the parameter estimation problem gets. That is what we expected. Nonetheless, all tested cases converge within 13 iterations.



(a) Error $\|\hat{p}_k - p^*\|_2$ in the Euclidian norm between estimated parameters $\hat{p}_k$ and ”true“ parameters p^* for every iteration k of the Gauss-Newton algorithm.

(b) Increment $\|\delta p_k\|_2$ for every Gauss-Newton iteration k is shown.

Figure 10.8.: Results of the parameter estimation with the Gauss-Newton algorithm for four different choices of the diffusion coefficient α.

The error between estimated and “true” parameters is reduced in all four cases, see Figure 10.8a. Regarding the size of the error between estimated and “true” parameters for the different diffusion coefficients, we do not see such a clear picture as for the number of iterations: for $\alpha = 10^{-3}$ we get the largest error for the four tested cases. That can be caused by the different perturbations of measurement errors. Taking the same perturbation or no measurement error could lead to a different result.

In the next step of sequential OED, we perform an OED run with the A-criterion. As before, out of 81 possible measurement points, the OED algorithm selects 8 measurement points via the sampling decisions $w_i, i = 1, .., 81$. For $\alpha = 10^{-2}$, the SQP algorithm converges after 20 major iterations. For $\alpha = 10^{-3}$, it takes 18 major iterations, for $\alpha = 10^{-5}$ it takes 16 major iterations, and for $\alpha = 10^{-9}$ it takes 18 major iterations. Thus all OED problems are solved within a similar number of iterations.

	$\alpha = 10^{-2}$		$\alpha = 10^{-3}$		$\alpha = 10^{-5}$		$\alpha = 10^{-9}$	
w_1	-		-		1	x	1	x
w_4	-		1	x	1	x	1	x
w_5	1	x	1	x	1	x	1	x
w_6	1	x	0.4125	x	0.2026		-	
w_7	1	x	-		-		-	
w_8	1	x	1	x	0.9490	x	1	x
w_9	0.6090	x	1	x	0.5060	x	0.2671	
w_{10}	0.9079	x	1	x	0.8708	x	1	x
w_{11}	-		-		0.8708	x	1	x
w_{14}	-		1	x	1	x	1	x
w_{15}	1	x	0.4070		0.2026		-	
w_{16}	0.5879		-		-		-	
w_{19}	0.8952	x	0.9732	x	-		-	
w_{24}	-		0.2073		0.2026		-	
w_{33}	-		-		0.1955		0.7328	x

Table 10.8.: Sampling decisions $w_i > 10^{-6}$ for different diffusion coefficients α. The symbol x marks selected measurement points.

Table 10.8 depicts the sampling decisions w_i which are greater 10^{-6}. Figure 10.9 shows the position of these optimized sampling decisions. We see, that the selected measurement points shift to the left as the diffusion coefficient gets smaller. As the solution of the PDE model gets more advection dominated, different measurement points are optimal.

For each diffusion case, we perform a second PE with the optimized measurement points from the preceding OED runs. Therefore, we round up the largest fractional weights until the number of eight measurement points is reached. In Table 10.8 the symbol x marks the selected measurement points.

The second PE converge after 3, 5, 3 and 4 iterations for $\alpha = 10^{-2}, 10^{-3}, 10^{-5}$ and 10^{-9}, respectively. Table 10.9 shows the value of the objective function Φ_A. We see that the objective function values are reduced for all four diffusion cases. Table 10.10 shows the standard deviations of the two parameters for the four different diffusion coefficients α. For all diffusion coefficients, the standard deviations of both parameters

α	Φ_A^0 before OED	Φ_A^1 after OED
10^{-2}	$8.55769 \cdot 10^{-4}$	$1.03212 \cdot 10^{-4}$
10^{-3}	$7.08039 \cdot 10^{-4}$	$4.51676 \cdot 10^{-5}$
10^{-5}	$3.51241 \cdot 10^{-4}$	$5.48265 \cdot 10^{-5}$
10^{-9}	$4.23966 \cdot 10^{-4}$	$6.19242 \cdot 10^{-5}$

Table 10.9.: Values of the OED criteria after the first PE (before OED) and after the second PE (after OED) for the four different choices of the diffusion coefficient α.

are reduced. Thus, they are estimated more significantly. Regarding the choice of the diffusion coefficient, we cannot see any patterns influencing the reduction of the standard deviations.

	std. dev. before OED		std. dev. after OED	
α	$\hat{p}_1$	$\hat{p}_2$	$\hat{p}_1$	$\hat{p}_2$
10^{-2}	17.14%	7.34%	4.34%	3.82%
10^{-3}	14.52%	5.57%	2.44%	2.78%
10^{-5}	11.77%	4.01%	2.58%	3.04%
10^{-9}	12.26%	4.31%	3.33%	2.97%

Table 10.10.: Standard deviations of estimated parameters after the first PE (std. dev. before OED) and after the second PE (std. dev. after OED) for the four different choices of the diffusion coefficient α.

We conclude that we successfully applied the developed methods and algorithms to a class of PDE problems. The methods work well for advection dominated diffusion advection reaction PDE model problems. For all four tested diffusion coefficients, the uncertainty of the parameters is reduced with the developed methods.

