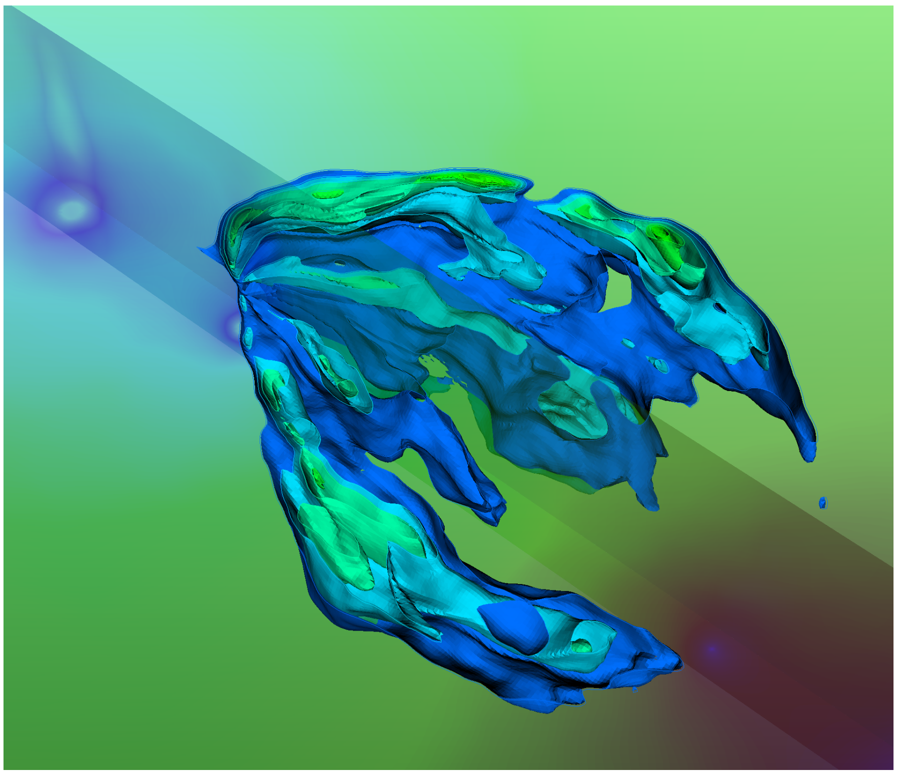
DISSERTATION
submitted to the
Combined Faculty for the Natural Sciences and for Mathematics
of the
Ruperto-Carola University of Heidelberg, Germany
for the degree of
Doctor of Natural Sciences

Put forward by

Dipl.-Math.techn. Adrian Q.T. Ngo
Born in: Saigon

Oral examination: 13.02.2015

Discontinuous Galerkin based Geostatistical Inversion of Stationary Flow and Transport Processes in Groundwater



Advisors: Prof. Dr. Peter Bastian
Prof. Dr.-Ing. Olaf A. Cirpka

To my loving parents

Acknowledgements

Being an interdisciplinary effort, this work would not have been possible without contributions from different sources. My gratitude goes to

- the Baden-Württemberg Stiftung for the financial support of the research project *HPC-8* which enabled the outcome presented in this work,
- the Faculty of Mathematics and Computer Sciences and the Interdisciplinary Center for Scientific Computing (IWR) at the Heidelberg University for providing an excellent infrastructure to complete a doctoral thesis,
- Peter Bastian for giving me the opportunity to work on this interesting subject in such a friendly and cooperative atmosphere as present in his workgroup *Parallel Computing*, for an exceptional education in numerical mathematics and for the many suggestions and comments,
- Olaf Cirpka for the coordination and supervision of the research project and from whom I have learned the theoretical aspects of the QLGA inversion method,
- Olaf Ippisch who taught me a great deal about soil physics, parallel computing and presentation techniques,
- Ronnie Schwede and Wei Li from whom I learned the algorithmic part of the inversion scheme and how the results are supposed to look like,
- Jorrit Fahlke who added me to the registration list for my first DUNE workshop (with which all began) and who always was patient and came up with a brilliant idea when I got stuck with implementation,
- Ole Klein for many fruitful discussions and whose view on the mathematical aspects of inversion clarified my own view,
- Felix Heimann who helped me countless times with really useful hints,
- Markus Blatt for a really efficient AMG-based linear solver,
- Rebecca de Cuveland who extended this powerful linear solver with new features,
- Steffen Müthing whose inexhaustible knowledge regarding software development and hardware issues not only speeded up PDELab but also my numerical experiments,
- all DUNE developers for their great work,
- Michael Geiger for constructive discussions,
- Sandra Barteit and Joachim Simon for their technical support,
- Ingrid Hellwig who was always helpful and kept bureaucracy at a minimal level,
- Pavel Hron and Jurgis Pods who invited me to join the IWR football team,
- Betty Luong and Maggie Quach for the grammatical corrections,
- and Ying who has always been there for me and gave me mental support.

Abstract

The hydraulic conductivity of a confined aquifer is estimated using the quasi-linear geostatistical approach (QLGA), based on measurements of dependent quantities such as the hydraulic head or the arrival time of a tracer. This requires the solution of the steady-state groundwater flow and solute transport equations, which are coupled by Darcy's law. The standard Galerkin finite element method (FEM) for the flow equation combined with the streamline diffusion method (SDFEM) for the transport equation is widely used in the hydrogeologists' community. This work suggests to replace the first by the two-point flux cell-centered finite volume scheme (CCFV) and the latter by the Discontinuous Galerkin (DG) method. The convection-dominant case of solute (contaminant) transport in groundwater has always posed a special challenge to numerical schemes due to non-physical oscillations at steep fronts. The performance of the DG method is experimentally compared to the SDFEM approach with respect to numerical stability, accuracy and efficient solvability of the occurring linear systems. A novel method for the reduction of numerical under- and overshoots is presented as a very efficient alternative to local mesh refinement. The applicability and software-technical integration of the CCFV / DG combination into the large-scale inversion scheme mentioned above is realized. The high-resolution estimation of a synthetic hydraulic conductivity field in a 3-D real-world setting is simulated as a showcase on Linux high performance computing clusters. The setup in this showcase provides examples of realistic flow fields for which the solution of the convection-dominant transport problem by the streamline diffusion method fails.

Zusammenfassung

Die hydraulische Leitfähigkeit in einem gespannten Grundwasserleiter wird mit dem QLGA Inversionsverfahren (Quasi-linearer geostatistischer Ansatz) geschätzt, indem man Messungen von abhängigen Größen wie etwa des hydraulischen Drucks und der Ankunftszeit eines eingeleiteten Tracers (z.B. gefärbter Indikator) auswertet. Dabei müssen die stationären Gleichungen für Strömung und Transport, die über Darcys Gesetz gekoppelt sind, numerisch gelöst werden. Bei Hydrogeologen ist die Standard Finite Elemente Methode (FEM) zur Lösung der Strömungsgleichung und die Methode der Stromlinien-Diffusion (SDFEM) zur Lösung der Transportgleichung weit verbreitet. In dieser praxisorientierten Arbeit wird dieser Ansatz durch das zellzentrierte Finite-Volumen-Verfahren (CCFV) kombiniert mit dem unstetigen Galerkin-Verfahren (DG) ersetzt. Der konvektionsdominante Stofftransport im Grundwasser stellt seit jeher eine Herausforderung an numerische Methoden dar. Das DG-Verfahren wird mit dem SDFEM-Verfahren verglichen, und zwar hinsichtlich Stabilität, Genauigkeit und der effizienten Lösbarkeit der auftretenden linearen Gleichungssysteme. Eine neue Methode zur Dämpfung von numerischen Unter- und Überschwinger wird als eine sehr effiziente Alternative zur lokalen Gitterverfeinerung präsentiert. Die Anwendbarkeit der Kombination CCFV / DG auf das erwähnte großskalige Inversionsverfahren wird realisiert. In einer dreidimensionalen Versuchsumgebung wird ein synthetisch generiertes Parameterfeld mit hoher Auflösung geschätzt. Wir demonstrieren damit eine parallele Inversion, die auf Linux HPC-Clusters durchgeführt werden kann. Die in diesem realistischen Beispiel auftretenden Strömungsfelder induzieren Transportprobleme, die im konvektionsdominanten Fall mit der Methode der Stromlinien-Diffusion nicht mehr lösbar sind.

Contents

Contents vii

List of Figures xi

List of Tables xiii

List of Symbols xvii

1 Introduction 1

2 Geostatistical Modeling 9

2.1 Hydraulic conductivity in the confined aquifer . . . 9

2.2 Spatial description of the hydraulic conductivity . . . 12

2.2.1 Stochastic model . . . 12

2.2.2 Discretization of the log hydraulic conductivity Y . . . 13

2.3 Random field generation algorithm . . . 14

3 Inversion 19

3.1 Deterministic approach . . . 19

3.2 Statistical approaches . . . 21

3.3 Bayesian inference . . . 23

3.3.1 Uncertain prior information . . . 23

3.3.2 Likelihood function . . . 24

3.3.3 Posterior distribution . . . 25

3.3.4 Link with the deterministic approach . . . 26

3.4 The quasi-linear geostatistical approach (QLGA) . . . 26

3.4.1 Gauss-Newton scheme for the MAP estimate . . . 26

3.4.2 Uncertainty quantification . . . 31

3.4.3 Test of unbiasedness . . . 32

3.4.4 An estimator for the optimal value of the objective function . . . 32

3.5 Inversion algorithm . . . 33

4 Model Equations and the Computation of Sensitivities **37**
4.1 Model equations 37
4.1.1 Groundwater flow 37
4.1.2 Subsurface solute transport 38
4.1.3 Weak formulation of coupled equations 42
4.2 Sensitivities and cross-covariance matrices 44
4.2.1 Computation of sensitivities 44
4.2.2 Adjoint equation method 45
4.2.3 Discretized sensitivities 52
4.2.4 Efficient computation of cross-covariance matrices 53
4.3 Subroutines of the inversion algorithm 54

5 Numerical Solution of Model and Adjoint Equations **57**
5.1 Types of equations 57
5.2 Discrete solution spaces 59
5.3 FEM / SDFEM 60
5.4 CCFV / DG 62
5.4.1 Preliminary definitions 62
5.4.2 Two-point flux cell-centered finite volume method (CCFV) . . . 63
5.4.3 Flux reconstruction 65
5.4.4 Discontinuous Galerkin method (DG) 66
5.5 Adaptive mesh refinement 68
5.6 Diffusive L^2-projection 71
5.7 Efficient solution of the arising linear systems 71
5.7.1 Flow equation and diffusive L^2-projection 72
5.7.2 Transport equation 72

6 Implementation **75**
6.1 The `DUNE` software framework 75
6.2 The `DUNE` module `dune-gesis` 76
6.2.1 Directory structure 76
6.2.2 External libraries 77
6.2.3 The main performance optimization steps 79

7 Numerical Studies **81**
7.1 An example with a regular solution 82
7.2 An example with a less regular solution 87
7.3 Post-processed DG(1) versus SDFEM 90
7.4 Forward transport in $\mathbf{2}$-D 92
7.5 Forward transport in $\mathbf{3}$-D 96
7.6 Simulations in a nested cells environment 99

7.6.1 The setting . . . 99
7.6.2 Scenario 1: Influence of adaptivity in a fully coupled 2-D inversion 102
7.6.3 Scenario 2: FEM/SDFEM vs. CCFV/DG(1) as forward solvers 106
7.6.4 Scenario 3: Parallel scalability tests for the forward solvers . . . 108
7.6.5 Scenario 4: Fully-coupled inversion with different strategies . . 110
7.6.6 Scenario 5: Varying the input data . . . 114
7.6.7 Overview of time consumption . . . 117

8 Conclusion and Outlook **119**

Appendix **125**

A Basic linear algebra . . . 125
1 Symmetric positive (semi-)definite matrices . . . 125
2 Matrix identities . . . 126
3 Quadratic functions . . . 127
4 Toeplitz and circulant matrices . . . 128
B Basic multivariate statistics . . . 131
1 Random vectors and probability distributions . . . 131
2 Joint distributions and marginalization . . . 132
3 Conditional distributions and stochastic independence . . . 133
C The multivariate Gaussian distribution . . . 134
1 Definition and basic properties . . . 134
2 Conditioning . . . 137
3 χ^2 distribution . . . 138
D Random fields and variogram models . . . 138
E Basic definitions of functional analysis . . . 141
1 Function spaces . . . 141
2 Broken Sobolev spaces . . . 144
3 Differentiation in Banach spaces . . . 145
F Hardware in use . . . 146
G Hydraulic parameters . . . 147

References **149**

List of Figures

2.1 Aquifer types 10
2.2 Periodic embedding in 1-D 16
2.3 Periodic embedding in 2-D 17
2.4 2-D random field samples: Comparing variogram models 18

5.1 Downwind numbering of locally refined mesh cells 74

7.1 2-D example by John: Analytical solution 82
7.2 2-D example by John: Comparing cell numbering strategies 84
7.3 2-D example by John: Numerical convergence behavior 86
7.4 2-D example by López and Sinusía: Reference solution 88
7.5 2-D example by López and Sinusía: Numerical convergence behavior . 89
7.6 2-D example by López and Sinusía: Warped plots 91
7.7 2-D example by López and Sinusía: Zoomed error plots 91
7.8 2-D forward problem: SDFEM / post-processed DG(k) / adaptive DG . 93
7.9 3-D forward problem: SDFEM / post-processed DG(k) / adaptive DG . 97
7.10 3-D nested cell setup: Cross-sectional plots 101
7.11 2-D inversion: Investigating the influence of adaptive refinement on the final result of a parameter estimation process. The temporal moments displayed are taken from setup #3. 104
7.12 3-D nested cells environment: Quality of different solutions of the transport equation for different Péclet numbers 107
7.13 3-D inversion: Comparing original with estimated fields (flow fields for setup #2) 112
7.14 3-D inversion: Estimated variance and histogram of the weighted error . 113
7.15 3-D inversion: Convergence behavior 115
7.16 3-D inversion: Convergence behavior 115
7.17 3-D inversion: undisturbed and disturbed data 116

D.1 Comparison of variogram models 140

List of Tables

5.1 Overview: Source and boundary terms for the diffusion equation 57
5.2 Overview: Source, reaction and b.c. terms for the transport equation . . 58
5.3 Overview: Source, reaction and b.c. terms for the transport equation . . 58

7.1 2-D example by John: Comparing cell numbering strategies 83
7.2 2-D forward solutions: Adaptive DG 94
7.3 2-D forward problem: DG(k) vs. SDFEM on structured meshes 95
7.4 3-D forward problem: Parallel adaptive DG 98
7.5 3-D forward problem: DG(k) vs. SDFEM on structured meshes 98
7.6 3-D nested cells environment with four different setups 100
7.7 2-D inversion: Influence of adaptivity 103
7.8 2-D inversion: Influence of adaptivity 105
7.9 3-D nested cells environment, comparing forward solvers 106
7.10 3-D nested cells environment, case of low mesh Péclet numbers 107
7.11 3-D nested cells environment, strong scalability test for CCFV/DG(1) . 108
7.12 3-D nested cells environment, weak scalability test for CCFV/DG(1) . . 109
7.13 3-D inversion: Average runtime per inversion step 111
7.14 3-D inversion: Total inversion runtimes 111
7.15 3-D inversion: Decreasing measurement error and disturbing data . . . 114
7.16 3-D inversion: Overview of time consumption 117

F.1 Computing clusters at the IWR Heidelberg 146
F.2 Multi-core machines at the IWR Heidelberg 146
G.1 Typical values for the porosity 147
G.2 Typical values for the hydraulic conductivity 147

List of Algorithms

2.1 Random Field Generation (by Cholesky factorization) 14
2.2 Random Field Generation (by Dietrich and Newsam) 15
3.1 QLGA (by Kitanidis, Cirpka and Nowak) 34
3.1 (continued): QLGA (by Kitanidis, Cirpka and Nowak) 35
4.1 Solving forward problems and take simulated measurements 54
4.2 Sensitivities and cross-covariance matrices 55
5.1 h-adaptive refinement 70

List of Symbols

Greek	Units	Description	Page
α_ℓ	$[m]$	longitudinal dispersivity	39
α_t	$[m]$	transversal dispersivity	39
$\boldsymbol{\beta}$	$[m\ s^{-1}]$	trend coefficients (random vector $\in \mathbb{R}^{N_\beta}$)	13
γ		DG penalization factor	67
γ		variogram function	13
δ_t^{SD}		SDFEM stabilization parameter	61
ϵ_k		k-th measurement error	24
ε		diffusion coefficient	61
η		adaptive error estimator	68
η_ε^ℓ		regularized point source	46
θ		porosity	9
$\boldsymbol{\theta}$		vector of structural parameters	12
κ	$[m^2]$	intrinsic permeability	11
μ_w	$[kg\,m^{-1}s^{-1}]$	dynamic viscosity of water	11
$\widehat{\boldsymbol{\xi}}$		coefficient vector	28, 29
ρ		probability density	131
ρ_w	$[kg\ m^{-3}]$	water density	11
σ_Y^2		variance in the variogram model, field variance	13
ϕ	$[m]$	hydraulic head	11
χ_r^2		chi square distribution with r degrees of freedom	138
$\chi_{1-\alpha}^2(r)$		100α-th percentile of a χ_r^2 distribution	33
ψ_ℓ^ϕ		adjoint state of hydraulic head $\phi(\vec{x})$ at $\vec{x}_\ell$	51
$\psi_\ell^{m_k}$		adjoint state of $m_k^c(\vec{x}),\ k \in \{0,1\}$, at $\vec{x}_\ell$	49
Γ_D		Dirichlet boundary	38
Γ_N		Neumann boundary	38
Γ_0		characteristic boundary	39
Γ_-		inflow boundary	39
Γ_+		outflow boundary	39
Ω		physical domain $\subset \mathbb{R}^d$	12

Latin	**Units**	**Description**	**Page**
c	$[kg\ m^{-3}]$	solute concentration	38
d		space dimension 2 or 3	12
$\boldsymbol{d}_{\text{obs}}$		vector of observed measurement values (observations)	1
f		face	62
$\boldsymbol{f}$		vector of simulated measurements (model function with argument $\boldsymbol{y}$)	1
g		gravitational acceleration	11
$\boldsymbol{g}$		observation operator	19
h		mesh size	59
h_ν		characteristic mesh size on refinement level ν	59
$\boldsymbol{h}_\ell$		ℓ-th row of the sensitivity matrix $\mathcal{H}$	53
ℓ_k	$[m]$	correlation length in x_k-direction, $k = 1, ..., d$	13
m_0^c	$[kg\ m^{-3}]$	zeroth order moment of solute concentration	40
m_1^c	$[kg\ s\ m^{-3}]$	first order moment of solute concentration	40
n		total number of mesh cells of $\mathcal{T}_h$ $(n = N$ for $\mathcal{T}_{h_0})$	60
n_k		number of mesh cells in x_k-direction, $k = 1, ..., d$	14
$\vec{n}, \vec{n}_f$		unit outer normal vector (on face f)	38, 60
$\boldsymbol{p}$		parameter field	19
$\vec{q}$	$[m\ s^{-1}]$	Darcy velocity	11
t		time	9, 40
t		mesh cell	59
$\vec{v}$	$[m\ s^{-1}]$	pore water velocity (seepage velocity), $\vec{v} = \vec{q}/\theta$	38
$\vec{x}$	$[m]$	coordinate vector $\in \mathbb{R}^d$	9
$\vec{x}_\ell$	$[m]$	measuring location	44, 45
$\tilde{w}_{\text{inj}},\tilde{w}_{\text{ext}}$	$[s^{-1}]$	injection/extraction rate	38,61,64
$\boldsymbol{y}$		discrete version of Y (random vector $\in \mathbb{R}^N$)	13
$\mathcal{B}_h$		set of boundary faces	62
$Cov[.,.]$		covariance	132
$\mathcal{C}_{dd}$		covariance matrix of measurement errors	24
D_{eff}	$[m^2\ s^{-1}]$	SWIP effective diffusivity	67
D_m	$[m^2\ s^{-1}]$	molecular Diffusion	39
$\mathcal{D}_S$	$[m^2\ s^{-1}]$	Scheidegger Dispersion tensor	39
$E[.]$		expected value	131
$\mathcal{E}_h$		set of interior faces	62
$\boldsymbol{F}$		general model function with argument $\boldsymbol{p}$	19
F		cumulative distribution function	131
Id		identity matrix	39, 53
$\mathcal{H}$		sensitivity matrix	27

$H^s(\Omega)$		Sobolev space of order s	143
$H(\text{div}, \Omega)$		Sobolev space of diffusive fluxes	143
$H^s(\Omega, \mathcal{T}_h)$		broken Sobolev space of order s	144
$H(\text{div}, \mathcal{T}_h)$		broken Sobolev space of diffusive fluxes	144
$\mathscr{J}$		objective function	25
$\widehat{\mathscr{J}}$		reduced cost functional	46
K	$[m\ s^{-1}]$	hydraulic conductivity	11
K_h^{eff}	$[m\ s^{-1}]$	harmonic average	64
$L^2(\Omega)$		space of square integrable functions	142
L		Lagrange function	46
L_k	$[m]$	domain length in x_k-direction, $k = 1, ..., d$	14
M		total number of measurement values	19
N		dimension of $\boldsymbol{y}$ ($n = N$ for $\mathcal{T}_{h_0}$)	13
N_β		number of different trends	13
P		number of parallel processes	108
P		probability measure	131
$\mathcal{P}_h^t$		mesh Péclet number	61
$\mathcal{R}_{yy}$		spatial covariance matrix (symmetric Toeplitz)	14
$\mathcal{S}_{yy}$		extended covariance matrix after periodic embedding (symmetric circulant)	14
$\mathcal{T}_h$		mesh with characteristic mesh size h	13, 59
Var[.]		variance	132
$W_{h,k}$		broken polynomial space	60
$W_{\text{inj}}, W_{\text{ext}}$		injection / extraction well	38,59
V_h		continuous polynomial space	60
$\mathcal{X}$		deterministic zonation matrix $\in \mathbb{R}^{N \times N_\beta}$	13
Y		log hydraulic conductivity: $Y = \ln K$	12

Random vectors and vectors in general are denoted by bold lower-case Greek or Latin letters. Coordinate or velocity vectors in the physical space $\mathbb{R}^d$ are denoted by lower case letters with an arrow on top. Matrices (except for the identity matrix) are written in upper-case calligraphic style.