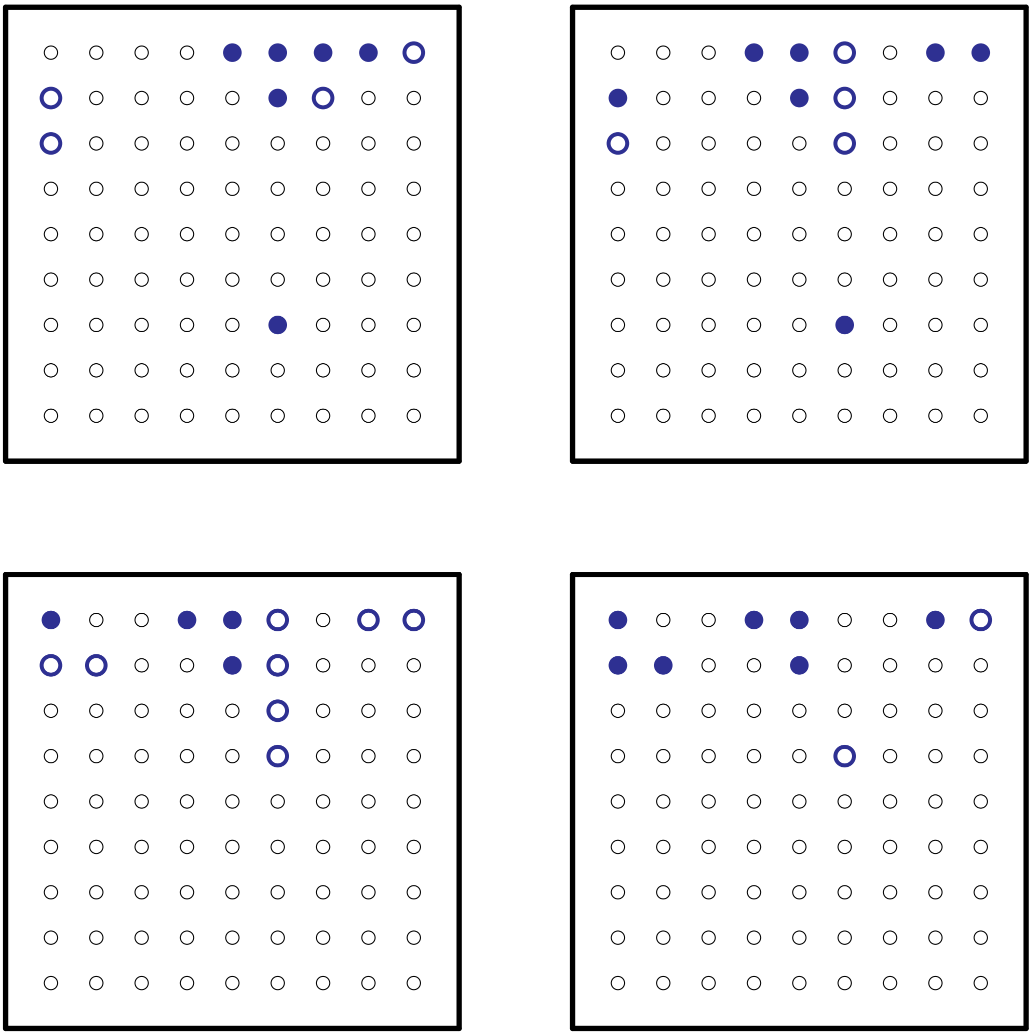


Figure 10.9.: Result of OED. The top left picture shows the solution for $\alpha = 10^{-2}$, the top right picture shows the solution for $\alpha = 10^{-3}$, the bottom left picture shows the solution for $\alpha = 10^{-5}$ and the bottom right picture shows the solution for $\alpha = 10^{-9}$. Blue points stand for $w_i = 1$, blue circles for $w_i \in (1, 10^{-6})$ and black circles for $w_i = 10^{-6}$.

11. Numerical study on mesh independence for PE and OED

In this chapter we study the mesh independence of the developed optimization methods. We first investigate a parameter estimation problem starting with the problem formulation. We compare the output of the PE algorithm for different refined grids of the underlying PDE simulation. In the next section we are concerned with an OED problem. We first introduce the problem formulation. We investigate if the developed methods are stable under grid refinement. Therefore, we compare the output of the OED algorithm for different refined grids of the underlying PDE simulation.

Throughout this chapter, we proceed in the same manner as in the preceding chapter: we generate measurement data by simulating the PDE model problem with the standard upwind discretization (4.5). We use for the optimization runs the developed differentiable discretization $\sigma_\mu(\beta, n)$ defined in equation (6.3) with $\mu = 0.1$. As accuracy of the iterative solver `GMRES` we choose 10^{-10} for all simulations, because of the considerations regarding the handling of the iterative solver in Section 8.2.

11.1. Parameter estimation

In this section we study mesh independence for parameter estimation with an advection dominated diffusion advection reaction PDE model problem. We solve this PE problem for different uniform grid refinements without measurement noise.

11.1.1. Problem formulation

We now investigate a different underlying PDE model problem: besides different numerical values for diffusion and advection factors, we include a reaction rate. The strong form reads

$$-0.0001\mathit{\Delta} y + \beta(p) \cdot \nabla y + 0.5y = 2 \quad \text{on } \Omega, \tag{11.1a}$$

$$y = 0.5 \quad \text{on } \mathit{\Gamma} = \partial\Omega. \tag{11.1b}$$

The domain Ω comprises the rectangle $[0,1] \times [1,2]$. The discrete PDE problem is: Find $y_h \in V_h$ such that

$$\alpha a_h(y_h, v_h) + b_h(p; y_h, v_h) + c_h(y_h, v_h) = f_h(v_h), \quad \forall v_h \in V_h,$$

with $a_h(y_h, v_h)$, $b_h(p; y_h, v_h)$, $c_h(y_h, v_h)$ and $f_h(v_h)$ defined in equation (7.10).

The discrete parameter estimation problem reads

$$\min_{p \in P} \frac{1}{2} \sum_{i=1}^{m} \left(\frac{\eta_i - h_{i,h}(p)}{\sigma_i} \right)^2 .$$

The components of the advection direction $\beta(p) = (p_1, p_2)^T$ are the two parameters to be estimated. Point measurements given by the value of the discrete PDE solution operator at the measurement points $h_{i,h}(p) := S_h(p)\Big|_{x=x_i^m}$, constitute the model response. We define eight measurement points with coordinates

$$\begin{aligned} x_1^m &= (0.2, 0.25), & x_2^m &= (0.4, 0.25), \\ x_3^m &= (0.6, 0.25), & x_4^m &= (0.8, 0.25), \\ x_5^m &= (0.2, 1.75), & x_6^m &= (0.4, 1.75), \\ x_7^m &= (0.6, 1.75), & x_8^m &= (0.8, 1.75). \end{aligned}$$

Figure 11.1 visualizes the placement of the measurement points in the domain $[0,1] \times [1,2]$.

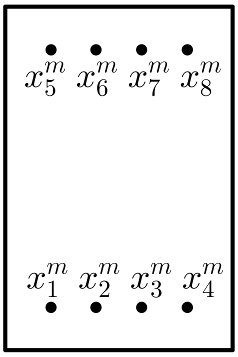

x_5^m x_6^m x_7^m x_8^m
x_1^m x_2^m x_3^m x_4^m

Figure 11.1.: Placement of the measurement points $x_i^m, i = 1, .., 8$, in the domain $\Omega = [0,1] \times [1,2]$.

For this study on mesh independence, we generate measurement data η_i without noise, that means $\varepsilon_i = 0, i = 1, .., 8$. The "true" parameter values are

$$p_1^* = 1, \quad p_2^* = 5.$$

We investigate two sets of start values for the parameters

$$\textbf{(a)} \quad p_1^0 = 3, \quad p_2^0 = 7, \quad \text{and} \quad \textbf{(b)} \quad p_1^0 = 6, \quad p_2^0 = 6.$$

For the sensitivity generation, the structure exploiting techniques depicted in Chapter 7 and the frozen adaptivity techniques from Chapter 8 are applied. Per Gauss-Newton iteration, we simulate the primal PDE problem and the two tangential PDE problems on one common grid, which is generated by the error sum strategy depicted in Section 8.1. As before, we choose for the Gauss-Newton algorithm as step size strategy the RMT globalization.

11.1.2. Comparison of grid refinements

We perform parameter estimation with the Gauss-Newton algorithm for different uniform grid refinements and the stopping criterion in equation (2.13) with $tol = 10^{-5}$. Figure 11.2 shows the error $\|\hat{p}_k - p^*\|_2$ in the Euclidian norm for every iteration of the Gauss-Newton iteration for a uniform refinement with 147,456 DoFs. We see the results for the two sets of start values **(a)** $p^0 = (3, 7)$ and **(b)** $p^0 = (6, 6)$. Compared to the example before in Section 10.1, the error decreases linearly in every iteration. Here, we do not have a sign change in the parameters, therefore, the error decreases from the first iteration. Furthermore, we did not add any noise to the measurements, that is why the error is much smaller in the final iteration. Comparing the two sets of start values, we see that start values **(a)**, which are closer to the "true" parameters, need 3 iterations less to converge than start values **(b)**, which are not as close to the "true" parameters.

Table 11.1 depicts the relative errors $\left\|\frac{\hat{p}-p^*}{p^*}\right\|_2$ in the final iteration and the number of iterations needed for five different refinements (number of DoFs) for the two sets of start values. For both sets of start values the relative error decreases, the finer the grid, the better the estimated parameter values. Considered separately, each set of start values needs the same number of iterations for all evaluated grid refinements. That gives strong evidence that the proposed methods and algorithms are mesh independent.

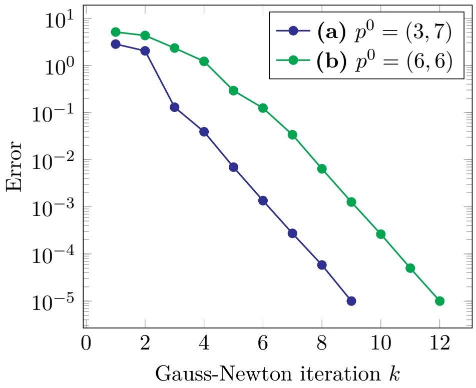


Figure 11.2.: Error $\|\hat{p}_k - p^*\|_2$ in the Euclidian norm between estimated parameters $\hat{p}_k$ and "true" parameters p^* for every iteration k of the Gauss-Newton algorithm for two sets of start values: **(a)** $p^0 = (3, 7)$ and **(b)** $p^0 = (6, 6)$.

	(a) $p^0 = (3, 7)$		**(b)** $p^0 = (6, 6)$	
# DoFs	rel error	# iter	rel error	# iter
576	0.003839896	9	0.003843293	12
2,304	0.001272682	9	0.001271076	12
9,216	0.000223607	9	0.000224018	12
36,864	0.000014000	9	0.000014000	12
147,456	0.000003000	9	0.000003000	12

Table 11.1.: Number of DoFs vs. relative error $\left\|\frac{\hat{p}-p^*}{p^*}\right\|_2$ and number of iterations for two sets of start values: **(a)** $p^0 = (3, 7)$ and **(b)** $p^0 = (6, 6)$.

Comparing the two sets of start values, we see a difference between the relative errors only in the sixth decimal place. On the two finest refinements the relative errors are identical. For all grid refinements set **(a)** needs 3 iterations less than set **(b)**. This can be explained by the fact, that set **(a)** is closer to the “true” parameter values than set **(b)**.

Figure 11.3 depicts the relative error vs. the number of DoFs for the two sets of start values **(a)** and **(b)**, corresponding to Table 11.1. Again, we see that for a finer grid refinement, the relative error decreases rapidly. On a finer grid, the estimation is more accurate. That gives strong evidence that the developed methods are stable under grid refinement.

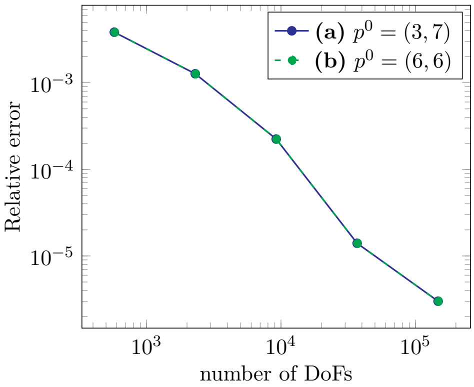


Figure 11.3.: Relative error $\left\|\frac{\hat{p}-p^*}{p^*}\right\|_2$ in the Euclidian norm between estimated parameters $\hat{p}$ and ”true“ parameters p^* vs. number of DoFs for uniform mesh refinement for two different sets of start values: **(a)** $p^0 = (3,7)$ (blue) and **(b)** $p^0 = (6,6)$ (green). Note, that the curves overlap each other.

11.2. Optimum experimental design

Let us now investigate mesh independence for OED. We study the same underlying PDE model problem as in the previous section for PE. We solve the OED problem with the same start values for different spatial uniform grid refinements.

11.2.1. Problem formulation

The setting is basically the same as in the preceding Section 11.1. The underlying PDE model is an advection dominated diffusion advection reaction PDE model problem, equation (11.1).

We define 135 possible measurement points $x_i^m, i = 1, .., 135$. The measurement points are equidistantly placed in the interior of the domain $\Omega = [0,1] \times [1,2]$:

$$\begin{array}{lll} x_1^m = (0.1, 0.125), & x_{10}^m = (0.1, 0.25), \quad \ldots, & x_{127}^m = (0.1, 1.875), \\ x_2^m = (0.2, 0.125), & x_{11}^m = (0.2, 0.25), \quad \ldots, & x_{128}^m = (0.2, 1.875), \\ \quad\vdots & \quad\vdots & \quad\vdots \\ x_9^m = (0.9, 0.125), & x_{18}^m = (0.9, 0.25), \quad \ldots, & x_{135}^m = (0.9, 1.875). \end{array}$$

The OED algorithm selects 8 of these points. The optimization variables are the sampling decisions $w_i, i = 1, .., 135$, each possible measurement point corresponds to one sampling decision w_i. As starting point for the optimization we weight all possible points uniformly and choose as start values for the sampling decisions $w_i^0 = 0.05926, i = 1, .., 135$. The sum of all sampling decisions equals the number of points to select: $\sum_i w_i = 8$.

We scale all parameters to 1, such that a difference in magnitude does not lead to a unilateral preference of a parameter in the OED, see Remark 3.2.3. That means we choose $\beta(p) = (1.5p_1, 4.6p_2)$ and $p_1 = p_2 = 1$. As objective function we choose the D-criterion $\Phi_D(C) = \det(C)^{\frac{1}{n_p}}$, that means we minimize the volume of the confidence ellipsoid of the parameters.

11.2.2. Comparison of grid refinements

We perform D-optimal design for five different uniform grid refinements. Table 11.2 reports the objective function value $\Phi_D(C)$ before and after the OED run and the standard deviations of p_1 and p_2 (std. dev. p_1, std. dev. p_2) before and after the OED run for the five different refinements (# DoFs).

	$\Phi_D(C)$		std. dev. p_1		std. dev. p_2	
# DoFs	before	after	before	after	before	after
576	2.21705	0.507429	222.09%	104.46%	124.15%	62.86%
2,304	2.15159	0.506812	216.49%	103.91%	122.14%	64.77%
9,216	2.10972	0.49649	212.69%	103.13%	121.15%	66.36%
36,864	2.07964	0.481361	209.95%	100.75%	120.51%	66.87%
147,456	2.0708	0.479062	209.02%	99.15%	120.42%	66.45%

Table 11.2.: Objective function $\Phi_D(C)$ and standard deviation (std. dev.) of parameters p_1 and p_2 before and after the OED runs for five different grid refinements (# DoFs).