Acronyms	**Description**	**Page**
2-D	two-dimensional	
3-D	three-dimensional	
AMG	algebraic multi-grid	71
BiCGSTAB	stabilized bi-conjugate gradient method	71
BCGS	BiCGSTAB	108, 109
BVP	boundary value problem	37
CG	conjugate gradient method	71
CM	conditional mean	22
DG	Discontinuous Galerkin	66
FEM	finite element method	60
GMRES	generalized minimal residual	71
MAP	maximum a posteriori	22
MCMC	Markov Chain Monte Carlo	22
MLE	maximum likelihood	21
PDE	partial differential equation	1
SDFEM	streamline diffusion method	4, 60
SUPG	streamline upwind Petrov-Galerkin	4
SWIP	symmetric weighted interior penalty	66

Chapter 1

Introduction

Assume that we have a mathematical model which relates a finite number of physical parameters $\boldsymbol{y} = (y_1, ..., y_N)^T$ to a collection of discrete observations $\boldsymbol{d}_{\text{obs}} = (d_1, ..., d_M)^T$ via a system of equations

$$\boldsymbol{d}_{\text{obs}} = \boldsymbol{f}(\boldsymbol{y}) \,. \tag{1.1}$$

The **forward problem** is to find $\boldsymbol{d}_{\text{obs}}$ given $\boldsymbol{y}$. Very often, evaluating the model function $\boldsymbol{f}$ involves a computationally demanding effort of solving a system of partial differential equations (PDEs). A mathematical problem is called **well-posed** in the sense of Hadamard [Mandelbrojt and Schwartz, 1965] if:

1. A solution exists.
2. The solution is unique.
3. The solution depends continuously on the data.

An **inverse problem** or **parameter identification problem**, here denoted (P1), is by definition the opposite of the forward problem: Given a set of measurements $\boldsymbol{d}_{\text{obs}}$, the task is to reconstruct the parameters $\boldsymbol{y}$. Inverse problems are usually ill-posed and require the application of stabilizing constraints if the number of measurement values is much less than the number of parameters ($M \ll N$). In practice, the measured data always contain some amount of noise (measurement error) and the model may not capture the physics of the simulated system (model error). The solution of an inverse problem does not necessarily satisfy the model equation (1.1) directly. In fact, one can only postulate that the discrepancy between simulated and observed measurements is minimal with respect to some appropriate norm.

A third class of problem in this context, not discussed in this work, is called the **model identification problem**: finding $\boldsymbol{f}$ given examples of $\boldsymbol{d}_{\text{obs}}$ and $\boldsymbol{y}$.

Inverse problems play an important role in science and technology. Usually, a direct measurement of material properties is practically impossible or very expensive. The only way is to calculate or estimate them from indirect measurements of related quantities for which we have a mathematical model.

The following idea provides the principle behind any iterative scheme for the solution of an inverse problem, provided that the problem is well-defined: Starting with an initial guess $\boldsymbol{y}_0$ for the unknown parameters we simulate the forward problem (1.1) and pick the simulated values from model run k at the measurement points. Unless $\boldsymbol{y}_k = \boldsymbol{y}$, the discrepancy between observed and simulated measurements will be much larger than the model and measurement errors. This can be used to indicate the quality of the estimation. The classical inversion approach is to define and minimize a cost function $\mathscr{J}(\boldsymbol{y})$ that contains the discrepancy and the stabilizing constraints. The statistical approach seeks for point estimates of a probability distribution that incorporates this information.

In this work, the unknown parameter of primary interest is the hydraulic conductivity field of an aquifer. Accurate knowledge of the hydraulic conductivity is of utmost importance for the management of groundwater resources and the remediation of contaminated aquifers. In Germany[1], more than 70% of the drinking water supply (for domestic, agricultural and industrial use) originates from groundwater. We simulate single-phase flow and transport processes in the saturated zone under steady-state conditions. Knowing the hydraulic conductivity and given the boundary conditions, it is relatively easy to compute the hydraulic head distribution. The corresponding forward problem is the **steady-state groundwater flow problem** (P2). From its solution, the flow field can be reconstructed and used to simulate transport processes. Due to very small dispersivities and molecular diffusion, (non-reactive) solute transport in groundwater can be convection-dominated. The transport of a conservative tracer can be described by the advection-dispersion equation. Considering temporal moments of its concentration in the coupled forward model leads to the **steady-state singularly perturbed convection-diffusion equation** (P3).

We are interested in the efficient solution of the three types of problems (P1)-(P3).

State of the art

While an increasing amount of data generally improves the quality of inversion schemes, the number of measurement points should be minimal for economical and ecological reasons. In order to make the most out of a low number of measurement points, it is essential that a parameter estimation scheme is able to cope with different types of measurements.

The **quasi-linear geostatistical approach (QLGA)** is a parameter estimation scheme with which spatial parameters such as the hydraulic conductivity can be estimated on

[1]Statistisches Bundesamt: Öffentliche Wasserversorgung 2010, Fachserie 19, Reihe 2.1.1, Erscheinungsfolge: dreijährlich, Erschienen am 05. Februar 2013, Artikelnummer: 2190211109004

the basis of direct or indirect measurements of related quantities such as the hydraulic head, the concentration of a tracer or its arrival time. The inversion scheme was developed by P. Kitanidis and later extended and made more efficient by O.A. Cirpka and W. Nowak. The following list of works contributed to its applicability and success in 2-D and low-resolution 3-D simulations of real world problems:

- dimensional reduction of the cokriging system enhancing the Gauss-Newton approach [Kitanidis, 1995],
- efficient computation of sensitivities based on adjoint states [Sun, 1994],
- extension of the adjoint-based inversion scheme to tracer data [Cirpka and Kitanidis, 2001],
- FFT-based methods for the efficient computation of the cross-covariance matrices [Nowak et al., 2003].

Since the main focus of this work lies on the numerical solution of the occurring boundary value problems (in 3-D), especially the convection-dominant solute transport equation, we do not elaborate on alternative parameter estimation schemes [Alcolea et al., 2006; Crestani et al., 2012; Doherty et al., 2010; Schöniger et al., 2012].

The steady-state groundwater flow equation is a linear elliptic PDE with a highly variable coefficient. For the solution of the linear system arising from a standard finite element or finite volume discretization, we apply a very efficient parallel solver based on algebraic multigrid (AMG) presented by Blatt [2010].

The numerical solution of convection-diffusion problems has a long tradition. Due to the fact that a linear monotonicity preserving scheme can be at most first-order accurate (Godunov's Theorem), all existing schemes suffer from a trade-off between numerical diffusion (too much smearing) and spurious oscillations (under- and overshoots) near internal or boundary layers where the gradient of the solution is very large (steep fronts). In each practical application, the decision has to be made whether the one or the other deficit can be tolerated, leading to the appropriate choice of a numerical scheme.

Amongst the vast literature on the subject, the books of Roos et al. [2008] and Kuzmin [2010] provide excellent overviews of state-of-the art classes of schemes. We give a short overview of the most prominent methods, list their main advantages and disadvantages, hereby drawing on the results presented by Augustin et al. [2011], who have worked on a special 2-D problem (Hemker problem). They compared the numerical solutions at specific cut lines with respect to the size of maximal under- and overshoots, the width of smeared internal layers and the performance in computing time, revealing important properties of the different schemes.

- The Scharfetter-Gummel scheme is a first-order finite volume scheme. It is efficient and oscillation-free, but the solution is strongly smeared at layers. A higher order extension is not available.

- The **Streamline Diffusion finite element method (SDFEM)**, also known as Streamline Upwind Petrov-Galerkin (SUPG) method, adds a residual-based stabilization term to the standard Galerkin method [Brooks and Hughes, 1982]. SDFEM belongs to the less time-consuming methods that are capable of resolving the steep fronts well. Due to its simplicity, it has been the mainstream approach for decades and a standard method in the hydrogeologists' community [Bear and Cheng, 2010; Cirpka and Kitanidis, 2001; Couto and Malta, 2008; Gordon et al., 2000; Nowak and Cirpka, 2006] where our practical application originates from. However, the optimal choice of a user-defined stabilization parameter is an open question.

- The Continuous Interior Penalty (CIP) method [Burman et al., 2010; Roos et al., 2008] adds a symmetric stabilization term to the standard Galerkin method that penalizes jumps of the gradient across faces (edge stabilization technique). It introduces connections between unknowns of neighboring mesh cells and leads to a discretization with a wider matrix stencil. Comparisons to SDFEM may be found in Burman and Hansbo [2004].

- Spurious Oscillations at Layers Diminishing (SOLD) methods, originally developed by Hughes et al. [1986] and further investigated by John and Knobloch [2007a,b, 2008], suppress oscillations caused by SDFEM by adding a further stabilization term introducing diffusion orthogonal to streamlines (crosswind diffusion). This term is in general *non-linear*. Therefore, a non-linear equation has to be solved for a linear problem. Furthermore, the stabilization term contains another user-defined parameter whose optimal choice might become difficult for complicated problems. SOLD methods are capable of reducing numerical oscillations at a higher computational cost. The larger the stabilization parameter, the better the reduction. However, non-linearity also increases and the iterative non-linear solver might not converge [Augustin et al., 2011].

- Algebraic Flux Correction (AFC) is a general approach to design high resolution schemes for the solution of time-dependent transport problems that ensure the validity of the discrete maximum principle [Kuzmin, 2006, 2010]. Whereas the aforementioned stabilization methods modify the bilinear form of a finite element method (FEM), AFC methods modify the linear system arising from a FEM discretization by adding discrete diffusion to the system matrix and appropriate anti-diffusive fluxes to the right hand side. The anti-diffusive fluxes are *non-linearly* dependent on the computed solution. Depending on whether the algebraic constraints are being imposed on the semi-discrete or the fully discrete level, flux limiters of TVD-type (total variation diminishing) or FCT-type (flux corrected transport) can be constructed. Only the FEM-TVD schemes can be used to solve the steady-state convection-diffusion equation directly. Using FEM-FCT schemes, a pseudo time stepping to the stationary limit of the associated time-dependent problem would deliver the steady-state solution [Kuzmin, 2006]. A suitable linearization technique for the anti-diffusive fluxes exists only for the FEM-FCT scheme [Kuzmin, 2009]. According to the studies by John and Schmeyer [2008, 2009], FEM-FCT schemes yield qualitatively the best solution and, beyond that, the linear FEM-FCT scheme is efficient. The authors recommend the linear FEM-FCT for the solution of instationary problems. Linearized AFC-based methods for the solution of the stationary transport problem are not available.

- **Discontinuous Galerkin (DG) methods** use piecewise polynomials, that are not required to be continuous across faces, to approximate the solution. The total number of degrees of freedom on a structured mesh with cuboidal cells is $O(n \cdot (k+1)^d)$ where n is the number of mesh cells, d is the dimension of the domain and k is the polynomial degree. Compared to a continuous Galerkin FEM method, a DG method using the same polynomial space on the same structured mesh requires more unknowns. This disadvantage is balanced by a long list of advantages that has made DG increasingly attractive in computational fluid dynamics in the last decade: DG methods are readily parallelizable, lead to discretizations with compact stencils (i.e. the unknowns in one mesh cell are only connected to the unknowns in the immediate neighboring cells), a higher flexibility in mesh design (non-conforming meshes are possible in adaptive h-refinement) and the availability of different polynomial degrees on different mesh cells (adaptive p-refinement). Furthermore, DG schemes satisfy the local, cell-wise mass balance which is a crucial property for transport processes in a porous medium. They are particularly well-suited for problems with discontinuous coefficients and effectively capture discontinuities in the solution. In the comparative study by Augustin et al. [2011], the DG method gives the best results regarding sharpness of the steep fronts and produces small errors with respect to reference cut lines, whereas under- and overshoots are larger than those produced by the SDFEM method. For the discretization of first-order hyperbolic problems, upwinding is incorporated into the formulation of DG schemes, evading the need for user-chosen artificial diffusion parameters. The books of Kanschat [2008b], Rivière [2008] and Pietro and Ern [2012] offer a comprehensive introduction to this class of methods.

For time-dependent problems, where explicit time stepping schemes combined with finite volume or DG discretizations can be used, slope limiters may be constructed from the solution of one time-step to preserve monotonicity in the following time-step. To the best of our knowledge, for the immediate solution of stationary problems, a post-processing technique of this type is not available.

In the simulation of many applications (e.g. biochemical reactions or combustion), the concentration of a species must attain physical values (numerical under- and over-shoots are not accepted) although the position of the plume may be allowed to be inaccurate. But in the context of a parameter estimation scheme, which allows for measurement errors, a small amount ($\approx 5\%$) of spurious oscillations in the solution is tolerable whereas the correct localization of steep fronts is of primary interest.

For high resolution 3-D simulations, direct sparse solvers are limited by their memory consumption. The main purpose of the stabilization term in the SDFEM method is not only to provide a solution with bounded under- and overshoots but also to improve the iterative solvability of the linear system arising from the SDFEM discretization. For an upwind scheme applied to a first-order hyperbolic problem, it is well-known that numbering the unknowns in a fashion that follows approximately the direction in which information is propagated will improve the performance and stability of iterative linear solvers of ILU or Gauss-Seidel type [Bey and Wittum, 1997; Hackbusch and Probst, 1997; Reed and Hill, 1973].

Contributions of this work

Real world problems simulated in 3-D at a high spatial resolution require so many unknowns ($10^6 - 10^8$) that their floating point representation would not fit into the memory of a single-CPU machine, let alone the long computing time a single calculation would require. Developments on the hardware market have pushed the usage of hyper-threading and multicore architectures during the last decade [Sutter, 03/2005]. Nowadays, academic research groups or small engineering companies can afford to purchase high-performance clusters with a handful of computing nodes providing the power of more than one hundred cores for their own usage.

The main contributions of this work comprise

- a realization of a fast solver for the convection-dominated transport equation based on reordering in flow direction and exploiting the upwind character of the DG scheme,

- the development of a diffusive L^2-projection for reducing non-physical oscillations near sharp fronts that are not resolved by the mesh,

- a systematic numerical comparison of SDFEM to DG for stationary convection-diffusion problems in the context of geostatistical inversion,

- a fully parallel implementation of all methods necessary for the complete inversion scheme, including the handling of multiple measurement types,
- a demonstration of the method on large scale 3-D problems with up to 26 million degrees of freedom in the parameter space (or 210 million degrees of freedom in the finite element solution space) and 512 measurement values of different types, run on a HPC cluster using up to 64 cores.

Outline

The remaining part of this work is structured as follows:

- Chapter 2 introduces the Geostatistical model and the FFT-based generation of random test fields,
- Chapter 3 derives the QLGA method from the Bayesian point of view,
- Chapter 4 introduces the forward problems. Furthermore, a derivation of the adjoint equations and the sensitivities based on the Lagrange formalism is presented and a description of the FFT-based computation of the cross-covariance matrices is given.
- Chapter 5 summarizes all boundary value problems and corresponding source terms occurring in Chapter 4. Afterwards, we consider two combinations of discretizations for their numerical solution: FEM/SDFEM and CCFV/DG.
  - * We present two approaches to reduce the under- and overshoots of the DG-solution of the transport problem:
    1. The first approach (Section 5.5) uses h-adaptive hanging-nodes refinement (1-irregular) on a cuboidal axis-parallel mesh based on the residual error estimator by Schötzau and Zhu [2009] combined with an error-fraction marking strategy.
    2. The second approach (Section 5.6) is a diffusive L^2-projection of the DG solution into the continuous Galerkin finite element subspace. It works directly on the structured coarse mesh.
  - * In addition, we have implemented an efficient way to solve the linear system for the DG discretization iteratively by exploiting a downwind cell-wise numbering of unknowns *before* the stiffness matrix is assembled (Section 5.7).
- Chapter 6 highlights implementation aspects which constitute a significant part of such a practical work.

- Numerical studies are presented in Chapter 7. The DG method is compared to a first order SDFEM implementation within the same code, i.e. the performance of the two different discretizations can be compared on the same computational grid using the same linear solver. A 3-D nested-cells environment generating a sufficiently complex steady-state flow field serves as the basis for a series of forward and inversion test cases.
- The results are summarized in Chapter 8.
- The appendix summarizes the mathematical basics required for the comprehension of this work.

Availability of the software

The numerical software used to perform the simulations is written in C++ and based on the libraries of the *Distributed and Unified Numerics Environment* `DUNE` [Bastian et al., 2008a,b, 2011] that provides MPI[1]-based parallel solution of partial differential equations. The implementation of the algorithms and numerical schemes presented in the chapters 2 – 5 are available in a new `DUNE` module named `dune-gesis` (Geostatistical Inversion based on Stationary Models), published by Ngo and Schwede [2014]. It was developed by the author in the course of his employment (2010-2014) at the *Interdisciplinary Center for Scientific Computing (IWR Heidelberg)* in cooperation with the workgroup for hydrogeology at the *Center for Applied Geosciences (ZAG Tübingen)*. The research project was funded by the *Baden-Württemberg Stiftung*[2] under the contract *HPC-8*.

[1]MPI (message passing interface) has the advantage that it can be used on both distributed and shared memory architectures. Multi-threading in `DUNE` is a topic of current research.

[2]http://www.bwstiftung.de/en

Chapter 2

Geostatistical Modeling

2.1 Hydraulic conductivity in the confined aquifer

We start with a brief clarification of some important terms related to hydrogeology which are relevant for the understanding of this work. For further details, we refer to [Bear and Cheng, 2010], one of the most comprehensive books on mathematical modeling of groundwater flow and contaminant transport.