We see, that for all five grid refinements, the objective value and the standard deviations of both parameters are reduced substantially. Nevertheless, the standard deviation of p_1 is even after the OED runs large, the parameter is still undetermined. Perhaps in this setting more than 8 measurements are needed to reliably estimate p_1.

Comparing the different refinements for the objective value we see that the finer the grid, the smaller the objective value. Even though the difference in the objective value between the coarsest and finest grid is small. Comparing the standard deviations of p_1 and p_2 for the different refinements, we see in parameter p_1 a decrease for finer grids, whereas we see in parameter p_2 an increase of the standard deviation for finer grids. That is because of the choice of the objective D-criterion. In the objective, both parameters are considered and overall, the standard deviations are reduced.

In Table 11.3, all sampling decisions w_i with $w_i > 10^{-6}$ are depicted for the five different mesh refinements. Except for the two coarsest grids, the same measurement points are determined to be important: $w_{113}, w_{114}, w_{115}, w_{122}, w_{123}, w_{124}, w_{130}, w_{131}, w_{132}$ and w_{133} are greater than 10^{-6}. In Table 11.3, the symbol x marks selected sampling decisions after rounding. We see, that although the fractional solution differs, in the rounded solution, exactly the same 8 measurement points are chosen on the three finest grids. Thus, on a sufficiently fine grid, the algorithm converges to the same optimal solution.

	576		2,304		9,216		36,864		147,456	
w_{103}	0.502395		-		-		-		-	
w_{105}	0.962958	x	-		-		-		-	
w_{112}	0.707485	x	-		-		-		-	
w_{113}	-		1	x	1	x	1	x	0.806951	x
w_{114}	-		1	x	0.180861		0.238939		0.232441	
w_{115}	1	x	0.803007	x	0.429423		0.227388		0.23245	
w_{122}	1	x	1	x	1	x	0.875151	x	0.866228	x
w_{123}	-		-		1	x	1	x	1	x
w_{124}	1	x	1	x	1	x	1	x	1	x
w_{130}	0.827037	x	0.856859	x	0.744777	x	0.688066	x	0.880984	x
w_{131}	1	x	1	x	1	x	0.970331	x	0.98082	x
w_{132}	-		0.340009		0.644813	x	1	x	1	x
w_{133}	1	x	1	x	1	x	1	x	1	x

Table 11.3.: Sampling decisions $w_i > 10^{-6}$ for different numbers of DoFs. The symbol x marks selected measurement points.

Figure 11.4 visualizes the solutions for the five different grid refinements. For all grid refinements, the chosen measurement points are in the upper central area of the domain. That means, the developed algorithms compute similar solutions for different grid refinements, which is a desirable property. The finer the grid, we see convergence to a set of measurement points. As we have seen before in the Table 11.3, even though the fractional sampling decisions are changing from finer grid to finer grid, the rounded solution chooses the same measurement points (right bottom picture). For a sufficiently fine grid, the solutions converge and the same measurement points are chosen. That gives strong evidence that the developed methods and algorithms are stable under grid refinement.

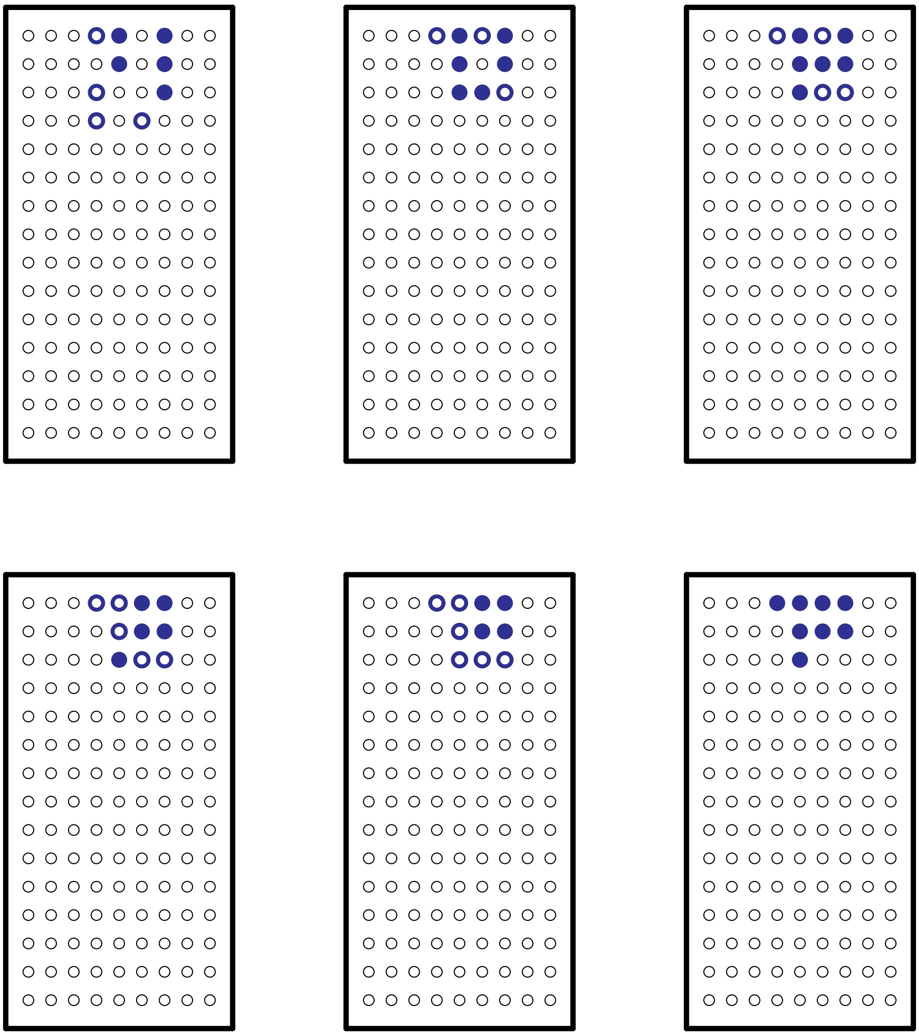


Figure 11.4.: Top (from left to right): 576 DoFs, $2,304$ DoFs, $9,216$ DoFs, Bottom: $36,864$ DoFs, $147,456$ DoFs and rounded weights for $147,456$ DoFs. Blue dots represent a selected measurement point $w_i = 1$, blue circles represent a measurement point with fractional sampling decision $10^{-6} < w_i < 1$ and black circles represent a not selected measurement point $w_i = 10^{-6}$.

12. Conclusion

In this thesis we successfully solved optimum experimental design (OED) problems for parameter estimation (PE) with PDE models. We developed efficient and accurate methods for sensitivity generation.

We proposed and analyzed a differentiable upwind discontinuous Galerkin discretization. We performed a rigorous convergence analysis for the differentiable discretization and finally arrived at an error estimate in the energy norm and a superconvergence result. We showed that the analysis also holds for a discretization of a diffusion advection reaction PDE model. We showed that the assumptions for the convergence analysis hold for a non normalized advection coefficient of the differentiable stabilization. In accordance with the standard upwind discretization for a non normalized advection coefficient [11], the convergence behavior changes, the non normalized advection coefficient influences the estimation constant. We showed that this behavior also holds for the differentiable stabilization. Numerical tests confirm the predicted behavior.

Furthermore, we developed methods for structure exploitation of the primal and tangential discretization schemes. We exploited the problem structure and reused the left hand side of the primal discretization when generating the tangential discretizations. Moreover, we exploited the structure of the discontinuous Galerkin finite element method. We differentiated only the core parts of the discretization by AD to efficiently generate the tangential discretizations. Numerical examples confirm the efficiency of the structure exploiting method. No memory issues appear while generating the tangential problems.

We froze all adaptive components to generate the consistent sensitivities. We developed a heuristic, the error sum strategy for grid refinement, to generate one common spatial grid, which is suitable for primal and tangential problems. Furthermore, we investigated the influence of the adaptive step number of the iterative solver. We concluded, that the two step approach with an accurate stopping criterion for the iterative solver is preferable in our setting and leads to consistent sensitivities. We demonstrated the developed methods for frozen adaptivity by numerical examples.

We implemented the developed methods in the new software `SeafaND-Optimizer`, short for *structure exploiting and frozen adaptivity numerical differentiation optimizer*. It is a software for efficient simulation, parameter estimation and optimum experimental design with PDE models. The core part of the software is a consistent sensitivity generation with the aforementioned methods. Furthermore, the `SeafaND-Optimizer` provides functionalities of `dealii` and `Amandus` for PDE simulation and of `VPLAN`, `PAREMERA` and `SNOPT` for optimization. This leads to an accurate and fast solution of the PE and OED optimization problems.

Numerical case studies for PE and OED problems with advection dominated 2D diffusion advection PDE models demonstrated the efficiency and accuracy of the methods. Each PDE simulation, one primal PDE problem and two tangential PDE problems, had approximately 131,500 degrees of freedom. With the developed methods for sensitivity generation, we efficiently and automatically generated consistent sensitivities. We tested the PE algorithm with three different noise levels. The behavior was as expected: the higher the noise level, the less accurate are the parameter estimates. We successfully performed a case study with different diffusion coefficients. Thus, the developed methods are suitable for a class of problems. We performed a numerical study on mesh independence for PE and OED problems. We concluded, that the study gives strong evidence that the developed algorithms are stable under mesh refinements.

Directions for future research

Let us finally state some promising directions for future research, which came up during the work on this thesis.

In this thesis we were concerned with the sampling design problem. For more complicated OED problems, where the controls directly enter the PDE model, we need second order sensitivities. The developed methods for structure exploitation of the discretization schemes and for frozen adaptivity lay a solid groundwork and can be extended to second order sensitivities. This requires generating and solving of second order sensitivity or adjoint PDEs.

Another interesting question for future research is the development of tailored error estimators for adaptive grid refinement for discontinuous Galerkin methods for PE and OED. The error estimators should incorporate the discretization errors of the sensitivities. The goal of the error estimators should be the accurate simulation of all PDE problems, the primal problem as well as the sensitivity problems.

List of Figures

List of Tables

Bibliography

[1] R. A. Adams. *Sobolev spaces*. Academic Press, 1975. 26

[2] J. Ahrens, B. Geveci, and C. Law. *Visualization handbook*, chapter 36 - ParaView: An end-user tool for large-data visualization, pages 717–731. Elsevier, 2005. 148

[3] S. Akindeinde and D. Wachsmuth. A-posteriori verification of optimality conditions for control problems with finite-dimensional control space. *Numerical Functional Analysis and Optimization*, 33(5):473–523, 2012. 15, 33, 63

[4] J. Albersmeyer. *Adjoint-based algorithms and numerical methods for sensitivity generation and optimization of large scale dynamic systems*. Dissertation, Heidelberg University, 2010. 17, 68

[5] J. Albersmeyer and H. G. Bock. Efficient sensitivity generation for large scale dynamic systems. Technical report, Deutsche Forschungsgemeinschaft, 2009. Priority program 1253 optimization with partial differential equations. 17, 68

[6] A. E. Altmann-Dieses, J. P. Schlöder, H. G. Bock, and O. Richter. Optimal experimental design for parameter estimation in column outflow experiments. *Water Resources Research*, 38(10):1186, 2002. 14

[7] A. E. Altmann-Dieses [A. Dieses]. *Numerical methods for optimization problems in water flow and reactive solute transport processes of xenobiotics in soils*. Dissertation, Heidelberg University, 2000. 14

[8] D. Arndt, W. Bangerth, D. Davydov, T. Heister, L. Heltai, M. Kronbichler, M. Maier, J.-P. Pelteret, B. Turcksin, and D. Wells. The `deal.II` library, version 8.5. *Journal of Numerical Mathematics*, 25(3):137–146, 2017. 144, 146, 148

[9] D. N. Arnold. An interior penalty finite element method with discontinuous elements. *SIAM Journal on Numerical Analysis*, 19(4):742–760, 1982. 15, 56, 100

[10] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, 2002. 15, 56, 100

[11] B. Ayuso and L. D. Marini. Discontinuous Galerkin methods for advection-diffusion-reaction problems. *SIAM Journal on Numerical Analysis*, 47(2):1391–1420, 2009. 99, 108, 109, 178

[12] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II – a general-purpose object-oriented finite element library. *ACM Transactions on Mathematical Software*, 33(4):24/1–24/27, 2007. 144, 146, 148

[13] Y. Bard. *Nonlinear parameter estimation.* Academic Press, 1974. 36, 38, 39, 40

[14] D. M. Bates and D. G. Watts. *Nonlinear regression analysis and its applications.* Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley, 1988. 36

[15] I. Bauer. *Numerische Verfahren zur Lösung von Anfangswertaufgaben und zur Generierung von ersten und zweiten Ableitungen mit Anwendungen bei Optimierungsaufgaben in Chemie und Verfahrenstechnik.* Dissertation, Heidelberg University, 1999. 17, 46, 47, 49, 68, 137

[16] I. Bauer, H. G. Bock, S. Körkel, and J. P. Schlöder. Numerical methods for optimum experimental design in DAE systems. *Journal of Computational and Applied Mathematics*, 120(1–2):1–25, 2000. 14, 48, 49

[17] R. Becker, M. Braack, and B. Vexler. Parameter identification for chemical models in combustion problems. *Applied Numerical Mathematics*, 54(3–4):519–536, 2005. 15, 16, 68, 69, 134

[18] R. Becker, D. Meidner, and B. Vexler. Efficient numerical solution of parabolic optimization problems by finite element methods. *Optimization Methods and Software*, 22(5):813–833, 2007. 15, 16, 68