Subsurface flow is flow through a **porous medium** (soil or rock). This medium consists of a solid matrix (mainly minerals or organic compounds) and pore space (void space). A porous medium can be characterized by a **representative elementary volume** (REV), a sample of sufficiently large volume such that a consistent ratio between the void space volume $V_{\text{void}}(t, \vec{x})$ and the sample's volume V_{REV} can be defined. This ratio is called the **total porosity**:

$$\theta(t, \vec{x}) = \frac{V_{\text{void}}(t, \vec{x})}{V_{\text{REV}}} \tag{2.1}$$

Soil porosity varies over a wide range of values, from $\theta = 0.1$ for sandstone to $\theta = 0.8$ for peat soil. We work with the assumption that the porosity is constant throughout the domain and that the porous medium does not change its shape: $\theta = 0.3$ is a typical value for gravel and sand. Groundwater hydrologists use the term **groundwater** primarily for that part of the subsurface water that occurs in the saturated zone where the entire interconnected void space is filled with water. A porous geological formation that contains groundwater is called an aquifer. A **confined aquifer** is (1) bounded from above and from below by impermeable formations, and (2) under pressure, i.e. the water level in a piezometer rises above the impermeable ceiling (potentiometric surface in the artesian well in Figure 2.1). A **phreatic** or **unconfined aquifer** is bounded from above by a phreatic surface (water table in Figure 2.1). The unsaturated zone or **vadose zone** is the layer between the ground surface and the water table. The pore space is partially filled with water. The rest is air. It is under direct influence of precipitation or evaporation

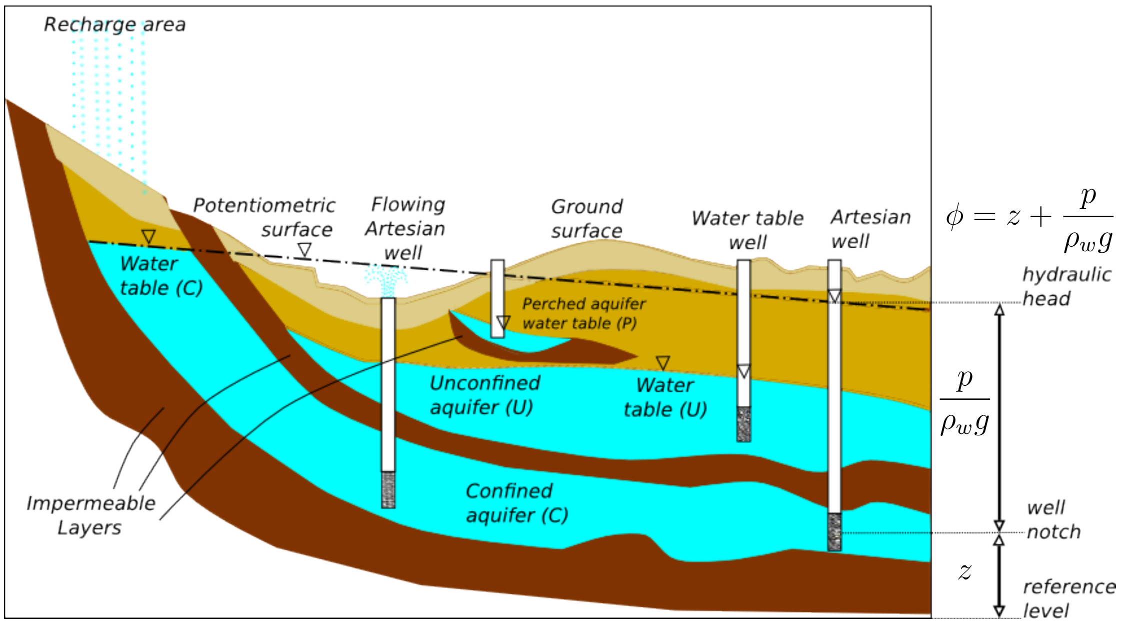


Figure 2.1: Cross-sectional cartoon of a geological formation, modified after the Colorado Geological Survey http://www.douglas.co.us/water/water-supply/what-is-an-aquifer/(last accessed: 2014-09-25)

leading to a natural recharge or discharge of the aquifer. Furthermore, the existence of surface water (stream, river, lake or wetland) may lead to a flooding or a drainage of the aquifer.

Natural flowing conditions that are nearly at steady-state over a long period of time (i.e. several days during which a tracer experiment is being conducted) can be expected, if at all possible, in a confined aquifer. In an unconfined aquifer, the influence of transient ambient flow may be reduced to be negligible in a nested-cell setup [Luo et al., 2006; Schwede et al., 2014]. Thus, it is possible to work with steady-state flow and transport models presented in [Cirpka and Kitanidis, 2001] and applied in this work. Steady-State equations can be solved much faster than transient equations. Unfortunately, we must point out that this computational advantage is involved with a drawback from the perspective of an experimenter's working in situ: it requires big efforts to maintain steady-state flowing conditions over a long period of time and to take valuable measurements of high quality.

The hydraulic conductivity is the most important hydraulic parameter in groundwater simulation. It describes the ease with which water can move through the pore space of the fully water-saturated porous medium. Under the assumption that water is incompressible (its density is constant), the basic law that governs water flow

- in the saturated zone
- under isothermal conditions
- at macroscopic level (laboratory or field scale)
- in an isotropic porous medium[1]

is given by the following version of **Darcy's Law**[2]:

$$\vec{q} = -K\ \nabla\phi \tag{2.2}$$

where the specific discharge $\vec{q}\,[m/s]$ is measured as the volume of water passing through a unit area of porous medium cross-section per unit time. $\phi[m]$ the **hydraulic head** (piezometric head), defined as

$$\phi = z + \frac{p}{\rho_w g} \tag{2.3}$$

where $z[m]$ is the elevation of the point (above some reference level) at which the measurement of ϕ is actually taken, $p[kg/(ms^2)]$ is the water pressure, $\rho_w[kg/m^3]$ is the mass density of water and $g[m/s^2]$ is the gravitational acceleration. The coefficient of proportionality $K[m/s]$ is the **hydraulic conductivity**. Whereas K depends on both the solid matrix and the fluid properties, the **intrinsic permeability** κ is a property of the pore geometry alone. Their relation is given by

$$K = \kappa\,\frac{\rho_w g}{\mu_w} \tag{2.4}$$

where $g \approx 9.81[m/s^2]$. At a reference temperature of 15°C, the density of water is $\rho_w = 999.1[kg/m^3]$ and the dynamic viscosity of water is $\mu_w = 1.139 \cdot 10^{-3}[kg/(ms)]$. Hence, an alternative formulation of Darcy's Law reads:

$$\vec{q} = -\frac{\kappa}{\mu_w}\,\nabla\left(p + \rho_w g z\right) \tag{2.5}$$

Tables G.1 and G.2 in the Appendix show typical values of the porosity and the hydraulic conductivity for different natural materials.

[1] At any given point, the hydraulic conductivity is not dependent on the direction. By contrast, anisotropic porous media are modeled by a second rank tensor of hydraulic conductivity.

[2] originally (1865) suggested on the basis of sand box column experiments and later recognized as a motion equation that can be derived from the microscopic differential balance equation of linear momentum by means of volume averaging, homogenization or mixture theory [Bear and Cheng, 2010]

2.2 Spatial description of the hydraulic conductivity

Throughout this work, the computational domain $\Omega \subset \mathbb{R}^d$ ($d \in \{2, 3\}$) is a rectangular cuboid. This is adequate for laboratory experiments with sandboxes. For simulations in field scale, we must work with the assumption that Ω is merely a block truncated out of the physical domain (the aquifer).

2.2.1 Stochastic model

We assume that the natural logarithm of the hydraulic conductivity

$$Y(\vec{x}) = \ln(K(\vec{x})) \tag{2.6}$$

is a random field with normal distribution[1], consisting of a trend function $\mu(\vec{x})$ and a zero-mean fluctuation $Y'(\vec{x})$.

$$Y(\vec{x}) = \mu(\vec{x}) + Y'(\vec{x}) \tag{2.7}$$

The linear trend model

$$\mu(\vec{x}) = \sum_{k=1}^{N_\beta} \beta_k X_k(\vec{x}) \tag{2.8}$$

can be used to describe a **zonation** of the domain Ω containing N_β zones provided that this knowledge is available: $X_k(\vec{x})$ is the indicator function for the k-th zone, inside which the log hydraulic conductivity fluctuates about a mean value β_k. It is important that the number of zones N_β is much smaller than the number of measurements M [Kitanidis, 1997]. The **drift** or **trend coefficient** vector $\boldsymbol{\beta} = (\beta_1, ..., \beta_{N_\beta})^T$ itself may be uncertain. The spatially correlated random fluctuation $Y'(\vec{x})$ is represented by the two-point covariance function R (D.7) which is parameterized by one of the variogram models (D.8)-(D.10). We may summarize the field variance σ_Y^2 and the correlation lengths ℓ_k into a vector of **structural parameters** $\boldsymbol{\theta} := (\sigma_Y^2, \ell_1, ..., \ell_d)^T$ and write

$$E[Y'(\vec{x}), Y'(\vec{y})] = R(\vec{x}, \vec{y} \big| \boldsymbol{\theta}) \tag{2.9}$$

as in [Kitanidis, 1993], indicating that $\boldsymbol{\theta}$ is known.

[1] The hydraulic conductivity is not necessarily log-normally distributed, but according to [Domenico and Schwartz, 1998], the histograms of values taken in many studies assume the shape of the bell curve.

2.2.2 Discretization of the log hydraulic conductivity Y

Approximating the log conductivity field $Y(\vec{x})$ by a cell-wise constant function on a structured mesh

$$\mathcal{T}_h = \bigcup_{j=0}^{N-1} t_j \tag{2.10}$$

with $N = n_1 \times ... \times n_d$ disjoint cuboidal and axis-parallel cells (with maximal edge length h) forming a partitioning of the domain Ω, $Y(\vec{x})$ can be represented as a d-dimensional array with exactly N entries. These entries can be arranged (in lexicographical order) to form a random vector[1]

$$\boldsymbol{y} = \big(Y(\vec{x}_0), ..., Y(\vec{x}_{N-1})\big)^T \in \mathbb{R}^N \tag{2.11}$$

where the $\vec{x}_j$ is the cell center of t_j. The discrete form of (2.7) then reads

$$\boldsymbol{y} = \mathcal{X}\boldsymbol{\beta} + \boldsymbol{y}' \tag{2.12}$$

where $\mathcal{X} \in \mathbb{R}^{N \times N_\beta}$ is the zonation matrix, $\boldsymbol{\beta} \in \mathbb{R}^{N_\beta}$ is the vector of trend coefficients and $\boldsymbol{y}' \sim \mathcal{N}_N(\boldsymbol{0}, \mathcal{R}_{yy})$ is the random fluctuation around the mean $\mathcal{X}\boldsymbol{\beta}$ with covariance $\mathcal{R}_{yy}$. This is equivalent to the statement

$$\boldsymbol{y} \sim \mathcal{N}_N(\mathcal{X}\boldsymbol{\beta}, \mathcal{R}_{yy}). \tag{2.13}$$

The covariance matrix $\mathcal{R}_{yy}$ has the entries

$$\begin{aligned}(\mathcal{R}_{yy})_{jk} &= R(\vec{x}_j, \vec{x}_k \big| \boldsymbol{\theta}) \\ &= \sigma_Y^2 - \gamma\big(\,\tilde{h}(\,\vec{x}_j,\,\vec{x}_k\,)\,\big)\end{aligned} \tag{2.14}$$

defined by the field variance σ_Y^2 and the semi-variogram function γ (D.7). The distance $\tilde{h}$ is scaled by the correlation lengths $\ell_1, ..., \ell_d$ as in (D.5). $\mathcal{R}_{yy}$ is a symmetric, positive semi-definite matrix, which has Toeplitz structure if the domain is discretized by a structured equidistant mesh. It can be fully characterized by its first column

$$(r_0, ..., r_{N-1})^T \quad \text{where} \quad r_j = R(\vec{x}_j, \vec{x}_0 \big| \boldsymbol{\theta}). \tag{2.15}$$

[1]In the following, we do not make a distinction between a random vector $\boldsymbol{Y}$ and its realization $\boldsymbol{y}$, c.f. Appendix B.

2.3 Random field generation algorithm

Theorem A.2, Lemma C.1 and Theorem C.1(a) (or Theorem C.2 for the degenerate case) suggest a straight forward, but inefficient method to generate realizations of a multivariate Gaussian distributed random vector $\boldsymbol{y}' \sim \mathcal{N}_N(\mathbf{0}, \mathcal{R}_{yy})$ from a white noise vector $\boldsymbol{x}' \sim \mathcal{N}_N(\mathbf{0}, \mathbf{Id})$.

Algorithm 2.1 Random Field Generation (by Cholesky factorization)

```
**Input:** Covariance matrix $\mathcal{R}_{yy} \in \mathbb{R}^{N \times N}$
1.) Generate i.i.d. variables $x_1, ..., x_N \sim \mathcal{N}(0,1)$ and set $\boldsymbol{x}' = (x_1, ..., x_N)^T$.
2.) Compute a Cholesky factorization $\mathcal{R}_{yy} = \mathcal{L}\mathcal{L}^T$.
3.) Set $\boldsymbol{y}' = \mathcal{L}\boldsymbol{x}'$.
```

The Cholesky factorization, which is the most cost-intensive part in this algorithm, generally requires $O(N^3)$ floating point operations. For Toeplitz matrices, the complexity can be reduced to $O(N^2)$ [Kailath, 1986]. For practical applications, where N can easily exceed 10^6, this is still too much. Dietrich and Newsam [1993, 1997] have presented a method to generate realizations $\boldsymbol{y}'$ which requires only $O(N \log N)$ operations by

1. embedding the Toeplitz matrix $\mathcal{R}_{yy} \in \mathbb{R}^{N \times N}$ into a non-negative definite circulant matrix $\mathcal{S}_{yy} \in \mathbb{R}^{N' \times N'}$ (with $N' = 2M'$ and $M' \geq N - 1$ for $d = 1$),

2. applying a two-step forward DFT to get an efficient matrix-vector multiplication $\boldsymbol{\eta} = \mathcal{B}\boldsymbol{\chi}$, where $\mathcal{B} \in \mathbb{R}^{N' \times N'}$ is a square root factor of $\mathcal{S}_{yy}$ and $\boldsymbol{\chi} \in \mathbb{C}^{N'}$ is a complex white noise vector,

3. and finally stripping off an N–dimensional sub-vector of $\boldsymbol{\eta}$ that has the desired distribution $\mathcal{N}_N(\mathbf{0}, \mathcal{R}_{yy})$.

The method works on a rectangular cuboidal domain $\Omega \subset \mathbb{R}^d$, discretized by a structured equidistant mesh $\mathcal{T}_h \subset \Omega$. Given the domain lengths L_k and the numbers of mesh cells n_k in each dimension $k = 1, ..., d$, the total number of mesh cells is $N = n_1 \cdots n_d$ and the mesh-sizes are $\Delta x_k = L_k / n_k$. For the most important variogram models in groundwater simulation, Dietrich and Newsam have listed various conditions for the choice of correlation lengths, domain lengths and resolution of the discretization under which the minimal circulant embedding ($M' = N - 1$ for $d = 1$) produces non-negative definite $\mathcal{S}_{yy}$. These parameters should be chosen in such a way that the correlation lengths of the random field do not exceed the domain lengths ($\ell_k \ll L_k$) and are resolved well enough by the mesh-sizes ($\ell_k \gg \Delta x_k$).

Algorithm 2.2 Random Field Generation (by Dietrich and Newsam)

Input: Domain extensions $\{L_k\}_{1,...,d}$, mesh cell numbers $N = n_1 \cdots n_d$, zonation matrix $\mathcal{X}$ and trend coefficients $\boldsymbol{\beta} \in \mathbb{R}^{N \times N_\beta}$, variogram model $\gamma(\tilde{h})$ with variance σ_Y^2 and correlation lengths $\ell_1, ..., \ell_d$.

(1) **Domain extension:** Define the extended mesh $\mathcal{T}'_h$ with the same mesh-sizes Δx_k as $\mathcal{T}_h$ such that $n'_k \geq 2n_k - 2$ and set $N' = n'_1 \cdots n'_d$. The extended domain Ω' is chosen such that $L'_k = n'_k \Delta x_k$.

(2) **Periodic circulant embedding:** Compute the first column $\boldsymbol{s} = (s_1, ..., s_{N'})^T$ of a circulant matrix $\mathcal{S}_{yy} \in \mathbb{R}^{N' \times N'}$ as follows:

$d = 1$: Define $\boldsymbol{r} = (r_0, ..., r_{N-1})^T$ as in (2.15).

$$\begin{aligned} s_k &= r_{k-1}, & k &= 1, ..., n_1 \\ s_{N'+2-k} &= r_{k-1}, & k &= 2, ..., n_1 - 1 \end{aligned} \tag{2.16}$$

A minimal periodic embedding ($N' = 2n_1 - 2$) is illustrated in Figure 2.2.

$d > 1$: Let $\vec{x}' \in \Omega'$ be the center of a mesh cell in $\mathcal{T}'_h$ and let $l \in \{1, ..., N'\}$ be the global (lexicographical) index of this cell. Then,

$$s_l := R(\vec{x}, \vec{0} | \boldsymbol{\theta}) \tag{2.17}$$

where $\vec{x} \in \Omega$ has the coordinates

$$x_k = \min\{\ x'_k,\ L'_k - x'_k\ \} \text{ for } k = 1, ..., d \tag{2.18}$$

For $d = 2$, this embedding in two directions is illustrated in Figure 2.3(a).

(3) Compute the vector of eigenvalues of $\mathcal{S}_{yy}$ as in (A.30) using the forward DFT:

$$\boldsymbol{\lambda} = \sqrt{N'} \cdot \mathcal{F}_{N'} \cdot \boldsymbol{s}. \tag{2.19}$$

(4) Generate a complex white noise vector $\boldsymbol{\chi} = \boldsymbol{\mu} + i\boldsymbol{\nu} \in \mathbb{C}^{N'}$ with independent and real white noise vectors $\boldsymbol{\mu}, \boldsymbol{\nu} \sim \mathcal{N}_{N'}(\mathbf{0}, \mathbf{Id})$.

(5) Compute the vector $\tilde{\boldsymbol{\chi}} \in \mathbb{C}^{N'}$ with entries

$$\tilde{\chi}_k = \chi_k \cdot \sqrt{\lambda_k} \qquad k = 1, ..., N' \tag{2.20}$$

and apply the forward DFT on $\tilde{\boldsymbol{\chi}}$ to get

$$\boldsymbol{\eta} = \mathcal{F}_{N'} \cdot \tilde{\boldsymbol{\chi}}. \tag{2.21}$$

(6) Define the vector $\boldsymbol{y}' \in \mathbb{R}^N$ with entries $y'_k = \mathrm{Re}(\eta_k)$ for $k = 1, ..., N$.