[19] R. Becker, D. Meidner, B. Vexler, and K. Pieper. Rodobo: A c++ library for optimization with stationary and nonstationary pdes. http://www.rodobo.org. 20

[20] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numerica*, 10:1–102, 2001. 140

[21] R. Becker and B. Vexler. A posteriori error estimation for finite element discretization of parameter identification problems. *Numerische Mathematik*, 96(3):435–459, 2004. 134

[22] R. Becker and B. Vexler. Mesh refinement and numerical sensitivity analysis for parameter calibration of partial differential equations. *Journal of Computational Physics*, 206(1):95–110, 2005. 15, 16, 46, 68, 134

[23] R. Becker and B. Vexler. Optimal control of the convection-diffusion equation

using stabilized finite element methods. *Numerische Mathematik*, 106(3):349–367, 2007. 68

[24] C. H. Bischof and H. M. Bücker. Computing derivatives of computer programs. In J. Grotendorst, editor, *Modern methods and algorithms of quantum chemistry: proceedings, second edition*, volume 3 of *NIC Series*, pages 315–327. John von Neumann Institute for Computing, Jülich, 2000. 72

[25] H. G. Bock. Numerical treatment of inverse problems in chemical reaction kinetics. In K. H. Ebert, P. Deuflhard, and W. Jäger, editors, *Modelling of chemical reaction systems*, volume 18 of *Springer Series in Chemical Physics*, pages 102–125. Springer, Heidelberg, 1981. 17, 35, 68, 146

[26] H. G. Bock. Recent advances in parameter identification techniques for ODE. In P. Deuflhard and E. Hairer, editors, *Numerical treatment of inverse problems in differential and integral equations*, pages 95–121. Birkhäuser, Boston, 1983. 17, 68

[27] H. G. Bock. *Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen*, volume 183 of *Bonner Mathematische Schriften*. Universität Bonn, 1987. 17, 34, 35, 36, 68

[28] H. G. Bock, S. Körkel, E. Kostina, and J. P. Schlöder. Robustness aspects in parameter estimation, optimal design of experiments and optimal control. In W. Jäger, R. Rannacher, and J. Warnatz, editors, *Reactive flows, diffusion and transport. From experiments via mathematical modeling to numerical simulation and optimization: final report of SFB (Collaborative Research Center) 359*, pages 117–146. Springer, 2007. 50

[29] H. G. Bock, E. Kostina, and J. P. Schlöder. On the role of natural level functions to achieve global convergence for damped Newton methods. In M. J. D. Powell and S. Scholtes, editors, *System modelling and optimization. Methods, theory and applications*, pages 51–74. Kluwer, 2000. 152

[30] M. Braack. *An adaptive finite element method for reactive-flow problems*. Dissertation, Heidelberg University, 1998. 138

[31] M. Braack and R. Rannacher. Adaptive finite element methods for low-Mach-number flows with chemical reactions. In H. Deconinck, editor, *30th computational fluid dynamics*, volume 1999-03 of *Lecture Series*. Karman Institute of Fluid Dynamics, Brussels, 1999. 138

[32] F. Brezzi, L. D. Marini, and E. Süli. Discontinuous Galerkin methods for first-order hyperbolic problems. *Mathematical Models and Methods in Applied Sciences*, 14(12):1893–1903, 2004. 59

[33] T. Carraro. *Parameter estimation and optimal experimental design in flow reactors.* Dissertation, Heidelberg University, 2005. 15, 39, 134, 138

[34] T. Carraro, V. Heuveline, and R. Rannacher. Determination of kinetic parameters in laminar flow reactors. I. Theoretical aspects. In W. Jäger, R. Rannacher, and J. Warnatz, editors, *Reactive flows, diffusion and transport. From experiments via mathematical modeling to numerical simulation and optimization: final report of SFB (Collaborative Research Center) 359*, pages 211–249. Springer, Berlin, 2007. 15, 16, 68, 134

[35] P. G. Ciarlet. *The finite element method for elliptic problems.* North-Holland, 1978. 26, 89

[36] F. Courty, A. Dervieux, B. Koobus, and L. Hascoët. Reverse automatic differentiation for optimum design: from adjoint state assembly to gradient computation. Technical Report RR-4363, INRIA, 2002. 16, 17, 68

[37] J. P. de S. R. Gago, D. W. Kelly, O. C. Zienkiewicz, and I. Babuška. A posteriori error analysis and adaptive processes in the finite element method: Part II – Adaptive mesh refinement. *International Journal for Numerical Methods in Engineering*, 19(11):1621–1656, 1983. 139

[38] A. Ern and J.-L. Guermond. *Theory and practice of finite elements*, volume 159 of *Applied Mathematical Sciences.* Springer, New York, 2010. 29, 56, 80

[39] T. Etling and R. Herzog. Optimum experimental design by shape optimization of specimens in linear elasticity. *SIAM Journal on Applied Mathematics*, 78(3):1553–1576, 2018. 15

[40] P. E. Farrell, D. A. Ham, S. W. Funke, and M. E. Rognes. Automated derivation of the adjoint of high-level transient finite element programs. *SIAM Journal on Scientific Computing*, 35(4):C369–C393, 2013. 20

[41] S. W. Funke and P. E. Farrell. A framework for automated PDE-constrained optimisation. *CoRR*, abs/1302.3894, 2013. (submitted). 20

[42] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review*, 47(1):99–131, 2005. 54, 146

[43] P. E. Gill, W. Murray, M. A. Saunders, and E. Wong. User's guide for SNOPT 7.6: Software for large-scale nonlinear programming. Center for Computational Mathematics Report CCoM 17-1, Department of Mathematics, University of California, San Diego, La Jolla, CA, 2017. 54, 146

[44] C. Goll, T. Wick, and W. Wollner. `DOpElib`: Differential equations and optimization environment; A goal oriented software library for solving PDEs and optimization problems with PDEs. *Archive of Numerical Software*, 5(2):1–14, 2017. 20

[45] J. Gopalakrishnan and G. Kanschat. A multilevel discontinuous Galerkin method. *Numerische Mathematik*, 95(3):527–550, 2003. 80, 86, 89, 91, 99, 100, 103, 105

[46] A. Griewank and A. Walther. *Evaluating derivatives, principles and techniques of algorithmic differentiation.* SIAM, Philadelphia, 2008. 16, 68, 71, 72, 137, 138

[47] C. Großmann and H.-G. Roos. *Numerik partieller Differentialgleichungen.* Teubner Studienbücher Mathematik. Teubner, Stuttgart, second edition, 1994. 26, 63

[48] L. Hascoët and V. Pascual. The Tapenade automatic differentiation tool: Principles, model, and specification. *ACM Transactions on Mathematical Software*, 39(3):1–43, 2013. 68, 146