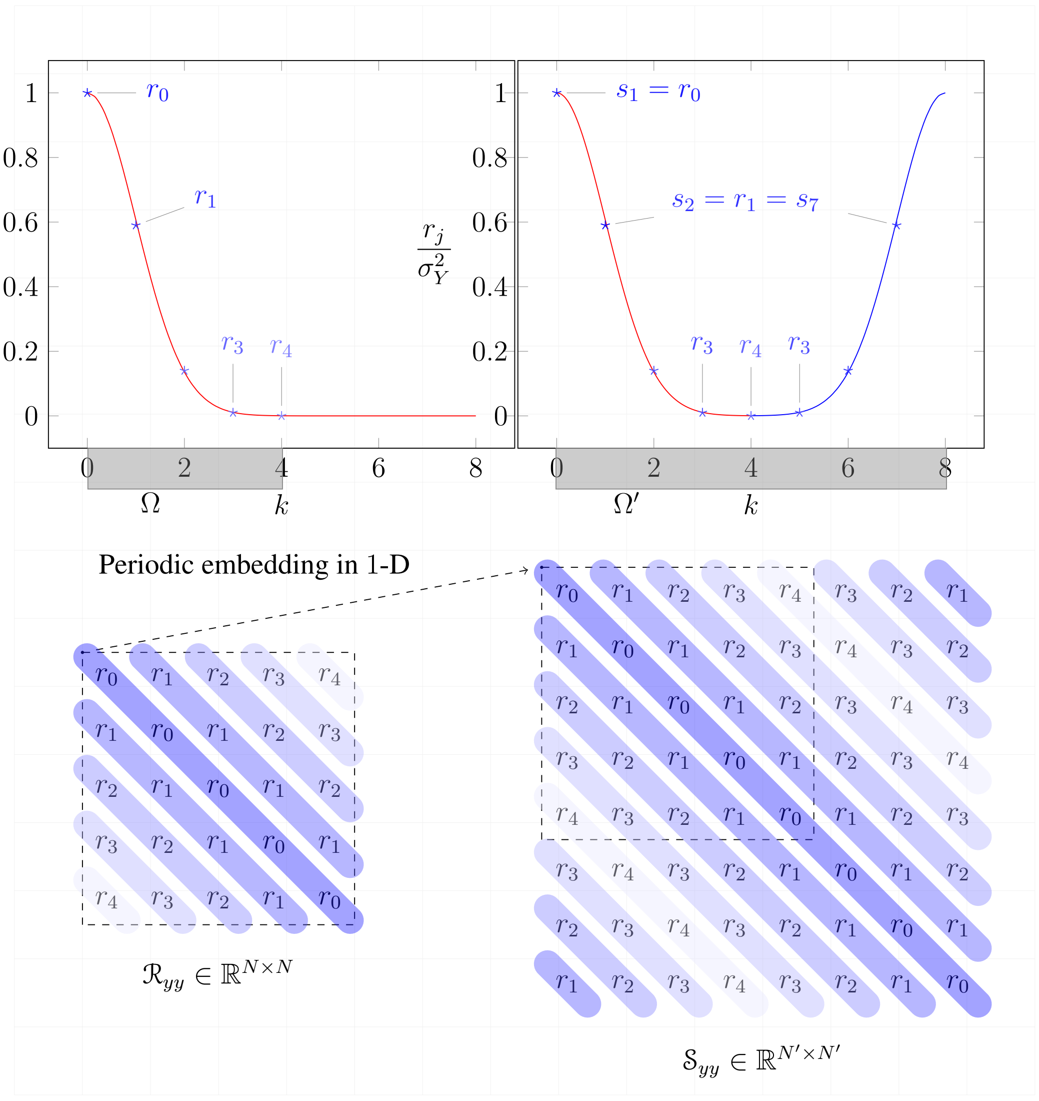
(7) Compute

$$\boldsymbol{y} = \mathcal{X}\boldsymbol{\beta} + \boldsymbol{y}'. \tag{2.22}$$

Output: $\boldsymbol{y} \sim \mathcal{N}_N(\mathcal{X}\boldsymbol{\beta}, \mathcal{R}_{yy})$

Remark 2.1. *For $d = 2$, $\mathcal{R}_{yy}$ is a block-Toeplitz matrix with Toeplitz blocks. For $d = 3$, $\mathcal{R}_{yy}$ is a block-Toeplitz matrix with block-Toeplitz blocks. Both of them are Toeplitz matrices. Likewise, the circulant matrix $\mathcal{S}_{yy}$ has a block circulant structure for $d > 1$.*

Remark 2.2. *If the embedding is not minimal, the entries $s_{n_1+1}, \ldots, s_{N'+2-n_1}$ may be chosen arbitrarily, most naturally by continuation of (2.15) to the extended domain such that the values of s_k are symmetric with respect to the mirror plane of the extended cuboid Ω' (Figure 2.2).*



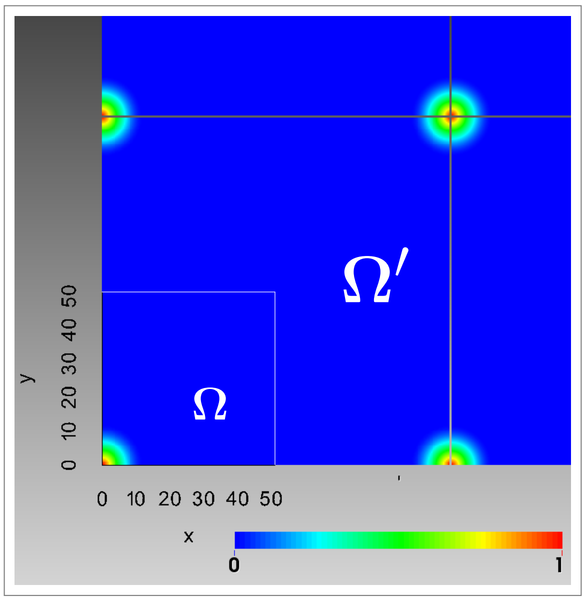
r_0
r_1
r_3
r_4
$\frac{r_j}{\sigma_Y^2}$
Ω
k
$s_1 = r_0$
$s_2 = r_1 = s_7$
r_3
r_4
r_3
Ω'
k
Periodic embedding in 1-D
$\mathcal{R}_{yy} \in \mathbb{R}^{N \times N}$
$\mathcal{S}_{yy} \in \mathbb{R}^{N' \times N'}$

Figure 2.2: Embedding of a domain $\Omega = [0, 4]$ into the extended periodic domain $\Omega' = [0, 8)$ on an equidistant mesh $x_k = x_0 + k\Delta x$ for the covariance function given by $r_k = \sigma_Y^2 - \gamma(x_k - x_0)$ where $\gamma(h) = \sigma_Y^2 \left(1 - \exp\left(-h^2/\ell^2\right)\right)$ is the Gaussian model (with correlation $\ell = \sqrt{2}$).

Remark 2.3. *Equations (2.20) and (2.21) of Algorithm 2.2 implement the square root of $\mathcal{S}_{yy}$ as defined in (A.32). Using Theorem C.2, it can be shown that both real and imaginary parts of $\boldsymbol{\eta} \in \mathbb{C}^{N'}$ have the desired distribution $\mathcal{N}(\mathbf{0}, \mathcal{S}_{yy})$. Due to the periodic embedding, any $N \times N$ block along the main diagonal of $\mathcal{S}_{yy}$ is a copy of $\mathcal{R}_{yy}$ (Figure 2.2). Therefore, in step 6, actually any consecutive N entries of Re($\boldsymbol{\eta}$) or Im($\boldsymbol{\eta}$) can be chosen as a random field $\boldsymbol{y}' \in \mathbb{R}^N$ with the desired distribution $\mathcal{N}_N(\mathbf{0}, \mathcal{R}_{yy})$.*

(a) periodicity of the mirrored covariance function

Ω : embedded domain
Ω': embedding domain



(b) periodicity of the extended field $Re(\boldsymbol{\eta})$ in Ω'

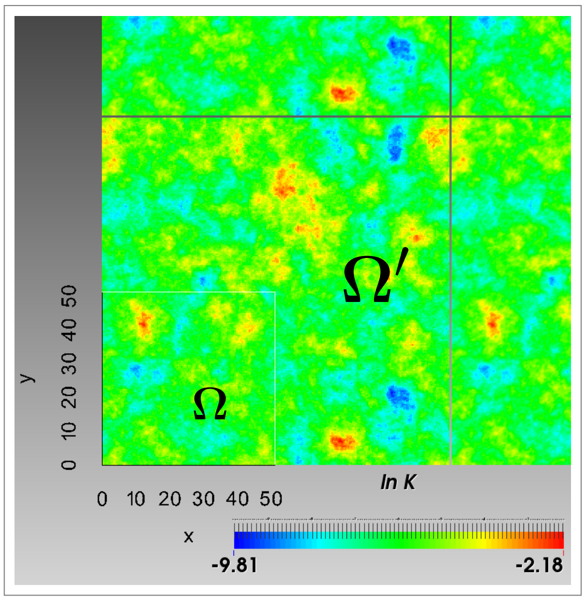
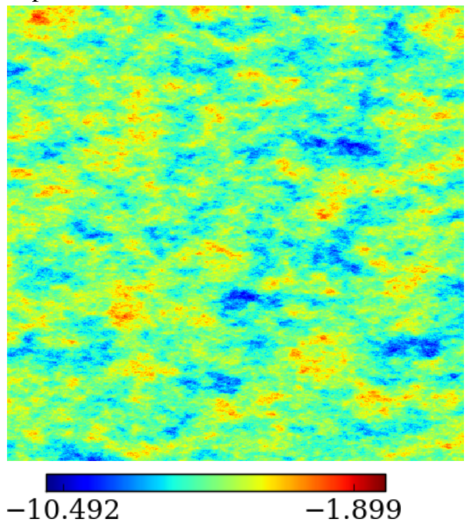


Figure 2.3: Periodic embedding in 2-D: $\Omega \subset \Omega'$.

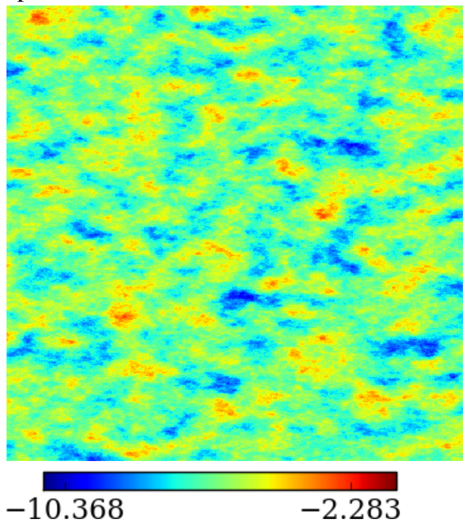
domain extensions $[m]$	$(L_x, L_y) = (51.2, 51.2)$
correlation lengths $[m]$	$(\ell_x, \ell_y) = (2, 1)$
embedded mesh resolution	$(n_x, n_y) = (512, 512) \Rightarrow N = 262,144$
embedding mesh resolution	$(n'_x, n'_y) = (1024, 1024) \Rightarrow N' = 1,048,576$
trend coefficient	$N_\beta = 1$, $\beta = -6.100$ with uncertainty *Var*$[\beta] = 0.0005$
variance of the field	$\sigma^2 = 1.000$

exponential



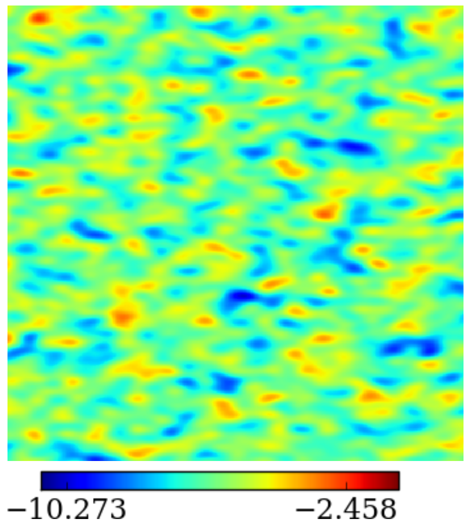
measured $(\beta, \sigma^2) = (-6.109, 0.961)$

spherical



measured $(\beta, \sigma^2) = (-6.104, 0.979)$

Gaussian



measured $(\beta, \sigma^2) = (-6.104, 0.967)$

all eigenvalues of $\mathcal{S}_{yy} \in \mathbb{R}^{N' \times N'}$ strictly positive
$\Rightarrow \mathcal{R}_{yy} \in \mathbb{R}^{N \times N}$ symmetric positive definite

$\mathcal{S}_{yy}$ symm. non-neg. def.
with $33,815$ pos. eigenvalues

Figure 2.4: Sampling random fields $Y = \ln K$ for different variogram models using the same random seed and the same correlation lengths (mesh resolution: 512×512). The exponential and spherical variogram functions yield almost identical fields with jumps between neighboring elements whereas all transitions are smooth in the Gaussian model (see also Figure D.1 in the Appendix).

Chapter 3

Inversion

3.1 Deterministic approach

Let $\Omega \subset \mathbb{R}^d$ be a rectangular cuboid. In the following, let $\mathcal{H}_{\text{D}}$ denote the "data space", $\mathcal{H}_{\text{P}}$ be the "parameter space" and $\mathcal{H}_{\text{U}}$ be the "state space". Measurement data[1] at discrete locations $\{\vec{x}_1, ..., \vec{x}_M\}$ yield $\mathcal{H}_{\text{D}} \subset \mathbb{R}^M$. A discretization of the parameter space as in §2.2.2 yields $\mathcal{H}_{\text{P}} \subset \mathbb{R}^{N+N_\beta}$. The state space $\mathcal{H}_{\text{U}}$ is typically the function space $H^1(\Omega)$.

In classical inverse problems, given a vector of observations $\boldsymbol{d}_{\text{obs}} \in \mathcal{H}_{\text{D}}$, the task is to find (an estimate of) the unknown parameter field $\boldsymbol{p} \in \mathcal{H}_{\text{P}}$ such that the model equation

$$\boldsymbol{d}_{\text{obs}} = \boldsymbol{F}(\boldsymbol{p}) \tag{3.1}$$

is fulfilled. Hereby, the **model function**

$$\begin{array}{rccc} \boldsymbol{F}: & \mathcal{H}_{\text{P}} & \longrightarrow & \mathcal{H}_{\text{D}} \\ & \boldsymbol{p} & \mapsto & \boldsymbol{d}_{\text{obs}} \end{array} \tag{3.2}$$

can be stated more precisely as a composition $\boldsymbol{F} = \boldsymbol{g} \circ u$ of a **solution operator** of the forward model for the measured quantity

$$\begin{array}{rccc} u: & \mathcal{H}_{\text{P}} & \longrightarrow & \mathcal{H}_{\text{U}} \\ & \boldsymbol{p} & \mapsto & u(\boldsymbol{p}) \end{array} \tag{3.3}$$

and an **observation operator** evaluating the predicted model outcome at discrete points:

$$\begin{array}{rccc} \boldsymbol{g}: & \mathcal{H}_{\text{U}} & \longrightarrow & \mathcal{H}_{\text{D}} \\ & u & \mapsto & \left(u|_{\vec{x}=\vec{x}_1}, ..., u|_{\vec{x}=\vec{x}_M}\right)^T . \end{array} \tag{3.4}$$

[1]Note that different types of measurements may be taken at a single location. In this case, M is not the total number measurement locations, but the total number of all measurements.

Only if $\boldsymbol{F}$ is injective, we can construct an inverse operator $\boldsymbol{F}^{-1} : \boldsymbol{F}[\mathcal{H}_{\mathrm{P}}] \to \mathcal{H}_{\mathrm{P}}$. Solving an inverse problem practically means to find an approximation of $\boldsymbol{F}^{-1}$ mapping the measured data $\boldsymbol{d}_{\mathrm{obs}}$ to a uniquely defined $\boldsymbol{p} \in \mathcal{H}_{\mathrm{P}}$.

The type of parameter estimation problems we want to solve is in general ill-posed and non-linear even if the underlying forward equation for a known parameter is linear (c.f. [Engl et al., 2000], §1.6).

For a linear model, $\boldsymbol{F}$ can be represented by a constant matrix $\mathcal{A} \in \mathbb{R}^{M\times(N+N_\beta)}$. If the number of measuring points M is much smaller than the number of parameters N, we face an under-determined linear system

$$\boldsymbol{d}_{\mathrm{obs}} = \mathcal{A}\boldsymbol{p}\,. \tag{3.5}$$

A practical solution may be defined by the least squares solution

$$\min_{\boldsymbol{p}\in\mathcal{H}_{\mathrm{P}}} \mathscr{J}(\boldsymbol{p}) := \frac{1}{2}\|\boldsymbol{d}_{\mathrm{obs}} - \mathcal{A}\boldsymbol{p}\|_2^2\,. \tag{3.6}$$

The best-approximate least squares solution is

$$\boldsymbol{p} = \mathcal{A}^{+}\boldsymbol{d}_{\mathrm{obs}} \tag{3.7}$$

where the Moore-Penrose generalized inverse $\mathcal{A}^{+}$ can be computed through a singular value decomposition of $\mathcal{A}$ (c.f.[Stoer and Bulirsch, 1996], §6.4). However, small singular values of $\mathcal{A}$ tend to zero as N increases and small error components δd_k in the data $\boldsymbol{d}_{\mathrm{obs}}$ corresponding to small singular values σ_k can be amplified arbitrarily much, namely with $1/\sigma_k$ (c.f. [Engl et al., 2000], §2.2), violating the continuity property for well-posedness.

Regularization is a concept that replaces an ill-posed problem by a series of well-posed or better-posed problems such that their solutions tend to $\boldsymbol{p}$ as the perturbations δd_k tend to zero (c.f. [Engl et al., 2000], §3.1). The most popular regularization technique is the well-known **Tikhonov-Phillips regularization method** where (3.6) is replaced by

$$\min_{\boldsymbol{p}\in\mathcal{H}_{\mathrm{P}}} \mathscr{J}_\alpha(\boldsymbol{p}) := \frac{1}{2}\|\boldsymbol{d}_{\mathrm{obs}} - \mathcal{A}\boldsymbol{p}\|_{\mathcal{H}_{\mathrm{D}}}^2 + \frac{\alpha}{2}\|\boldsymbol{p} - \boldsymbol{p}^\star\|_{\mathcal{H}_{\mathrm{P}}}^2. \tag{3.8}$$

In the first term, the norm $\|.\|_{\mathcal{H}_{\mathrm{D}}}$ may incorporate weighting of model and measurement errors (see §3.3.2). The second term restricts the space in which the unknown parameters are sought by imposing **prior** assumptions in the form of $\boldsymbol{p}^\star$ which is independent of the measurements used for the current inversion. $\boldsymbol{p}^\star$ can be an educated guess from experts or the result of a preceding estimation. Deviations from $\boldsymbol{p}^\star$, measured in the norm $\|.\|_{\mathcal{H}_{\mathrm{P}}}$, are penalized. The importance of this term is weighted by the regularization parameter $\alpha > 0$. This new minimization problem is a compromise between data fitting and keeping the penalization term small, i.e. enforcing stability (c.f. [Engl et al., 2000],

§5.1). A well-posed problem can be **ill-conditioned** (if $\kappa(\mathcal{A}) = \|\mathcal{A}\|\|\mathcal{A}^+\| \gg 1$) which makes its iterative solution non-robust. Therefore, using a stable algorithm increases the chance of finding a numerical solution.

This regularization concept can directly be applied to the general case with non-linear $\boldsymbol{F}$:

$$\min_{\boldsymbol{p}\in\mathcal{H}_{\mathrm{P}}} \mathscr{J}_\alpha(\boldsymbol{p}) := \frac{1}{2}\|\boldsymbol{d}_{\mathrm{obs}} - \boldsymbol{F}(\boldsymbol{p})\|^2_{\mathcal{H}_{\mathrm{D}}} + \frac{\alpha}{2}\|\boldsymbol{p} - \boldsymbol{p}^\star\|^2_{\mathcal{H}_{\mathrm{P}}}. \tag{3.9}$$

An iterative scheme based on the successive linearization of $\boldsymbol{F}$ (c.f. [Kaipio and Somersalo, 2005], §2.3), the non-linear Landweber iteration or Newton-type schemes may be used to solve this problem ([Engl et al., 2000], §11). Thus, the main objective of regularization theory is to find appropriate norms in the spaces $\mathcal{H}_{\mathrm{D}}$ and $\mathcal{H}_{\mathrm{P}}$ and the regularization parameter α such that the solution best fits the prior information $\boldsymbol{p}^\star$ [Bal, 2012]. We will see in the next section that the Bayesian approach devises all these ingredients.

3.2 Statistical approaches

The main advantage in the use of stochastic methods is that the unknown parameters *and their uncertainty* can be estimated at the same time [Sun, 1994]. In statistical inference, the vector of observations $\boldsymbol{d}_{\mathrm{obs}} \in \mathbb{R}^M$ is regarded as a realization of random vector $\boldsymbol{d}$ and we are interested in the probability distribution (described by some density $\varrho = \varrho_d$) that this random vector is drawn from. Thereby, we may distinguish between **point estimates** (*What are the most probable values of the parameters* $\mathbf{p}$*?*) and **interval estimates** (*In what interval are the values of* $\mathbf{p}$ *with* 90% *probability?*)

In the frequentist statistics, the probability of an event is defined as the limit of its relative frequency as the number of repeated trials tends to infinity. In the frequentist inference, the unknown parameters $\boldsymbol{p}$ are treated as deterministic. One of the most popular point estimators in statistics is the **maximum likelihood (MLE)** estimator $\boldsymbol{p}_{\mathrm{MLE}}$. It answers the question *"Which values of* $\mathbf{p}$ *are most likely to reproduce the vector of observations* $\mathbf{d}_{obs}$*?"*. Formally, it is defined as the solution of the optimization problem

$$\max_{\boldsymbol{p}\in\mathbb{R}^N} \varrho(\boldsymbol{d}|\boldsymbol{p}). \tag{3.10}$$

Note that in the MLE approach, only $\boldsymbol{d}$ is a random quantity for which a probability density is defined. Often, it corresponds to solving the classical inverse problem without regularization and is therefore ill-conditioned (cf. [Kaipio and Somersalo, 2005], §3.1.1).

In the Bayesian inference, both $\boldsymbol{d}$ and $\boldsymbol{p}$ are treated as random vectors. The marginal density $\varrho(\boldsymbol{p})$, which is independent of $\boldsymbol{d}$, is called the **prior density**. The conditional density $\varrho(\boldsymbol{d}|\boldsymbol{p})$ is called the **likelihood function** following (3.10) and the conditional density

$$\varrho(\boldsymbol{p}|\boldsymbol{d}) = \frac{\varrho(\boldsymbol{p}) \cdot \varrho(\boldsymbol{d}|\boldsymbol{p})}{\varrho(\boldsymbol{d})} \tag{3.11}$$

according to Bayes' formula (B.17) is called the **posteriori density**. In the Bayesian context, this is the solution of the inversion problem. Point estimates can be drawn from this distribution, for example the **maximum a posteriori (MAP)** estimator[1] $\boldsymbol{p}_{\text{MAP}}$, which solves the optimization problem

$$\boldsymbol{p}_{\text{MAP}} = \arg\max_{\boldsymbol{p}\in\mathbb{R}^N} \varrho(\boldsymbol{p}|\boldsymbol{d}), \tag{3.12}$$

or the **conditional mean** $\boldsymbol{p}_{\text{CM}}$, defined as

$$\boldsymbol{p}_{\text{CM}} = E[\boldsymbol{p}|\boldsymbol{d}] = \begin{bmatrix} \int\limits_{\mathbb{R}^N} y_0 \cdot \varrho(\boldsymbol{p}|\boldsymbol{d})\, \boldsymbol{dy} \\ \vdots \\ \int\limits_{\mathbb{R}^N} y_{N-1} \cdot \varrho(\boldsymbol{p}|\boldsymbol{d})\, \boldsymbol{dy} \end{bmatrix}. \tag{3.13}$$

Estimating the conditional mean $\boldsymbol{p}_{\text{CM}}$ requires the evaluation of high-dimensional integrals. Usually, Markov Chain Monte Carlo (MCMC) methods such as the Metropolis-Hastings algorithm are constructed to sample from the posterior distribution. For a useful characterization of the posterior distribution of an inverse problem, a high number of such samples is needed ([Kaipio and Somersalo, 2005], §3.6).

Computing the MAP estimate $\boldsymbol{p}_{\text{MAP}}$ leads to an optimization problem. For its solution, a vast repertoire of gradient based methods can be used (see e.g. [Nocedal and Wright, 2006]).

In general, the MAP estimate and the conditional mean differ. There is a constant debate on which of these two estimates are to be preferred [Burger and Lucka, 2014]. A complete description of a probability distribution requires the specification of the mean and the covariance matrix, but the MAP estimate corresponds to the solution given by the classical Tikhonov regularization, as we will see later in §3.3.4.

[Kitanidis, 1995] suggests a combination of sampling and optimization that generates a series of conditional realizations out of an unconditional realization. It is called the method of "smallest possible modification" that allows reproduction of data within their measurement error.

In [Kitanidis, 1995], the quasi-linear geostatistical theory is developed to compute an MLE estimator for the structural parameters $\boldsymbol{\theta}$ based on the measurements $\boldsymbol{d}$. A

[1] Be careful: It is very easy to get confused with the terms MAP and MLE. In [Kitanidis, 1996], the author states that what he calls MAP is called MLE in [Carrera and Neuman, 1986].

lower bound for the mean square estimation error of $\boldsymbol{\theta}$ is given by the inverse of the Fisher information matrix (Cramer-Rao inequality). This $\boldsymbol{\theta}$-fitting step is required to provide prior information to the Bayesian framework. The theory is derived for the case of cokriging with unknown trend coefficient vector $\boldsymbol{\beta}$. In [Nowak, 2005; Nowak and Cirpka, 2004], the theory is extended to the case of cokriging with uncertain trend.

Under the assumption that the structural parameters

$$\boldsymbol{\theta} = (\sigma_Y^2, \ell_1, ..., \ell_d)^T \tag{3.14}$$

(field variance and correlation lengths) are known (and not part of the estimation), we seek for the MAP estimator (for the trend $\boldsymbol{\beta}$ and the log conductivity $\boldsymbol{y}$) and its uncertainty.

3.3 Bayesian inference

This section is a summary of the theory developed in the works of Kitanidis [1986, 1996] and Nowak [2005]. All the quantities involved (parameters, trend coefficients, data and measurement errors) are considered to be random variables. Furthermore, we assume that their probability distributions can be represented by a non-degenerate multivariate Gaussian density function with symmetric positive definite covariance matrix.

According to (3.11), the task is

1. to construct a prior distribution $\varrho(\boldsymbol{p})$ that reflects all available information of the unknown parameters $\boldsymbol{p}$ *prior* to any measurements,
2. to construct a likelihood function $\varrho(\boldsymbol{d}|\boldsymbol{p})$ that describes the uncertainty in the relation between the measurements $\boldsymbol{d}$ and the model function in (1.1),
3. to develop a method to compute the posterior distribution $\varrho(\boldsymbol{p}|\boldsymbol{d})$.

3.3.1 Uncertain prior information

The list of prior information available to us comprises of the correlation lengths ℓ_k (D.5), the variance σ^2 (D.7) and the choice of an appropriate variogram model. Along with the domain lengths L_k and mesh-sizes Δx_k, this handful of numbers is sufficient to identify the covariance matrix $\mathcal{R}_{yy}$ introduced in (2.14).

A general framework for *cokriging with uncertain mean* ($\mathcal{X}\boldsymbol{\beta}$ in (2.12)), which includes the two cases "cokriging with known mean" and "cokriging with unknown mean" as special cases, was developed by Nowak [2005], §6.1.1. The vector of trend coefficients $\boldsymbol{\beta} \in \mathbb{R}^{N_\beta}$ itself is considered to be a random vector and part of the estimation process. This means that we want to estimate

$$\boldsymbol{p} := (\boldsymbol{y}, \boldsymbol{\beta})^T, \tag{3.15}$$

where we identify $\boldsymbol{y} = \mathcal{X}\boldsymbol{\beta} + \boldsymbol{y}'$ (2.12) with the random field $\boldsymbol{y}|\boldsymbol{\beta}$ with conditional mean $\mathcal{X}\boldsymbol{\beta}$ and conditional covariance $\mathcal{C}_{yy|\beta} \equiv \mathcal{R}_{yy}$:

$$\boldsymbol{y} \equiv \boldsymbol{y}|\boldsymbol{\beta} \sim \mathcal{N}_N(\mathcal{X}\boldsymbol{\beta}, \mathcal{R}_{yy}) \tag{3.16}$$

$$\Leftrightarrow \quad \varrho(\boldsymbol{y}|\boldsymbol{\beta}) \propto \exp\left(\frac{1}{2}(\boldsymbol{y} - \mathcal{X}\boldsymbol{\beta})^T \mathcal{R}_{yy}^{-1}(\boldsymbol{y} - \mathcal{X}\boldsymbol{\beta})\right).$$

We assume that

$$\boldsymbol{\beta} \sim \mathcal{N}_{N_\beta}(\boldsymbol{\beta}^\star, \mathcal{C}_{\beta\beta}) \tag{3.17}$$

$$\Leftrightarrow \quad \varrho(\boldsymbol{\beta}) \propto \exp\left(\frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta}^\star)^T \mathcal{C}_{\beta\beta}^{-1}(\boldsymbol{\beta} - \boldsymbol{\beta}^\star)\right)$$

where $\boldsymbol{\beta}^\star$ is the deterministic trend, "polluted" with some predefined uncertainty $\mathcal{C}_{\beta\beta}$ (measurement errors). According to (B.14), the joint distribution density is given by

$$\varrho(\boldsymbol{p}) = \varrho(\boldsymbol{y}, \boldsymbol{\beta}) = \varrho(\boldsymbol{y}|\boldsymbol{\beta})\varrho(\boldsymbol{\beta}). \tag{3.18}$$

3.3.2 Likelihood function

The model equation (3.1) may be written as

$$\boldsymbol{d}_{\text{obs}} = \boldsymbol{F}(\boldsymbol{p}) := \boldsymbol{f}(\boldsymbol{y}) := \boldsymbol{g} \circ u(\boldsymbol{y}) \tag{3.19}$$

if we keep in mind that $\boldsymbol{y} \equiv \boldsymbol{y}|\boldsymbol{\beta}$. Equation (3.19) would hold only if the physical model (represented by the solution operator u) were a perfect model of reality and if there were no errors in the measurement (represented by the observation operator $\boldsymbol{g}$). But in practice, all physical measurements are exposed to uncertainties. The simplest and most commonly used error model assumes a Gaussian distribution for these kinds of errors and mutual independence between error and measurement. This so-called **additive noise model** has the form

$$\boldsymbol{d}_{\text{obs}} = \boldsymbol{f}(\boldsymbol{y}) + \boldsymbol{e} \quad \text{with} \quad \boldsymbol{e} \sim \mathcal{N}_M(\boldsymbol{0}, \mathcal{C}_{dd}). \tag{3.20}$$

Two measurements taken at different measuring points are assumed to be independent of each other. Thus, the covariance matrix $\mathcal{C}_{dd}$ containing the relative and absolute errors of $\boldsymbol{d}_{\text{obs}}$ is diagonal with entries

$$\epsilon_k^2 = \left(\epsilon_\ell^{(\text{rel})} d_\ell^{(\text{obs})} + \epsilon_\ell^{(\text{abs})}\right)^2. \tag{3.21}$$

In the context of Bayesian inference, the vector of observations $\boldsymbol{d}_{\text{obs}}$ must be understood as a realization of a random vector conditional on the given realization of the parameter field $\boldsymbol{y}$ and given trend $\boldsymbol{\beta}$. Thus, equation (3.20) may be interpreted as

$$\boldsymbol{d}\big|\boldsymbol{p} = \boldsymbol{d}_{\text{obs}}, \tag{3.22}$$

or

$$\boldsymbol{d}\big|\boldsymbol{p} - \boldsymbol{f}(\boldsymbol{y}) \sim \mathcal{N}_M(\mathbf{0}, \mathcal{C}_{dd}), \tag{3.23}$$

or

$$\boldsymbol{d}\big|\boldsymbol{p} \sim \mathcal{N}_M(\boldsymbol{f}(\boldsymbol{y}), \mathcal{C}_{dd}), \tag{3.24}$$

or

$$\begin{aligned} \varrho\big(\boldsymbol{d}\big|\boldsymbol{p}\big) &= \varrho\big(\boldsymbol{d}\big|(\boldsymbol{y},\boldsymbol{\beta})\big) \\ &\propto \exp\left(-\frac{1}{2}\big(\boldsymbol{d}_{\text{obs}} - \boldsymbol{f}(\boldsymbol{y})\big)^T \cdot \mathcal{C}_{dd}^{-1} \cdot \big(\boldsymbol{d}_{\text{obs}} - \boldsymbol{f}(\boldsymbol{y})\big)\right). \end{aligned} \tag{3.25}$$

3.3.3 Posterior distribution

The marginal density $\varrho(\boldsymbol{d})$ in Bayes' theorem (3.11) is not depending on the parameter field $\boldsymbol{y}$. Therefore, it plays merely the role of a normalization factor. Bayes' theorem applied to our problem reduces to the relation

$$\begin{aligned} & \varrho\big(\boldsymbol{p}\big|\boldsymbol{d}\big) && \propto && \varrho\big(\boldsymbol{d}\big|\boldsymbol{p}\big) \cdot \varrho(\boldsymbol{p}) \\ \overset{(3.18),(3.25)}{\Longleftrightarrow} \quad & \varrho\big((\boldsymbol{y},\boldsymbol{\beta})\big|\boldsymbol{d}\big) && \propto && \varrho\big(\boldsymbol{d}\big|(\boldsymbol{y},\boldsymbol{\beta})\big) \cdot \varrho(\boldsymbol{y}|\boldsymbol{\beta}) \cdot \varrho(\boldsymbol{\beta}) \,. \end{aligned} \tag{3.26}$$

The MAP estimate (3.12) can be derived through maximizing $\varrho\big((\boldsymbol{y},\boldsymbol{\beta})\big|\boldsymbol{d}\big)$ or minimizing its negative natural logarithm

$$-\ln \varrho\big((\boldsymbol{y},\boldsymbol{\beta})\big|\boldsymbol{d}\big) = -\ln \varrho\big(\boldsymbol{d}\big|(\boldsymbol{y},\boldsymbol{\beta})\big) - \ln(\boldsymbol{y}|\boldsymbol{\beta}) - \ln(\boldsymbol{\beta}) + const. \tag{3.27}$$

Inserting (3.25), (3.16) and (3.17) into (3.27) and neglecting the additive constant, we obtain the **objective function**[1]

$$\boxed{\begin{aligned} \mathscr{J}(\boldsymbol{p}) := & \underbrace{\frac{1}{2}\big(\boldsymbol{d}_{\text{obs}} - \boldsymbol{f}(\boldsymbol{y})\big)^T \cdot \mathcal{C}_{dd}^{-1} \cdot \big(\boldsymbol{d}_{\text{obs}} - \boldsymbol{f}(\boldsymbol{y})\big)}_{=:\ \mathscr{J}^{d}(\boldsymbol{p})} \\ & + \underbrace{\frac{1}{2}(\boldsymbol{y} - \mathcal{X}\boldsymbol{\beta})^T \cdot \mathcal{R}_{yy}^{-1} \cdot (\boldsymbol{y} - \mathcal{X}\boldsymbol{\beta})}_{=:\ \mathscr{J}^{y}(\boldsymbol{p})} + \underbrace{\frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta}^\star)^T \cdot \mathcal{C}_{\beta\beta}^{-1} \cdot (\boldsymbol{\beta} - \boldsymbol{\beta}^\star)}_{=:\ \mathscr{J}^{\beta}(\boldsymbol{p})} . \end{aligned} \quad (3.28)}$$

[1] cost function, performance index, performance measure

for our minimization problem. The likelihood term $\mathscr{J}^d$ accounts for the discrepancy between simulated and real measurements whereas the prior terms $\mathscr{J}^\beta$ and $\mathscr{J}^y$ penalize variations from the structural parameters $\boldsymbol{\beta}^\star$ and $\boldsymbol{\theta}$. For a given estimate $\widehat{\boldsymbol{p}} = (\widehat{\boldsymbol{y}}, \widehat{\boldsymbol{\beta}})$, the terms $\mathscr{J}^\beta(\widehat{\boldsymbol{p}})$ and $\mathscr{J}^d(\widehat{\boldsymbol{p}})$ can be evaluated directly after solving the differential equations required to take the simulated measurements. The computation of $\mathscr{J}^y(\widehat{\boldsymbol{p}})$, however, requires some preparations as we will see in §3.4.1.

3.3.4 Link with the deterministic approach

Looking back, we see that the objective function (3.28) assumes exactly the form of a Tikhonov functional (3.9)

$$\mathscr{J}_\alpha(\boldsymbol{p}) = \frac{1}{2}\|\boldsymbol{d}_{\text{obs}} - \boldsymbol{F}(\boldsymbol{p})\|^2_{\mathcal{H}_\text{D}} + \frac{\alpha}{2}\|\boldsymbol{p} - \boldsymbol{p}^\star\|^2_{\mathcal{H}_\text{P}} \tag{3.29}$$

for the parameter $\boldsymbol{p} := (\boldsymbol{y}, \boldsymbol{\beta})^T$ with $\alpha = 1$, $\boldsymbol{p}^\star = (\mathcal{X}\boldsymbol{\beta}, \boldsymbol{\beta}^\star)^T$ and $\boldsymbol{F}(\boldsymbol{p}) := \boldsymbol{f}(\boldsymbol{y})$, if we apply the square root factorizations

$$\begin{array}{rcll} \mathcal{C}_{dd}^{-1} & = & \mathcal{D}^T\mathcal{D}, & \mathcal{D} \in \mathbb{R}^{M\times M}, \\ \mathcal{R}_{yy}^{-1} & = & \mathcal{Q}^T\mathcal{Q}, & \mathcal{Q} \in \mathbb{R}^{N\times N}, \\ \mathcal{C}_{\beta\beta}^{-1} & = & \mathcal{B}^T\mathcal{B}, & \mathcal{B} \in \mathbb{R}^{N_\beta\times N_\beta} \end{array} \tag{3.30}$$

and define the Hilbert space norms $\|.\|_{\mathcal{H}_\text{D}} := \|\mathcal{D}(.)\|_2$ and $\|.\|_{\mathcal{H}_\text{P}} := \|\mathcal{A}(.)\|_2$ where

$$\mathcal{A} = \begin{bmatrix} \mathcal{Q} & \mathbf{0} \\ \mathbf{0} & \mathcal{B} \end{bmatrix}. \tag{3.31}$$

This does not mean that the Bayesian approach and the Tikhonov regularization are equivalent. Whereas the Bayesian solution is an entire probability distribution, the Tikhonov solution can be regarded as a single sample from that distribution.

3.4 The quasi-linear geostatistical approach (QLGA)

3.4.1 Gauss-Newton scheme for the MAP estimate

Applying the square-root factorizations from (3.30) and defining the residuals

$$\boldsymbol{r}_1(\boldsymbol{y}, \boldsymbol{\beta}) := \mathcal{D}(\boldsymbol{d}_{\text{obs}} - \boldsymbol{f}(\boldsymbol{y})) \tag{3.32}$$

$$\boldsymbol{r}_2(\boldsymbol{y}, \boldsymbol{\beta}) := \mathcal{Q}(\boldsymbol{y} - \mathcal{X}\boldsymbol{\beta}) \tag{3.33}$$

$$\boldsymbol{r}_3(\boldsymbol{y}, \boldsymbol{\beta}) := \mathcal{B}(\boldsymbol{\beta} - \boldsymbol{\beta}^\star) \tag{3.34}$$

our **non-linear least-squares problem** reads:

$$\min_{(\boldsymbol{y},\boldsymbol{\beta})} \mathscr{J}(\boldsymbol{y},\boldsymbol{\beta}) := \frac{1}{2}\sum_i \left\|\boldsymbol{r}_i(\boldsymbol{y},\boldsymbol{\beta})\right\|_2^2 \tag{3.35}$$

subject to a set of ν **PDE constraints** (ν being the number of measurement types) of the form

$$C(\,Y,\ u(Y)\,) = 0 \quad \text{in } \Omega \cup \partial\Omega \tag{3.36}$$

specified by related forward problems which we will describe in detail in the next chapter (see §4.1.3).

The Jacobians of the residuals $\boldsymbol{r}_i$ are given by:

$$\begin{array}{llll} \mathfrak{J}_1(\boldsymbol{y},\boldsymbol{\beta}) = [& -\mathcal{D}\mathcal{H} & \mathbf{0} &] \in \mathbb{R}^{M\times(N+N_\beta)} \\ \mathfrak{J}_2(\boldsymbol{y},\boldsymbol{\beta}) = [& \mathcal{Q} & -\mathcal{Q}\mathcal{X} &] \in \mathbb{R}^{N\times(N+N_\beta)} \\ \mathfrak{J}_3(\boldsymbol{y},\boldsymbol{\beta}) = [& \mathbf{0} & \mathcal{B} &] \in \mathbb{R}^{N_\beta\times(N+N_\beta)} \end{array} \tag{3.37}$$

where

$$\mathcal{H} := \frac{\partial \boldsymbol{f}(\boldsymbol{y})}{\partial \boldsymbol{y}^T} = \left[\left.\frac{\partial u(\boldsymbol{y})}{\partial y_j}\right|_{\vec{x}=\vec{x}_\ell} \right]_{\ell,j} \quad \in \quad \mathbb{R}^{M\times N} \tag{3.38}$$

is the Jacobian of the model function. Since it is a measure for how sensitively the model outcome reacts to changes in the parameter field, it is called the **sensitivity matrix**. Setting up this matrix is the computationally most demanding part of the inversion scheme. We will see in section §4.2.3 how the entries of $\mathcal{H}$ can be computed efficiently.