[49] L. Hascoët, M. Vázquez, and A. Dervieux. Automatic differentiation for optimum design, applied to sonic boom reduction. In V. Kumar, M. L. Gavrilova, C. J. K. Tan, and P. L'Ecuyer, editors, *Computational science and its applications – ICCSA 2003*, volume 2668 of *Lecture Notes in Computer Science*, pages 85–94. Springer, Berlin, 2003. 16, 17

[50] R. Herzog and F. Ospald. Parameter identification for short fiber-reinforced plastics using optimal experimental design. *International Journal for Numerical Methods in Engineering (NME)*, 110(8):703–725, 2017. 15

[51] R. Herzog and I. Riedel. Sequentially optimal sensor placement in thermoelastic models for real time applications. *Optimization and Engineering*, 16(4):737–766, 2015. 49

[52] R. Herzog [R. Griesse] and B. Vexler. Numerical sensitivity analysis for the quantity of interest in PDE-constrained optimization. *SIAM Journal on Scientific Computing*, 29(1):22–48, 2007. 15, 16

[53] H. Hesse. *Multiple shooting and mesh adaptation for PDE constrained optimization problems.* Dissertation, Heidelberg University, 2008. 134

[54] M. Hinze, R. Pinnau, M. Ulbrich, and S. Ulbrich. *Optimization with PDE constraints*, volume 23 of *Mathematical modelling: theory and applications.* Springer, 2009. 26, 32, 33, 63, 69

[55] D. Holfeld, P. Stumm, and A. Walther. Structure exploiting adjoints for finite element discretizations. In G. Leugering, S. Engell, A. Griewank, M. Hinze, R. Rannacher, V. Schulz, M. Ulbrich, and S. Ulbrich, editors, *Constrained optimization and optimal control for partial differential equations*, volume 160 of *International series of numerical mathematics*, pages 183–196. Springer, Basel, 2012. 16, 17, 68

[56] D. Janka. *Sequential quadratic programming with indefinite Hessian approximations for nonlinear optimum experimental design for parameter estimation in differential–algebraic equations.* Dissertation, Heidelberg University, 2015. 14, 42, 49, 50, 54, 66, 148

[57] D. Janka, S. Körkel, and H. G. Bock. Direct multiple shooting for nonlinear optimum experimental design. In T. Carraro, M. Geiger, S. Körkel, and R. Rannacher, editors, *Multiple shooting and time domain decomposition methods*, volume 9 of *Contributions in mathematical and computational sciences*, pages 115–141. Springer, Cham, 2015. 50

[58] C. Johnson. *Numerical solution of partial differential equations by the finite element method.* Dover Publications, Mineola, New York, 2009. 15, 29, 56, 57, 58, 77, 80

[59] C. Johnson and J. Pitkäranta. An analysis of the discontinuous Galerkin method for a scalar hyperbolic equation. *Mathematics of Computation*, 46(173):1–26, 1986. 18, 80

[60] G. Kanschat. *Discontinuous Galerkin methods for viscous incompressible flow.* Advances in Numerical Mathematics. Teubner Research, first edition, 2007. 80, 86, 89, 91, 99, 104

[61] G. Kanschat, P. Esser, N. Sharma, D. Arndt, A. Bettendorf, J. Gedicke, D. Jando, A. Khan, P. Lucero, S. Meggendorfer, P. Siehr, N. Shakir, and M. Pesarin. Amandus: Simulations based on multilevel Schwarz methods. `https://bitbucket.org/guidokanschat/amandus`, 2017. Accessed: 2017-06-09. 144, 146

[62] D. W. Kelly, J. P. de S. R. Gago, O. C. Zienkiewicz, and I. Babuška. A posteriori error analysis and adaptive processes in the finite element method: Part I - error analysis. *International Journal for Numerical Methods in Engineering*, 19:1593–1619, 1983. 139

[63] R. Kircheis. *Structure exploiting parameter estimation and optimum experimental design methods and applications in microbial enhanced oil recovery.* Dissertation, Heidelberg University, 2015. 15, 17, 35, 42, 50, 146, 148, 152

[64] R. Kircheis and S. Körkel. Parameter estimation for DAE models in a multiple experiment context. In *Proceedings in applied mathematics and mechanics (PAMM)*, volume 11, pages 715–716, 2011. 50

[65] R. Kircheis and S. Körkel. Parameter estimation for high-dimensional PDE models using a reduced approach. In T. Carraro, M. Geiger, S. Körkel, and R. Rannacher, editors, *Multiple shooting and time domain decomposition methods*, volume 9 of *Contributions in Mathematical and Computational Sciences*, pages 143–157. Springer, Cham, 2015. 50

[66] S. Körkel. *Numerische Methoden für Optimale Versuchsplanungsprobleme bei nichtlinearen DAE-Modellen.* Dissertation, Heidelberg University, 2002. 14, 36, 42, 46, 47, 48, 49, 50, 66, 67, 145, 148

[67] S. Körkel, H. Arellano-Garcia, J. Schöneberger, and G. Wozny. Optimum experimental design for key performance indicators. In B. Braunschweig and X. Joulia, editors, *Proceedings of 18th european symposium on computer aided process engineering ESCAPE 18*, volume 25, 2008. 46

[68] S. Körkel, I. Bauer, H. G. Bock, and J. P. Schlöder. A sequential approach for nonlinear optimum experimental design in DAE systems. In F. Keil, W. Mackens, H. Voß, and J. Werther, editors, *Scientific computing in chemical engineering II, Simulation, image processing, optimization, and control*, pages 338–345. Springer, 1999. 49

[69] S. Körkel, E. Kostina, H. G. Bock, and J. P. Schlöder. Numerical methods for optimal control problems in design of robust optimal experiments for nonlinear dynamic processes. *Optimization Methods and Software*, 19(3–4):327–338, 2004. 50

[70] G. Kriwet. *Methods for model calibration and design of optimal experiments for partial differential equation models.* Dissertation, Philipps Universität Marburg, 2016. 15, 17

[71] M. Kronbichler and K. Kormann. A generic interface for parallel cell-based finite element operator application. *Computers & Fluids*, 63:135–147, 2012. 117

[72] H. C. La. *Dual control for nonlinear model predictive control.* Dissertation, Heidelberg University, 2016. 46

[73] H. C. La, A. Potschka, J. P. Schlöder, and H. G. Bock. Dual control and online optimal experimental design. *SIAM Journal on Scientific Computing*, 39(4):B640–B657, 2017. 46

[74] P. Lesaint and P. A. Raviart. On a finite element method for solving the neutron transport equation. In *Publications mathématiques et informatique de Rennes*, number S4, pages 1–40, 1974. 18, 99

[75] T. Lohmann, H. G. Bock, and J. P. Schlöder. Numerical methods for parameter estimation and optimal experiment design in chemical reaction systems. *Industrial and Engineering Chemistry Research*, 31(1):54–57, 1992. 14, 35, 45, 49

[76] I. Neitzel, K. Pieper, B. Vexler, and D. Walter. A sparse control approach to optimal sensor placement in PDE-constrained parameter estimation problems. *Numerische Mathematik*, 2019. 15, 49

[77] J. Nocedal and S. J. Wright. *Numerical optimization.* Springer series in operations research and financial engineering. Springer, second edition, 2006. 34, 35, 50, 51, 52, 54, 71

[78] S. A. Orszag. Spectral methods for problems in complex geometries. *Journal of Computational Physics*, 37(1):70–92, 1980. 117

[79] F. Ospald and R. Herzog. Optimal experimental design to identify the average stress-strain response in short fiber-reinforced plastics. In V. Bach and H. Fassbender, editors, *Special issue: Joint 87th annual meeting of the international association of applied mathematics and mechanics (GAMM) and Deutsche Mathematiker-Vereinigung (DMV), Braunschweig 2016*, volume 16 of *Proceedings in applied mathematics and mechanics (PAMM)*, pages 673–674, 2016. 15

[80] A. Potschka. *A direct method for the numerical solution of optimization problems with time-periodic PDE constraints.* Dissertation, Heidelberg University, 2011. 68

[81] F. Pukelsheim. *Optimal design of experiments.* Wiley series in probability and mathematical statistics. Wiley, New York, 1993. 45, 46

[82] A. Quarteroni and A. Valli. *Numerical approximation of partial differential*

equations, volume 23 of *Springer series in computational mathematics.* Springer, Berlin, 2008. 28, 29

[83] R. Rannacher and B. Vexler. Adaptive finite element discretization in PDE-based optimization. *GAMM-Mitteilungen*, 33(2):177–193, 2010. 15, 68, 132, 134

[84] W. Reed and T. Hill. Triangular mesh methods for the neutron transport equation. Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, Los Alamos, New Mexico, 1973. 15, 18, 56

[85] B. Rivière. *Discontinuous Galerkin methods for solving elliptic and parabolic equations, theory and implementation.* Frontiers in applied Mathematics. SIAM, Philadelphia, 2008. 28, 58, 63

[86] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7(3):856–869, 1986. 136

[87] S. Sager. Sampling decisions in optimum experimental design in the light of Pontryagin's maximum principle. *SIAM Journal on Control and Optimization*, 51(4):3181–3207, 2013. 46, 49

[88] J. P. Schlöder. *Numerische Methoden zur Behandlung hochdimensionaler Aufgaben der Parameteridentifizierung*, volume 187 of *Bonner Mathematische Schriften.* Universität Bonn, 1988. 35, 50, 146

[89] A. Schmidt. *Direct methods for PDE-constrained optimization using derivative-extended POD reduced-order models.* Dissertation, Heidelberg University, 2014. 17, 26, 33, 35, 63, 68

[90] P. Stumm. *Strukturausnutzende Adjungiertenberechnungen für die nichtlineare Optimierung.* Dissertation, TU Dresden, 2011. 16, 17

[91] P. Stumm, A. Walther, J. Riehme, and U. Naumann. Structure-exploiting automatic differentiation of finite element discretizations. In C. H. Bischof, H. M. Bücker, P. Hovland, U. Naumann, and J. Utke, editors, *Advances in automatic differentiation*, volume 64 of *Lecture Notes in Computational Science and Engineering*, pages 339–349. Springer, Berlin, 2008. 16, 18, 19, 68

[92] K. Stüwe. *Geodynamics of the lithosphere: an introduction.* Springer, Berlin, second edition, 2007. 28

[93] F. Tröltzsch. *Optimal control of partial differential equations, theory, methods and applications*, volume 112 of *Graduate studies in mathematics.* American

Mathematical Society, Providence, Rhode Island, 2010. Translated from the 2005 German original by Jürgen Sprekels. 32, 33, 63

[94] R. Verfürth. *A review of a posteriori error estimation and adaptive mesh-refinement techniques.* Advances in numerical mathematics. Wiley, 1996. 15, 132, 140

[95] B. Vexler. *Adaptive finite element methods for parameter identification problems.* Dissertation, Heidelberg University, 2004. 33, 63

[96] S. F. Walter. *Structured higher-order algorithmic differentiation in the forward and reverse mode with application in optimum experimental design.* Dissertation, Humboldt-University at Berlin, 2011. 14, 42, 46, 49, 180

[97] C. K. F. Weiler. *Optimum experimental design for the identification of Gaussian disorder mobility parameters in charge transport models of organic semiconductors.* Dissertation, Heidelberg University, 2014. 15, 17, 36, 39, 42, 50, 52, 66

Erratum

After the completion of this thesis, I observed a mix-up of nomenclature in Chapter 10. The numerical results remain unchanged.

On page 151 the original text is

We disturb the values of the measurement functions by an additive normally distributed error with zero mean and three different standard deviations, $\sigma_i = 0.1, \sigma_i = 0.2,$ *and* $\sigma_i = 0.3, i = 1, .., 8.$ *This results in variances of* $\sigma_i^2 = 0.01, \sigma_i^2 = 0.04,$ *and* $\sigma_i^2 = 0.09, i = 1, .., 8,$ *respectively. Hence, we get three sets of measurement data with* 1%, 4% *and* 9% *noise and corresponding perturbations:*

$$\begin{aligned} 1\% :& \quad \varepsilon = (0.040, -0.066, -0.019, -0.004, 0.023, -0.009, 0.019, -0.008), \\ 4\% :& \quad \varepsilon = (0.048, 0.197, -0.150, 0.121, 0.157, 0.048, 0.312, 0.095), \\ 9\% :& \quad \varepsilon = (-0.390, 0.393, 0.251, 0.488, 0.355, 0.352, -0.536, 0.034). \end{aligned}$$

Instead of stating the measurement noise in terms of percentage points, it is more precise to state the different standard deviations. The text changes to

We disturb the values of the measurement functions by an additive normally distributed error with zero mean and three different standard deviations, $\sigma_i = 0.1, \sigma_i = 0.2,$ *and* $\sigma_i = 0.3, i = 1, .., 8.$ *This results in variances of* $\sigma_i^2 = 0.01, \sigma_i^2 = 0.04,$ *and* $\sigma_i^2 = 0.09, i = 1, .., 8,$ *respectively. Hence, we get three sets of measurement data with corresponding perturbations:*

$$\begin{aligned} \sigma_i = 0.1 :& \quad \varepsilon = (0.040, -0.066, -0.019, -0.004, 0.023, -0.009, 0.019, -0.008), \\ \sigma_i = 0.2 :& \quad \varepsilon = (0.048, 0.197, -0.150, 0.121, 0.157, 0.048, 0.312, 0.095), \\ \sigma_i = 0.3 :& \quad \varepsilon = (-0.390, 0.393, 0.251, 0.488, 0.355, 0.352, -0.536, 0.034). \end{aligned}$$

In the same manner throughout the rest of Chapter 10, the percentage points 1%, 4% and 9% have to be replaced by the standard deviations $\sigma_i = 0.1, \sigma_i = 0.2$, and $\sigma_i = 0.3$, respectively. In particular, they have to be replaced on pages 153 – 164, including Table 10.1 and the captions of Table 10.2, Figure 10.5, Table 10.3 and Table 10.4.