The gradient of the objective functional is

$$\nabla\mathscr{J}(\boldsymbol{y},\boldsymbol{\beta}) = \sum_i \mathfrak{J}_i^T \boldsymbol{r}_i \overset{(3.30)}{=} \begin{bmatrix} -\mathcal{H}^T \mathcal{C}_{dd}^{-1}(\boldsymbol{d}_{\text{obs}} - \boldsymbol{f}(\boldsymbol{y})) + \mathcal{R}_{yy}^{-1}(\boldsymbol{y} - \mathcal{X}\boldsymbol{\beta}) \\ \\ -\mathcal{X}^T\mathcal{R}_{yy}^{-1}(\boldsymbol{y} - \mathcal{X}\boldsymbol{\beta}) + \mathcal{C}_{\beta\beta}^{-1}(\boldsymbol{\beta} - \boldsymbol{\beta}^\star) \end{bmatrix} \tag{3.39}$$

The Gauss-Newton Hessian of $\mathscr{J}(\boldsymbol{y},\boldsymbol{\beta})$ is:

$$\nabla^2\mathscr{J}(\boldsymbol{y},\boldsymbol{\beta}) \doteq \sum_i \mathfrak{J}_i^T \mathfrak{J}_i^T\,. \tag{3.40}$$

Of course, this is not the full Hessian, because second order terms are skipped due to the linearization of the model function $\boldsymbol{f}(\boldsymbol{y})$.

The **Gauss-Newton scheme** for (3.35) with line search reads

$$\underbrace{\begin{bmatrix} \boldsymbol{y}_{k+1} \\ \\ \boldsymbol{\beta}_{k+1} \end{bmatrix}}_{=\boldsymbol{p}_{k+1}} = \underbrace{\begin{bmatrix} \boldsymbol{y}_k \\ \\ \boldsymbol{\beta}_k \end{bmatrix}}_{=\boldsymbol{p}_k} + \alpha \underbrace{\begin{bmatrix} \widehat{\boldsymbol{y}} - \boldsymbol{y}_k \\ \\ \widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_k \end{bmatrix}}_{=:\Delta\widehat{\boldsymbol{p}}_k} \qquad \alpha \in (0,1] \tag{3.41}$$

where the Gauss-Newton direction in iteration k is computed by solving the linear system

$$\left(\sum_i \mathcal{J}_i^T(\boldsymbol{p}_k)\mathcal{J}_i^T(\boldsymbol{p}_k)\right) \cdot \Delta\widehat{\boldsymbol{p}}_k = -\nabla\mathscr{J}(\boldsymbol{p}_k)\,. \tag{3.42}$$

This is equivalent to solving

$$\begin{bmatrix} -\mathcal{H}_k^T\mathcal{C}_{dd}^{-1}\mathcal{H}_k + \mathcal{R}_{yy}^{-1} & -\mathcal{R}_{yy}^{-1}\mathcal{X} \\ -\mathcal{X}^T\mathcal{R}_{yy}^{-1} & \mathcal{X}^T\mathcal{R}_{yy}^{-1}\mathcal{X} + \mathcal{C}_{\beta\beta}^{-1} \end{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{y}} \\ \widehat{\boldsymbol{\beta}} \end{bmatrix} = \begin{bmatrix} \mathcal{H}_k^T\mathcal{C}_{dd}^{-1}\boldsymbol{d}_k \\ \mathcal{C}_{\beta\beta}^{-1}\boldsymbol{\beta}^\star \end{bmatrix} \tag{3.43}$$

where

$$\mathcal{H}_k := \frac{\partial \boldsymbol{f}(\boldsymbol{y}_k)}{\partial \boldsymbol{y}^T} \tag{3.44}$$

and

$$\boldsymbol{d}_k := \boldsymbol{d}_{\text{obs}} - \Big(\boldsymbol{f}(\boldsymbol{y}_k) - \mathcal{H}_k\boldsymbol{y}_k\Big) \tag{3.45}$$

is the vector of measurements, adjusted for the linearization about the last iterate $\boldsymbol{y}_k$.

In our application, $\mathscr{O}(N) \geq 10^6$ and $\mathscr{O}(M) \leq 10^3$. A direct computation of the inverse $\mathcal{R}_{yy}^{-1}$ is inconvenient. Instead of solving the large linear system (3.43) for $\widehat{\boldsymbol{y}} \in \mathbb{R}^N$, [Kitanidis, 1995] has reformulated this problem as a small linear system for the computation of a coefficient vector $\widehat{\boldsymbol{\xi}} \in \mathbb{R}^M$. The estimated log conductivity field $\widehat{\boldsymbol{y}}$ is then represented as a linear combination of M cross-covariance fields which is outlined in the following: Subtracting $\mathcal{H}_k^T\mathcal{C}_{dd}^{-1}\mathcal{H}_k\mathcal{X}\widehat{\boldsymbol{\beta}}$ on both sides of the first line of (3.43) leads to the equation

$$\begin{aligned} &(\mathcal{R}_{yy}^{-1} + \mathcal{H}_k^T\mathcal{C}_{dd}^{-1}\mathcal{H}_k)\cdot(\widehat{\boldsymbol{y}} - \mathcal{X}\widehat{\boldsymbol{\beta}}) = \mathcal{H}_k^T\mathcal{C}_{dd}^{-1}\cdot(\boldsymbol{d}_k - \mathcal{H}_k\mathcal{X}\widehat{\boldsymbol{\beta}}) \\ \Leftrightarrow\quad &\widehat{\boldsymbol{y}} - \mathcal{X}\widehat{\boldsymbol{\beta}} = (\mathcal{R}_{yy}^{-1} + \mathcal{H}_k^T\mathcal{C}_{dd}^{-1}\mathcal{H}_k)^{-1}\cdot\mathcal{H}_k^T\mathcal{C}_{dd}^{-1}\cdot(\boldsymbol{d}_k - \mathcal{H}_k\mathcal{X}\widehat{\boldsymbol{\beta}})\,. \end{aligned} \tag{3.46}$$

Applying the matrix identity (A.15) on the block matrix

$$\begin{bmatrix} \mathcal{R}_{yy}^{-1} & \mathcal{H}_k^T \\ \mathcal{H}_k & -\mathcal{C}_{dd} \end{bmatrix},$$

the term

$$(\mathcal{R}_{yy}^{-1} + \mathcal{H}_k^T\mathcal{C}_{dd}^{-1}\mathcal{H}_k)^{-1}\mathcal{H}_k^T\mathcal{C}_{dd}^{-1}$$

containing the inverse of an $N \times N$-matrix can be simplified to

$$\mathcal{R}_{yy}\mathcal{H}_k^T(\mathcal{C}_{dd} + \mathcal{H}_k\mathcal{R}_{yy}\mathcal{H}_k^T)^{-1}$$

which only requires the inverse of an $M \times M$-matrix. This conversion is the first big contribution to make this scheme applicable to real-world problems. Hence, equation (3.46) becomes

$$\widehat{\boldsymbol{y}} = \mathcal{X}\widehat{\boldsymbol{\beta}} + \mathcal{R}_{yy}\mathcal{H}_k^T \underbrace{(\mathcal{C}_{dd} + \mathcal{H}_k\mathcal{R}_{yy}\mathcal{H}_k^T)^{-1}(\boldsymbol{d}_k - \mathcal{H}_k\mathcal{X}\widehat{\boldsymbol{\beta}})}_{=:\widehat{\boldsymbol{\xi}}} . \tag{3.47}$$

Inserting this into the second line of (3.43), we get:

$$\mathcal{C}_{\beta\beta}^{-1}\widehat{\boldsymbol{\beta}} - \mathcal{X}^T\mathcal{H}_k^T\widehat{\boldsymbol{\xi}} = \mathcal{C}_{\beta\beta}^{-1}\boldsymbol{\beta}^\star . \tag{3.48}$$

Combining (3.47) with (3.48) yields the linear system of cokriging equations

$$\underbrace{\begin{bmatrix} \mathcal{H}_k\mathcal{R}_{yy}\mathcal{H}_k^T + \mathcal{C}_{dd} & \mathcal{H}_k\mathcal{X} \\ (\mathcal{H}_k\mathcal{X})^T & -\mathcal{C}_{\beta\beta}^{-1} \end{bmatrix}}_{=:\,\mathcal{M}_k^{\text{cokriging}}\ =:\,\mathcal{M}} \begin{bmatrix} \widehat{\boldsymbol{\xi}} \\ \widehat{\boldsymbol{\beta}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{d}_k \\ -\mathcal{C}_{\beta\beta}^{-1}\boldsymbol{\beta}^\star \end{bmatrix} \tag{3.49}$$

and an estimate for the next iteration is given by (3.47):

$$\widehat{\boldsymbol{y}} = \mathcal{X}\widehat{\boldsymbol{\beta}} + \mathcal{R}_{yy}\mathcal{H}_k^T\widehat{\boldsymbol{\xi}} . \tag{3.50}$$

The columns of $\mathcal{H}_k^T$ are the lines of $\mathcal{H}_k$ each of which represents the **sensitivity field** for one measuring point. We call the columns $\boldsymbol{z}_i \in \mathbb{R}^N$ of the cross-covariance matrix $\mathcal{R}_{yy}\mathcal{H}_k^T$ the corresponding **cross-covariance fields**. Therefore, an alternative representation of (3.50) is

$$\widehat{\boldsymbol{y}} = \mathcal{X}\widehat{\boldsymbol{\beta}} + \sum_{i=1}^{M} \widehat{\xi}_i \cdot \boldsymbol{z}_i . \tag{3.51}$$

Remark 3.1. *Due to its small size, the linear system (3.49) can be solved directly using LU decomposition. Different types of measurements (hydraulic head, tracer concentration and arrival times) result in large differences ranging in several orders of magnitude between the sensitivity fields, which are exactly the rows of $\mathcal{H}_k$. This produces an ill-conditioned cokriging matrix $\mathcal{M}$. Row equilibration is a natural strategy to balance out these differences. We solve the equivalent system*

$$\mathcal{D}\mathcal{M}\boldsymbol{\nu} = \mathcal{D}\boldsymbol{b} \tag{3.52}$$

where $\mathcal{D}$ is a diagonal matrix with entries

$$\mathcal{D}_{\ell,\ell} := \left(\sum_{m=1}^{M+N_\beta} |\mathcal{M}_{\ell,m}| \right)^{-1} , \qquad \ell = 1, ..., M + N_\beta . \tag{3.53}$$

Applying a **line search** with scaling factor $\alpha = \alpha_j = 2^{-j} (j = 0, 1, 2, ...)$

the next iterate (3.41)

$$\begin{bmatrix} \boldsymbol{y}_{k+1} \\ \\ \boldsymbol{\beta}_{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{y}_{k} \\ \\ \boldsymbol{\beta}_{k} \end{bmatrix} + \alpha_j \begin{bmatrix} \widehat{\boldsymbol{y}} - \boldsymbol{y}_{k} \\ \\ \widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_{k} \end{bmatrix} \qquad \alpha \in (0, 1] \tag{3.54}$$

is finally accepted if the cost function decreases:

$$\mathscr{J}(\boldsymbol{y}_{k+1}, \boldsymbol{\beta}_{k+1}) < \mathscr{J}(\boldsymbol{y}_{k}, \boldsymbol{\beta}_{k}) \, . \tag{3.55}$$

Whereas the first and last terms in (3.28) can be evaluated directly by inserting $\boldsymbol{p}_k = (\boldsymbol{y}_k, \boldsymbol{\beta}_k)^T$ into

$$\mathscr{J}^d(\boldsymbol{p}) = \frac{1}{2}\big(\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{d}_{\text{obs}}\big)^T \cdot \mathcal{C}_{dd}^{-1} \cdot \big(\boldsymbol{f}(\boldsymbol{y}) - \boldsymbol{d}_{\text{obs}}\big) \tag{3.56}$$

and

$$\mathscr{J}^\beta(\boldsymbol{p}) = \frac{1}{2}\big(\boldsymbol{\beta} - \boldsymbol{\beta}^\star\big)^T \cdot \mathcal{C}_{\beta\beta}^{-1} \cdot \big(\boldsymbol{\beta} - \boldsymbol{\beta}^\star\big), \tag{3.57}$$

a naïve evaluation of the prior term $\mathscr{J}^y$ would require the inverse of $\mathcal{R}_{yy}$. Following the derivations in [Nowak, 2005; Nowak and Cirpka, 2004], the prior term resulting from successive linearization can be evaluated efficiently by reusing the blocks computed for the setup of the cokriging matrix (3.49):

$$\begin{aligned} \mathscr{J}^y(\widehat{\boldsymbol{p}}) &= {}^1\!/\!_2 \cdot \big(\widehat{\boldsymbol{y}} - \mathcal{X}\widehat{\boldsymbol{\beta}}\big)^T \cdot \mathcal{R}_{yy}^{-1} \cdot \big(\widehat{\boldsymbol{y}} - \mathcal{X}\widehat{\boldsymbol{\beta}}\big) \\ &\overset{(3.50)}{=} {}^1\!/\!_2 \cdot \big(\mathcal{R}_{yy}\mathcal{H}_k^T\widehat{\boldsymbol{\xi}}\,\big)^T \cdot \mathcal{R}_{yy}^{-1}\mathcal{R}_{yy}\mathcal{R}_{yy}^{-1} \cdot \big(\mathcal{R}_{yy}\mathcal{H}_k^T\widehat{\boldsymbol{\xi}}\,\big) \\ &= {}^1\!/\!_2 \cdot \widehat{\boldsymbol{\xi}}^T \underbrace{\mathcal{H}_k \cdot \mathcal{R}_{yy} \cdot \mathcal{H}_k^T}_{=:\, \widehat{\mathcal{Q}}_{dd} \,\in\, \mathbb{R}^{M\times M}} \widehat{\boldsymbol{\xi}} \\ &= {}^1\!/\!_2 \cdot \widehat{\boldsymbol{\xi}}^T \cdot \widehat{\mathcal{Q}}_{dd} \cdot \widehat{\boldsymbol{\xi}} \end{aligned} \quad . \tag{3.58}$$

$\mathscr{J}^y(\widehat{\boldsymbol{p}})$ is understood to be evaluated at $\alpha_0 = 1$. For $\alpha_j < 1$, the linear line search (3.54) may be extended to

$$\widehat{\boldsymbol{\xi}}_{k+1} = \widehat{\boldsymbol{\xi}}_k + \alpha_j \cdot \big(\widehat{\boldsymbol{\xi}} - \widehat{\boldsymbol{\xi}}_k\big) \tag{3.59}$$

$$\mathcal{Q}_{dd,k+1} = \mathcal{Q}_{dd,k} + \alpha_j \cdot \big(\widehat{\mathcal{Q}}_{dd} - \mathcal{Q}_{dd,k}\big) \tag{3.60}$$

where $\mathcal{Q}_{dd,k}$ and $\widehat{\boldsymbol{\xi}}_k$ are retrieved from the storage of the last iteration. Hence,

$$\mathscr{J}^{y}(\boldsymbol{p}_{k+1}) \quad = \quad {}^1\!/\!_2 \cdot \widehat{\boldsymbol{\xi}}^T_{k+1} \cdot \mathcal{Q}_{dd,k+1} \cdot \widehat{\boldsymbol{\xi}}_{k+1} \tag{3.61}$$

Starting with the **initial guess** (a homogeneous field)

$$\begin{bmatrix} \boldsymbol{y}_0 \\ \\ \boldsymbol{\beta}_0 \end{bmatrix} := \begin{bmatrix} \mathcal{X}\boldsymbol{\beta}^\star \\ \\ \boldsymbol{\beta}^\star \end{bmatrix} \tag{3.62}$$

it is clear that $\mathscr{J}^{\beta}(\boldsymbol{p}_0) = \mathscr{J}^{y}(\boldsymbol{p}_0) = 0$. We set $\mathcal{Q}_{dd,0} := \mathbf{0}$ and $\widehat{\boldsymbol{\xi}}_0 = \mathbf{0}$. The dominating term that needs to be reduced is the discrepancy between simulated and real measurements $\mathscr{J}^{d}(\boldsymbol{p}_0)$. For its calculation, the model function needs to be evaluated. For the initial guess $\boldsymbol{p}_0$, this calculation can be skipped by setting $\mathscr{J}(\boldsymbol{p}_0)$ to a very large value. The **final stopping criterion** is reached when the new estimate remains almost unchanged or when the cost function cannot be reduced any further. We denote the estimated solution to which the iterative scheme converges by $\boldsymbol{y}_{\text{est}}$ and $\boldsymbol{\beta}_{\text{est}}$.

3.4.2 Uncertainty quantification

Note that the expression (3.28) is not quadratic with respect to $\boldsymbol{y}$ since the model function $\boldsymbol{f}(\boldsymbol{y})$ is generally non-linear. Therefore, $\boldsymbol{y}$ does not obey a Gaussian distribution. However, in each iteration of the quasi-linear approach, our MAP estimator $\boldsymbol{y}_{\text{est}}$ for the log conductivity $\boldsymbol{y}|\boldsymbol{d}$ is 'made' normally distributed by linearization of $\boldsymbol{f}(\boldsymbol{y})$. Thus, it can be considered as equivalent to a conditional mean estimator (C.14) since all random quantities involved are normally distributed. Therefore, if the estimator could be computed perfectly, the solution $\boldsymbol{y}|\boldsymbol{d}$ would follow a distribution which is defined by the conditional mean estimator $\boldsymbol{y}_{\text{est}}$ and the conditional covariance $\mathcal{R}_{yy|d}$:

$$\boldsymbol{y}|\boldsymbol{d} \sim \mathcal{N}(\boldsymbol{y}_{\text{est}}, \mathcal{R}_{yy|d}) \,. \tag{3.63}$$

Successive linearization about the last estimate $\boldsymbol{y}_k$ yields the approximation[1]

$$\mathcal{R}_{yy|d_k} = \mathcal{R}_{yy} - \begin{bmatrix} \mathcal{H}_k\mathcal{R}_{yy} \\ \\ \mathcal{X}^T \end{bmatrix}^T \cdot [\mathcal{M}_k^{\text{cokriging}}]^{-1} \cdot \begin{bmatrix} \mathcal{H}_k\mathcal{R}_{yy} \\ \\ \mathcal{X}^T \end{bmatrix} \tag{3.64}$$

which can be found in [Li et al., 2005]. This expression can be used as a lower bound for the conditional covariance $\mathcal{R}_{yy|d}$ (see also [Nowak, 2005], §6.2.2 and the references cited therein) which expresses the uncertainty of our estimator for $\boldsymbol{y}|\boldsymbol{d}$. In most other applications, the uncertainty of a statistical estimator has to be determined through a costly generation of conditional realizations (see e.g. [Tan et al., 2012]).

[1] We skip its derivation here, for which the marginal distribution of $\boldsymbol{y}$ with the generalized covariance matrix ($\mathcal{G}_{yy} = \mathcal{R}_{yy} + \mathcal{X}\mathcal{C}_{\beta\beta}\mathcal{X}^T$) must be taken into account.

3.4.3 Test of unbiasedness

Equation (3.63) is another way of saying that the difference between true and estimated parameters have zero mean and covariance $\mathcal{R}_{yy|d}$. In this case, $\boldsymbol{y}_{\text{est}}$ is called an **unbiased estimator** for $\boldsymbol{y}|\boldsymbol{d}$ [Li et al., 2005]. Since $\boldsymbol{y}_{\text{true}}$ can be interpreted as a sample drawn from the distribution (3.63), this is equivalent to

$$\boldsymbol{\varepsilon} := \mathcal{Q}_{yy|d}^{-1} \cdot (\boldsymbol{y}_{\text{true}} - \boldsymbol{y}_{\text{est}}) \sim \mathcal{N}(\mathbf{0}, \mathbf{Id}) \tag{3.65}$$

according to Theorem C.1(b), provided that $\mathcal{R}_{yy|d}$ is positive definite with square root $\mathcal{Q}_{yy|d} \in \mathbb{R}^{N\times N}$. The computation of the weighted error ε requires a Cholesky decomposition of $\mathcal{R}_{yy|d}$ which is costly if N is very large. Let $\sigma_1^2, ..., \sigma_N^2$ be the diagonal entries of $\mathcal{R}_{yy|d}$. We denote by

$$\mathbf{v}_{\text{est}} = (\sigma_1^2, ..., \sigma_N^2)^T \tag{3.66}$$

the field of estimated variances. If $\mathcal{R}_{yy|d}$ is diagonal dominant,

$$\widehat{\boldsymbol{\varepsilon}} := \text{diag}(\sigma_1^{-1}, ..., \sigma_N^{-1}) \cdot (\boldsymbol{y}_{\text{true}} - \boldsymbol{y}_{\text{est}}). \tag{3.67}$$

is a good approximation for the weighted error.

Remark 3.2. *Note that the diagonal entries of $\mathcal{R}_{yy}$ are simply the field variance σ_Y^2. This simplifies the computation of the diagonal entries of $\mathcal{R}_{yy|d_k}$ in (3.64).*

3.4.4 An estimator for the optimal value of the objective function

If we assume that the distributions (3.20) and (3.17) hold for the estimated solution $\boldsymbol{p}_{k+1} = (\boldsymbol{y}_{k+1}, \boldsymbol{\beta}_{k+1})^T$ and additionally

$$\widehat{\boldsymbol{\xi}}_{k+1} \sim \mathcal{N}_M(\mathbf{0}, \mathcal{Q}_{dd,k+1}) \tag{3.68}$$

holds for the coefficient vector $\widehat{\boldsymbol{\xi}}_{k+1}$, then it is trivial to show that

$$\begin{aligned} 2 \times \mathscr{J}^d(\boldsymbol{p}_{k+1}) &\sim \chi_M^2 & (3.69)\\ 2 \times \mathscr{J}^y(\boldsymbol{p}_{k+1}) &\sim \chi_M^2 & (3.70)\\ 2 \times \mathscr{J}^\beta(\boldsymbol{p}_{k+1}) &\sim \chi_{N_\beta}^2 & (3.71)\end{aligned}$$

using Theorem C.1(b). Additivity of the χ^2-distribution leads to

$$2 \times \mathscr{J}(\boldsymbol{p}_{k+1}) \sim \chi_{2M+N_\beta}^2 \, . \tag{3.72}$$

Due to Remark C.4,

$$\begin{aligned} \mu := E[\mathscr{J}(\boldsymbol{p}_{k+1})] &\approx M + N_\beta/2 & (3.73)\\ \sigma^2 := Var[\mathscr{J}(\boldsymbol{p}_{k+1})] &\approx 2M + N_\beta & (3.74)\end{aligned}$$

This may be used to construct a confidence interval estimator of the form

$$P(\mu - \sigma \leq \mathscr{J} \leq \mu + \sigma) \approx 68.27\%, \tag{3.75}$$
$$P(\mu - 2\sigma \leq \mathscr{J} \leq \mu + 2\sigma) \approx 95.45\%, \tag{3.76}$$
$$P(\mu - 3\sigma \leq \mathscr{J} \leq \mu + 3\sigma) \approx 99.73\%, \tag{3.77}$$

known as the so-called **three-sigma rule** or **68–95–99.7 rule** for normal distributions. However, we will see in the 3-D inversion example presented in Chapter 7, that the range of the optimal value of $\mathscr{J}$ it is not easily matched, but depends strongly on the magnitude of the measurement errors.

Targeting for the objective function of a minimization problem to fall below a certain threshold may be a more convenient choice. In the χ^2 distribution tables, the 100α-th **percentile** is often denoted by $\chi^2_{1-\alpha}(r)$ where r is the degree of freedom and 100α is the percentage of all outcoming values that are smaller than the threshold. To be more precise, we choose $\alpha = 0.95$ and look up the value

$$\mathscr{J}^* := \frac{1}{2} \cdot \chi^2_{0.05}(2M + N_\beta) \ . \tag{3.78}$$

Since

$$P(\mathscr{J} \leq \mathscr{J}^*) = 95\% \ , \tag{3.79}$$

our goal is reached if $\mathscr{J}(\boldsymbol{p}_{k+1}) \leq \mathscr{J}^*$.

3.5 Inversion algorithm

The most important steps of the Gauss-Newton based QLGA-method are outlined in the following algorithm. The details for the computations of the forward and adjoint problems in steps (7) and (12.3) are clarified in Chapters 4 and 5. If the original parameter field $\boldsymbol{y}_{\text{true}}$ is available, at the end, a histogram for the weighted error (3.67) can be plotted and compared against the standard normal distribution to check for the unbiasedness of the estimate $\boldsymbol{y}_{\text{est}}$.

Algorithm 3.1 QLGA (by Kitanidis, Cirpka and Nowak)

```
 1: Input:
 2: - all input data required for Algorithm 2.2;
 3: - vector of observations $\boldsymbol{d}_{\text{obs}} := (u_1, ..., u_M)^T$ from measurements of a quantity
    $u$ at discrete points $\vec{x}_1, ..., \vec{x}_M$ with measurement errors given by $\mathcal{C}_{dd}$ (3.21). For
    different indices $\ell \neq j$, $u_\ell$ and $u_j$ may be of different types ($\phi$, $m_0^c$ or $m_1^c$);
 4: - uncertainties of the trend coefficients $\boldsymbol{\beta}^\star$ given by $\mathcal{C}_{\beta\beta}$ (3.17);

 5: Initialization steps:
 6: (1) – (3) Same as in Algorithm 2.2.

 7: Iterative scheme starts here:
 8: (4) $k = 0$;
 9: if solution $(\boldsymbol{y}_{\text{est}}, \boldsymbol{\beta}_{\text{est}})^T$ from a previous inversion exists then
10:     (4.1) Initialize $\boldsymbol{y}_0 = \boldsymbol{y}_{\text{est}}$; $\boldsymbol{\beta}_0 = \boldsymbol{\beta}_{\text{est}}$;
11: else
12:     (4.2) Initialize $\boldsymbol{y}_0 = \mathcal{X}\boldsymbol{\beta}^\star$; $\boldsymbol{\beta}_0 = \boldsymbol{\beta}^\star$;            ▷ // Homogeneous guess
13: end if
14: (5) Initialize $\mathscr{J}(\boldsymbol{p}_0) = 10^{12}$;            ▷ // Any large value fulfills the purpose.
15:
16: while $k < k_{\max}$ do            ▷ // Outer iteration loop
17:     (6) Identify $\boldsymbol{p}_k = (\boldsymbol{y}_k, \boldsymbol{\beta}_k)^T$;
18:
19:     (7) Call Algorithm 4.2 with input $\boldsymbol{y}_k$;            ▷ // Get $\boldsymbol{f}(\boldsymbol{y}_k)$, $\mathcal{H}_k^T$ and $\mathcal{R}_{yy}\mathcal{H}_k^T$ .
20:
21:     (8) Evaluate adjusted vector of measurements as in (3.45);
            $$\boldsymbol{d}_k = \boldsymbol{d}_{\text{obs}} - \Big(\boldsymbol{f}(\boldsymbol{y}_k) - \mathcal{H}_k \boldsymbol{y}_k\Big)$$
22:     (9) Setup and store the matrix $\mathcal{M}_k^{\text{cokriging}}$ from (3.49);
23:     (10) Solve the cokriging system (3.49) with row equilibration (3.53);
            $$\mathcal{D}\mathcal{M}_k^{\text{cokriging}} \cdot \begin{bmatrix} \widehat{\boldsymbol{\xi}} \\ \\ \widehat{\boldsymbol{\beta}} \end{bmatrix} = \mathcal{D} \begin{bmatrix} \boldsymbol{d}_k \\ \\ -\mathcal{C}_{\beta\beta}^{-1}\boldsymbol{\beta}^\star \end{bmatrix}$$
```

Algorithm 3.1 (continued): QLGA (by Kitanidis, Cirpka and Nowak)

```
24:     (11) Evaluate $\widehat{\boldsymbol{y}}$ as in (3.50) and compute the Gauss-Newton direction:
            $$\widehat{\boldsymbol{y}} = \mathcal{X}\widehat{\boldsymbol{\beta}} + \mathcal{R}_{yy}\mathcal{H}_k^T\widehat{\boldsymbol{\xi}}\,, \qquad \Delta\widehat{\boldsymbol{p}}_k = \begin{bmatrix} \widehat{\boldsymbol{y}} - \boldsymbol{y}_k \\ \\ \widehat{\boldsymbol{\beta}} - \boldsymbol{\beta}_k \end{bmatrix}$$
25:     (12) Set $\alpha = 1.0$;
26:     while $\alpha > \alpha_{\text{min}}$ do                                   ▷ // Line search loop
27:         (12.1) $\boldsymbol{p}_{k+1} = \boldsymbol{p}_k + \alpha\Delta\widehat{\boldsymbol{p}}_k$;
28:         (12.2) Identify $\boldsymbol{p}_{k+1} = (\boldsymbol{y}_{k+1}, \boldsymbol{\beta}_{k+1})^T$;
29:
30:         (12.3) Call Algorithm 4.1 with input $\boldsymbol{y}_{k+1}$;      ▷ // Get $\boldsymbol{f}(\boldsymbol{y}_{k+1})$ for (12.5).
31:
32:         (12.4) Compute $\widehat{\boldsymbol{\xi}}_{k+1}$ and $\mathcal{Q}_{dd,k+1}$ as in (3.59) and (3.60);
33:         (12.5) Compute the objective function as in (3.56), (3.57) and (3.61):
34:             $\mathscr{J}(\boldsymbol{p}_{k+1}) = \mathscr{J}^d(\boldsymbol{p}_{k+1}) + \mathscr{J}^\beta(\boldsymbol{p}_{k+1}) + \mathscr{J}^y(\boldsymbol{p}_{k+1})$;
35:
36:         if $\mathscr{J}(\boldsymbol{p}_{k+1}) > \mathscr{J}(\boldsymbol{p}_k)$ then
37:             (12.6) $\alpha = \alpha/2$;
38:             if $\alpha <= \alpha_{\text{min}}$ and $\mathscr{J}(\boldsymbol{p}_{k+1}) > \mathscr{J}(\boldsymbol{p}_k)$ then
39:                 (12.7) exit;                                          ▷ // The algorithm got stuck!
40:             end if
41:         else
42:             if $\mathscr{J}(\boldsymbol{p}_k) - \mathscr{J}(\boldsymbol{p}_{k+1}) > \Delta_J$ and $\|\boldsymbol{y}_{k+1} - \boldsymbol{y}_k\|_\infty > \Delta_{\boldsymbol{y}}$ then
43:                 (12.8) $\boldsymbol{p}_k = \boldsymbol{p}_{k+1}$;      ▷ // Accept new estimate as real improvement.
44:                 (12.9) $k = k + 1$;
45:             else
46:                 break;                                               ▷ // Convergence reached.
47:             end if
48:         end if
49:     end while                                                        ▷ // Line search loop
50: end while                                                            ▷ // Outer iteration loop
51:
52: (13) Store the estimated parameter field to $\boldsymbol{y}_{\text{est}} \in \mathbb{R}^N$;
53:
54: (14) Compute the diagonal entries of $\mathcal{R}_{yy|d_k}$ (3.64) and
55:      store the result to $\mathbf{v}_{\text{est}} = (\sigma_1^2, ..., \sigma_N^2)^T$;   ▷ // Simplified UQ (see Remark 3.2)
56:
```

Chapter 4

Model Equations and the Computation of Sensitivities

In this chapter, we state the forward model PDEs realizing the solution operator (3.3) and derive the adjoint PDEs for the computation of the gradient of the solution operator. In the next chapter, we will introduce efficient discretization schemes for the numerical solution of the boundary value problems (BVPs) presented here.

4.1 Model equations

Physical models describing flow and transport processes in a confined aquifer are well developed and can be found in the textbooks by Bear and Cheng [2010] or De Marsily [1986] or in the lecture notes by Roth [2012]. Since the computational domain $\Omega \subset \mathbb{R}^d$ is truncated out of the physical domain (§2.2), the boundary conditions are unknown. They need to be approximated by **artificial boundary conditions**. To minimize their influence on the flow in the region where pumping tests and tracer experiments are being conducted, Ω should be chosen large enough such that these artificial boundaries are sufficiently far away from the region of interest.

4.1.1 Groundwater flow

For simplicity, we assume that the computational domain Ω is a bounded rectangular cuboid in $\mathbb{R}^d$ $(d = 2, 3)$ in which the boundary is subdivided into a non-empty Dirichlet boundary (Γ_D) and a Neumann boundary (Γ_N) section. We consider the *steady-state groundwater flow equation*:

$$\text{div}\big(-K\nabla\phi\big) = \tilde{w}_{\text{inj}} - \tilde{w}_{\text{ext}} \quad \text{in} \quad \Omega \tag{4.1}$$

subject to the boundary conditions:

$$\phi = \hat{\phi}_D \quad \text{on } \Gamma_D \tag{4.2}$$

$$\vec{n}\cdot(-K\nabla\phi) = 0 \quad \text{on } \Gamma_N \tag{4.3}$$

in which $\vec{n}(\vec{x})$ is the unit outer normal vector, $K(\vec{x}) > 0$ is the spatially variable, but locally isotropic hydraulic conductivity $[m/s]$, $\phi(\vec{x})$ is the hydraulic head $[m]$ and the source terms on the right hand side prescribe the rates (volumetric flux per unit volume $[1/s]$) of injection and extraction wells $W_{\text{inj}}, W_{\text{ext}} \subsetneq \Omega$:

$$\tilde{w}_{\text{inj}}(\vec{x}) \begin{cases} > 0 & \vec{x} \in W_{\text{inj}} \\ = 0 & \vec{x} \in \Omega \backslash W_{\text{inj}} \end{cases} \qquad \tilde{w}_{\text{ext}}(\vec{x}) \begin{cases} > 0 & \vec{x} \in W_{\text{ext}} \\ = 0 & \vec{x} \in \Omega \backslash W_{\text{ext}} \end{cases} \tag{4.4}$$

This is an elliptic equation for which the coefficient K may be highly variable. The fluid motion is driven by an ambient flow prescribed by a difference in the hydraulic head between inflow and outflow boundaries (where $\hat{\phi}_D$ is defined) and by injection and extraction wells located within the domain. The flow field is induced by the head distribution (Darcy's law (2.2)):

$$\vec{q} = -K\nabla\phi \quad \text{in} \quad \Omega. \tag{4.5}$$

$\vec{q}(\vec{x})$ is sometimes called the **Darcy velocity** $[m/s]$. It is related to the pore water velocity (or seepage velocity) $\vec{v}(\vec{x})$ via

$$\vec{q} = \theta\vec{v} \tag{4.6}$$

where θ is the dimensionless porosity. Throughout this work, the porosity θ is supposed to be a known constant.

4.1.2 Subsurface solute transport

A conservative tracer used to track flow motion has no influence on the flow itself. Its concentration $c(t,\vec{x})$ $[kg/m^3]$ is described by the transient convection-diffusion-reaction equation [Bear and Cheng, 2010]

$$\frac{\partial(\theta c)}{\partial t} + \text{div}(-\mathfrak{D}\nabla c) + \vec{q}\,\nabla c = \tilde{w}_{\text{inj}}\,\tilde{c}_{\text{inj}} - \tilde{w}_{\text{ext}}\,c \quad \text{in } (0,T]\times\Omega \tag{4.7}$$

in which $\tilde{c}_{\text{inj}}(t,\vec{x})$ is the concentration at the injection well and

$$\mathcal{D} = \theta \mathcal{D}_S \tag{4.8}$$

is the dispersion tensor $[m^2/s]$ given by Scheidegger [1961]:

$$\mathcal{D}_S = \left(\alpha_\ell - \alpha_t\right) \frac{\vec{v} \cdot \vec{v}^T}{\| \vec{v} \|_2} + \left(\alpha_t \, \|\vec{v}\|_2 + D_m\right) \mathbf{Id}, \tag{4.9}$$

where α_ℓ and α_t are the longitudinal and transversal dispersivities $[m]$, D_m is the molecular diffusion coefficient $[m^2/s]$ and **Id** is the identity matrix.

Initial and boundary conditions

The initial distribution is supposed to be zero:

$$c(0, \vec{x}) = 0 \quad \text{in} \quad \Omega \tag{4.10}$$

For the transport equation, we distinguish three types of boundaries, inflow, outflow and characteristic boundary:

$$\begin{array}{rcl} \Gamma_- & = & \{\vec{x} \in \partial\Omega : \vec{q}(\vec{x}) \cdot \vec{n}(\vec{x}) < 0\} \\ \Gamma_+ & = & \{\vec{x} \in \partial\Omega : \vec{q}(\vec{x}) \cdot \vec{n}(\vec{x}) > 0\} \\ \Gamma_0 & = & \{\vec{x} \in \partial\Omega : \vec{q}(\vec{x}) \cdot \vec{n}(\vec{x}) = 0\} \end{array} \tag{4.11}$$

Tracer in a constant concentration may enter over a fixed time period $T_{\text{inj}} > 0$ through the injection well or somewhere on the inflow boundary:

$$c(t, \vec{x}) \;=\; \hat{c}_D(\vec{x}) \quad \text{on} \quad \Gamma_- \tag{4.12}$$

On the whole boundary $\partial\Omega = \Gamma_- \cup \Gamma_+ \cup \Gamma_0$, we assume that the flux is non-diffusive for convection dominant transport:

$$\vec{n} \cdot (-\mathcal{D}\nabla c) \;=\; 0 \quad \text{on} \quad \Gamma_- \cup \Gamma_+ \cup \Gamma_0 \tag{4.13}$$

This implies the no-flux condition for impermeable boundaries:

$$\vec{n} \cdot (-\mathcal{D}\nabla c + \vec{q}c) \;=\; 0 \quad \text{on} \quad \Gamma_0 \tag{4.14}$$

Boundary value problem for the temporal moments

We are interested in the steady-state solution ($\partial c/\partial t = 0$), or more adequately, in the zeroth or first order temporal moments of the resident concentration (4.7). Naturally, c has compact support in time and space:

- the input of tracer is not endless,

- the injection space is restricted to the space occupied by the injection well or is a limited part of the inflow boundary.

In the formal derivation of the differential equations for the temporal moments, we further assume that c is always smooth enough such that the strong form of the differential operators exists and such that the order for the two operations *differentiation in space* and *integration in time* can be exchanged.

Definition 4.1. *Let $s \geq 0$ and $c \in C^1(0,T;C^s(\Omega))$*[1]*. Furthermore, let $c(t,\vec{x}) = 0$ for $t > T$. The k-**th temporal moment** of c is defined by*

$$\mathscr{M}_k[c](\vec{x}) := m_k^c(\vec{x}) = \int\limits_{t=0}^{\infty} t^k c(t,\vec{x})\, dt. \tag{4.15}$$

From the definition of c we conclude that $m_k^c < \infty$ and $m_k^c \in C^s(\Omega)$. [Harvey and Gorelick, 1995] uses the Laplace transform to derive the moment equations from (4.7). We apply the operator $\mathscr{M}_k$ directly, term by term, onto the differential equation (4.7). For the first term, we use integration by parts:

$$\mathscr{M}_k\left[\frac{\partial(\theta c(t,\vec{x}))}{\partial t}\right] = -\int\limits_{t=0}^{\infty} k t^{k-1}\theta c(t,\vec{x})\, dt + \underbrace{\Big[t^k c(t,\vec{x})\Big]_0^{\infty}}_{=0-0} = -k\theta m_{k-1}^c(\vec{x}) \tag{4.16}$$

Due to the time-independence of the spatial differential operators, the Darcy velocity $\vec{q}$ and the diffusion tensor $\mathcal{D}$, these operations and factors can be drawn out of the temporal integral in the remaining terms of the left hand side of (4.7):

$$\begin{aligned}\mathscr{M}_k\Big[\mathrm{div}\big(-\mathcal{D}\nabla c(t,\vec{x}) + \vec{q}\,c(t,\vec{x})\big) + \tilde{w}_{\text{ext}}\,c(t,\vec{x})\Big] \\ = \mathrm{div}\big(-\mathcal{D}\nabla m_k^c(\vec{x}) + \vec{q}\,m_k^c(\vec{x})\big) + \tilde{w}_{\text{ext}} m_k^c(\vec{x})\end{aligned} \tag{4.17}$$

Likewise, the boundary term in (4.13) becomes

$$\mathscr{M}_k\Big[\vec{n}\cdot(-\mathcal{D}\nabla c)\Big] = \vec{n}\cdot(-\mathcal{D}\nabla m_k^c). \tag{4.18}$$

[1] $C^1(0,T;C^s(\Omega))$ be the space of functions which are continuously differentiable with respect to $t \in (0,T)$ and s-times continuously differentiable with respect to $\vec{x} \in \Omega$.

Assuming that we inject a tracer in a constant concentration $\tilde{c}_{\text{inj}}$ at a constant rate $\tilde{w}_{\text{inj}}$ through a well over a fixed period of time $T_{\text{inj}} > 0$, the right hand side term becomes

$$\begin{aligned} \mathscr{M}_k\left[\tilde{w}_{\text{inj}}\tilde{c}_{\text{inj}}\right] &= \tilde{w}_{\text{inj}} \cdot \underbrace{{}^{(T_{\text{inj}})^{k+1}}\!/\!_{(k+1)} \cdot \tilde{c}_{\text{inj}}}_{\downarrow} \\ &=: \tilde{w}_{\text{inj}} \cdot \tilde{m}_k^{\text{inj}} \end{aligned} \tag{4.19}$$

In the same manner, we get

$$\mathscr{M}_k\left[\hat{c}_D\right] = {}^{(T_{\text{inj}})^{k+1}}\!/\!_{(k+1)} \cdot \hat{c}_D =: \hat{m}_k^c \tag{4.20}$$

for (4.12) on the inflow boundary.

Summing up (4.16)-(4.19), we obtain the *steady-state transport equations* for m_k^c:

$$\operatorname{div}(-\mathcal{D}\nabla m_k^c + \vec{q}\, m_k^c) + \tilde{w}_{\text{ext}}\, m_k^c = \tilde{w}_{\text{inj}}\, \tilde{m}_k^{\text{inj}} + k\theta m_{k-1}^c \qquad \text{in } \Omega, \tag{4.21}$$

subject to the boundary conditions:

$$m_k^c = \hat{m}_k^c \qquad \text{on } \Gamma_- \tag{4.22}$$

$$\vec{n} \cdot (-\mathcal{D}\nabla m_k^c) = 0 \qquad \text{on } \Gamma \tag{4.23}$$

Physical meaning of the temporal moments

The zeroth moment at an observation location $\vec{x}_\ell$ is the total tracer mass that passes through a cross-sectional area $\delta A(\vec{x}_\ell)$, divided by the volumetric discharge through $\vec{x}_\ell$, $q_A(\vec{x}_\ell) = \delta A \cdot \vec{q} \cdot \vec{n}_A$. In other words, the product $m_0^c(\vec{x}_\ell) \cdot q_A(\vec{x}_\ell)$ is the accumulated mass passing $\vec{x}_\ell$.
The first moment divided by the zeroth moment of solute concentration at $\vec{x}_\ell$ is the mean arrival time of the solute:

$$\overline{T}_{\text{arrival}}(\vec{x}_\ell) = \frac{m_1^c(\vec{x}_\ell)}{m_0^c(\vec{x}_\ell)}. \tag{4.24}$$

For convection dominant transport with very small transversal dispersivities α_t, the plot of the zeroth moment m_0^c in a 2-D simulation shows almost a binary distribution revealing only the shape of the plume which is typically a connected region permeated by the solute, lying in a subdomain surrounding the wells (see Figure 7.8). Inside the plume, m_0^c assumes approximately the value at the injection location and, outside the plume, $m_0^c = 0$. In 2-D, the information content is not very high, whereas in 3-D, the permeated region may encapsulate regions which are much less permeable: A cross-sectional plot of a 3-D simulation of m_0^c for the original Y-field in Figure 7.12 (or Figure 7.13) reveals those obstructions. The first moment or the mean arrival time is inversely proportional to the hydraulic conductivity (Darcy's Law) averaged over the travel path from the injection to the observation point and monotonically increasing with travel distance. It contains much more information about the structure of the permeated region.

4.1.3 Weak formulation of coupled equations

We seek the solutions $\{s, t, u\} := \{\phi, m_0^c, m_1^c\}$ of (4.1) and (4.21) in the Sobolev space $H^1(\Omega)$ and we assume that the logarithm of the hydraulic conductivity $Y \in L^2(\Omega)$. In the following, we will use the abbreviations

$$\mathcal{H}_Y := L^2(\Omega) \quad \text{and} \quad \mathcal{H}_s := \mathcal{H}_t := \mathcal{H}_u := H^1(\Omega) \tag{4.25}$$

to improve the readability. Note that the dual space of $H^1(\Omega)$ is denoted by $H^1(\Omega)^*$.

Groundwater equation

Let $\hat{s}_D \in H^1(\Omega)$ be the extension of $\hat{\phi}_D \in H^{1/2}(\Gamma_D)$. The trace theorem guarantees the existence of a trace operator $\gamma_s : H^1(\Omega) \ni \hat{s}_D \mapsto \hat{\phi}_D \in L^2(\Gamma_D)$ [Braess, 2007]. Let

$$S_D := \{v \in H^1(\Omega) : \gamma_s v = 0 \text{ on } \Gamma_D\} \, . \tag{4.26}$$

Let $G : \mathcal{H}_Y \times \mathcal{H}_s \longrightarrow \mathcal{H}_s^*$ be a bounded operator (linear with respect to s) defined by the duality pairing of $\mathcal{H}_s$ and $\mathcal{H}_s^*$:

$$\Big\langle \psi^s, \, G(Y, s) \Big\rangle_{\mathcal{H}_s, \mathcal{H}_s^*} := \big(\nabla\psi^s, \, K(Y)\nabla s \,\big)_{0,\Omega} - \big(\psi^s, \, \tilde{w}_{\text{inj}} - \tilde{w}_{\text{ext}} \big)_{0,\Omega} \tag{4.27}$$

The extra dependency on the parameter Y will play an important role later in §4.2.2 in the derivation of the sensitivities. The weak formulation of the BVP (4.1) reads:

Find $s \in \hat{s}_D + S_D$ such that

$$\Big\langle \psi^s, \, G(Y, s) \Big\rangle_{\mathcal{H}_s, \mathcal{H}_s^*} = 0 \qquad \forall \, \psi^s \in S_D \, , \tag{4.28}$$

Transport equations

Similarly, for the zeroth order temporal moment of concentration, let $\hat{t}_D \in H^1(\Omega)$ be the extension of $\hat{m}_0^c \in H^{1/2}(\Gamma_-)$ (with trace operator γ_t). Let

$$T_D := \{v \in H^1(\Omega) : \gamma_t v = 0 \text{ on } \Gamma_-\} \, . \tag{4.29}$$

Let $M_0 : \mathcal{H}_Y \times \mathcal{H}_s \times \mathcal{H}_t \longrightarrow \mathcal{H}_t^*$ be a bounded operator (linear in t) defined via the duality pairing of $\mathcal{H}_t$ and $\mathcal{H}_t^*$

$$\begin{aligned} \Big\langle \psi^t ,\ M_0(Y,s,t) \Big\rangle_{\mathcal{H}_t,\mathcal{H}_t^*} := & \left(\nabla\psi^t ,\ \mathcal{D}\nabla t - \vec{q}\,t \right)_{0,\Omega} + \left(\psi^t , \vec{n}\cdot\vec{q}\,t \right)_{0,\Gamma} \\ & + \left(\psi^t ,\ \tilde{w}_{\text{ext}}\, t - \tilde{w}_{\text{inj}}\, \tilde{m}_0^{\text{inj}} \right)_{0,\Omega} , \end{aligned} \tag{4.30}$$

keeping in mind that

$$\vec{q} = \vec{q}\,(Y,s) = -K(Y)\nabla s \,.$$

The weak form of the BVP (4.21), $k = 0$, reads:

Find $t \in \hat{t}_D + T_D$ such that

$$\Big\langle \psi^t ,\ M_0(Y,s,t) \Big\rangle_{\mathcal{H}_t,\mathcal{H}_t^*} = 0 \qquad \forall\, \psi^t \in T_D \,, \tag{4.31}$$

Analogously, for the first order temporal moment of concentration, let $\hat{u}_D \in H^1(\Omega)$ be the extension of $\hat{m}_1^c \in H^{1/2}(\Gamma_-)$ (with trace operator γ_u) and

$$U_D := \{ v \in H^1(\Omega) : \gamma_u v = 0 \text{ on } \Gamma_- \} \,. \tag{4.32}$$

Let $M_1 : \mathcal{H}_Y \times \mathcal{H}_s \times \mathcal{H}_t \times \mathcal{H}_u \longrightarrow \mathcal{H}_u^*$ be a bounded operator (linear in u) be defined by the duality pairing of $\mathcal{H}_u$ and $\mathcal{H}_u^*$

$$\begin{aligned} \Big\langle \psi^u ,\ M_1(Y,s,t,u) \Big\rangle_{\mathcal{H}_u,\mathcal{H}_u^*} := & \left(\nabla\psi^u ,\ \mathcal{D}\nabla u - \vec{q}\,u \right)_{0,\Omega} + \left(\psi^u , \vec{n}\cdot\vec{q}\,u \right)_{0,\Gamma} \\ & + \left(\psi^u ,\ \tilde{w}_{\text{ext}}\, u - \tilde{w}_{\text{inj}}\, \tilde{m}_1^{\text{inj}} - \theta t \right)_{0,\Omega} \end{aligned} \tag{4.33}$$

The weak form of the BVP (4.21), $k = 1$, reads:

Find $u \in \hat{u}_D + U_D$ such that

$$\Big\langle \psi^u ,\ M_1(Y,s,t,u) \Big\rangle_{\mathcal{H}_u,\mathcal{H}_u^*} = 0 \qquad \forall\, \psi^u \in U_D \,, \tag{4.34}$$

Remark 4.1. *In each of the weak formulations shown here, the Dirichlet boundary condition is built into the solution space. This option is convenient when working with standard Galerkin finite elements and sufficient for the formulation of the BVPs as PDE constraints later in the derivation of the adjoint equations. When working with discontinuous Galerkin (DG) methods, Dirichlet boundary conditions are imposed weakly and the same solution space can be chosen for both the solution and the test function [Arnold et al., 2002; Nitsche, 1971]. Solutions of DG methods are sought in discrete subspaces of the so-called broken Sobolev spaces $H^s(\mathcal{T}_h)$ (with integer s) which depend on the mesh $\mathcal{T}_h$ [Rivière, 2008].*

4.2 Sensitivities and cross-covariance matrices

The evaluation of the sensitivity matrix (3.38)

$$[\mathcal{H}]_{\ell j} = \left[\frac{\partial f_\ell(\boldsymbol{y})}{\partial y_j}\right]_{\ell j} = \left[\left.\frac{\partial u(\boldsymbol{y})}{\partial y_j}\right|_{\vec{x}=\vec{x}_\ell}\right]_{\ell j} \quad \text{for} \quad \ell \in \{1, ..., M\}, j \in \{1, ..., N\}$$

and the linearized cross-covariance matrix (N is very large)

$$\mathcal{R}_{YY}\mathcal{H}^T$$

for the cokriging system (3.49) and the prior term (3.58) of the objective function are the computationally most demanding tasks in each iteration of the (discretized) inversion algorithm.

4.2.1 Computation of sensitivities

An overview of methods to compute the partial derivatives of the model function (3.19) with respect to the model parameters for general non-linear inverse problems can be found in the article of McGillivray and Oldenburg [1990] or in §2.6 of the monograph of Uciński [2004]. Some of the most important methods are

- the finite difference approximation:

$$\frac{\partial f_\ell(\boldsymbol{y})}{\partial y_j} \approx \left.\frac{u(\boldsymbol{y} + \varepsilon_j \boldsymbol{e}_j) - u(\boldsymbol{y})}{\varepsilon_j}\right|_{\vec{x}=\vec{x}_\ell} \tag{4.35}$$

  where $\boldsymbol{e}_j$ is a canonical basis vector and the choice $\varepsilon_j = \sqrt{eps}$ is optimal (c.f. [Nocedal and Wright, 2006], §8.1), where eps is the unit round-off ($eps = 10^{-16}$ for double-precision). u must be computed once for $\boldsymbol{y}$ and once for each direction $\boldsymbol{e}_j$ ($j = 1, ..., N$). In total, this amounts to the solution of $(N + 1)$ forward problems.

- the automatic (or algorithmic) differentiation method (AD): Given an algorithm that computes values of a function, this technique generates (step-by-step) an algorithm that computes derivative values of the function with respect to specified parameters. It is based on the observation that any function, no matter how complicated, can be split up into a sequence of elementary operations involving one or two arguments at a time (c.f. [Nocedal and Wright, 2006], §8.2). Repeated application of the chain rule to the composition of these elementary operations yields derivative information that are exact to machine precision. A selected list of AD software can be found on the website `http://www.autodiff.org`.

One of the main difficulties in using AD tools is the need to analyze and debug generated code that is not written by oneself. Other pitfalls are discussed by Rall and Corliss [1996].

- the adjoint equation method: We will see that this method is by far superior in terms of computational cost, if $N \gg M$. This method of choice is discussed in-depth by Cirpka and Kitanidis [2001] and Sun [1994] using *small perturbation theory*. We will show an alternative derivation of the very same adjoint equations using the *Lagrange formalism*.

4.2.2 Adjoint equation method

We apply the Lagrangian formalism to the adjoint approach [Hinze et al., 2009] for the derivation of the gradient of the quantity u with respect to the parameter Y taken at *a single* measuring point $\vec{x}_\ell$. The approach is based on defining the cost functional for the minimization of the state u itself (without actually solving the minimization problem). Following the principle *"First optimize, then discretize!"*, the discretization of Y (2.11) is postponed to the very end, whereas (4.35) is a direct approximation of the gradient with respect to the discretized parameter field $\boldsymbol{y}$.

We will explain this procedure in detail for the first[1] order moment of solute concentration: m_1^c is indirectly depending on the parameter $Y = \ln(K)$ through a chain of dependent **solution operators**

$$Y \xrightarrow{s} \phi \xrightarrow{t} m_0^c \xrightarrow{u} m_1^c \tag{4.36}$$

provided that the forward problems from section §4.1.3 can be solved. Let

$$\begin{array}{rcllcl}
Y & & & \in & L^2(\Omega) & (\text{ parameter space }) \\
\phi & = & s(Y) & \in & s_D + S_D \subset H^1(\Omega) & (\text{ solution space for } \phi \text{ }) \\
m_0^c & = & t(Y, s(Y)) & \in & t_D + T_D \subset H^1(\Omega) & (\text{ solution space for } m_0^c \text{ }) \\
m_1^c & = & u(Y, s(Y), t(Y, s(Y))) & \in & u_D + U_D \subset H^1(\Omega) & (\text{ solution space for } m_1^c \text{ })
\end{array}$$

fulfill the coupled boundary value problems (4.28), (4.31) and (4.34).
The goal is to get a computable term for the expression

$$u'(Y)\Big|_{\vec{x}=\vec{x}_\ell} . \tag{4.37}$$

[1]Note that the variable u is identified with m_1^c ONLY in this subsection.

Let $\varepsilon > 0$ and let $B_\varepsilon(\vec{x}_\ell) := \{\vec{x} \in \mathbb{R}^d : \|\vec{x} - \vec{x}_\ell\|_2 < \varepsilon\}$ be an open ball around a measuring point $\vec{x}_\ell$. Let

$$\eta_\varepsilon^\ell(\vec{x}) := \frac{1}{(\varepsilon\sqrt{2\pi})^d} \exp\left(-\frac{\|\vec{x} - \vec{x}_\ell\|_2^2}{2\varepsilon^2}\right) \tag{4.38}$$

represent the volume-averaged approximation of a point source. This function approaches the Dirac delta distribution $\delta(\vec{x} - \vec{x}_\ell)$ as $\varepsilon \longrightarrow 0$. We consider the "cost" functional

$$\mathscr{J}(s,t,u,Y) := \overline{u}(\vec{x}_\ell) = \left(\, \eta_\varepsilon^\ell,\ u \,\right)_{0,\Omega} = \int\limits_\Omega \eta_\varepsilon^\ell(\vec{x})\, u(\vec{x})\, d\Omega \tag{4.39}$$

realizing a volume-averaged measurement of a quantity u that fulfills the coupled PDE constraints (4.28), (4.31) and (4.34).

The **Lagrange function** for (4.39) is given by

$$\begin{aligned} L : (\mathcal{H}_s \times \mathcal{H}_t \times \mathcal{H}_u) \times \mathcal{H}_Y \times (\mathcal{H}_s \times \mathcal{H}_t \times \mathcal{H}_u) &\longrightarrow \mathbb{R} \\ L(s,t,u,Y,\psi^s,\psi^t,\psi^u) :=& \mathscr{J}(s,t,u,Y) + \Big\langle \psi^s, G(Y,s) \Big\rangle_{\mathcal{H}_s,\mathcal{H}_s^*} \\ &+ \Big\langle \psi^t, M_0(Y,s,t) \Big\rangle_{\mathcal{H}_t,\mathcal{H}_t^*} + \Big\langle \psi^u, M_1(Y,s,t,u) \Big\rangle_{\mathcal{H}_u,\mathcal{H}_u^*} \end{aligned} \tag{4.40}$$

with **Lagrange multipliers** ψ^s, ψ^t, ψ^u. Due to the PDE constraints, the Lagrange function L coincides with the cost functional $\mathscr{J}$. Thus, the **reduced cost functional** is

$$\widehat{\mathscr{J}}(Y) := L(\,Y,\ s(Y),\ t(Y,s(Y)),\ u(Y,s(Y),t(Y,s(Y))),\ \psi^s,\ \psi^t,\ \psi^u\,)\,, \tag{4.41}$$

which essentially depends on Y.

In our case, the **sensitivity** of u can be computed as the directional derivative of the reduced cost functional with respect to Y. Applying the chain rule, we get for an *arbitrarily chosen* direction $\delta Y \in \mathcal{H}_Y$:

$$\begin{aligned} \widehat{\mathscr{J}}'(Y)(\delta Y) = L'_Y(\delta Y) + \left\{L'_s + L'_t t'_s + L'_u(u'_s + u'_t t'_s)\right\}\big(\underbrace{s'(Y)(\delta Y)}_{=:\,\delta s}\big) \\ + \left\{L'_t + L'_u u'_t\right\}\big(\underbrace{t'_Y(\delta Y)}_{=:\,\delta t}\big) \\ + L'_u\big(\underbrace{u'_Y(\delta Y)}_{=:\,\delta u}\big)\,. \end{aligned} \tag{4.42}$$

If we choose the Lagrange multipliers ψ^s, ψ^t and ψ^u in such a way that the **Karush-Kuhn-Tucker** conditions for the **adjoint equations**[1]

$$\begin{aligned} L'_u\big(\delta u\big) &= 0 && (4.43)\\ L'_t\big(\delta t\big) &= 0 && (4.44)\\ L'_s\big(\delta s\big) &= 0 && (4.45) \end{aligned}$$

hold, all that remains for the sensitivity (4.42) will be

$$\widehat{\mathscr{J}}'(Y)\big(\delta Y\big) = L'_Y\big(\delta Y\big). \tag{4.46}$$

The Lagrange multipliers and the directions δs, δt and δu need to be chosen from appropriately defined subspaces of $H^1(\Omega)$ which will be explained below.

Let us first compute the partial derivatives in (4.43)-(4.45). In the calculation of these functional derivatives, we make use of

(I) Green's theorem (E.8),

(II) the fact that all terms from the PDE constraints without contribution to the derivative vanish.

Furthermore,

(III) we neglect the dependencies between the Scheidegger Dispersion tensor (4.9) and the parameter Y and the hydraulic head $s = \phi$:

$$\frac{\partial \mathcal{D}}{\partial Y} = 0 \quad \text{and} \quad \frac{\partial \mathcal{D}}{\partial s} = 0\,.$$

(IV) Since $K = K(Y) = \exp(Y)$, we have:

$$\frac{\partial K}{\partial Y} = K'(Y) = K \quad \text{and} \quad \frac{\partial \vec{q}}{\partial Y} = \frac{\partial(-K\nabla s)}{\partial Y} = -K\nabla s = \vec{q}\,.$$

The adjoint equation (4.43) corresponding to $u = m_1^c$ reads:

$$L'_u(\delta u) = \mathscr{J}'_u(\delta u) + D_u\Big\langle \psi^u, M_1(Y,s,t,u)\Big\rangle_{\mathcal{H}_u,\mathcal{H}_u^*}(\delta u) \tag{4.47}$$

Clearly,

$$\mathscr{J}'_u(\delta u) = D_u\Big[\big(\,\eta_\varepsilon^\ell,\ u\,\big)_{0,\Omega}\Big](\delta u) \overset{(E.25)}{=} \big(\eta_\varepsilon^\ell, \delta u\big)_{0,\Omega} \tag{4.48}$$

[1]The reverse order ($u \to t \to s$) of the adjoint equations becomes evident in (4.42).

We split up the derivative of the duality pairing (4.33):

$$D_u\Big[\big(\,\nabla\psi^u,\ \mathcal{D}\nabla u\,\big)_{0,\Omega}\Big](\delta u) \overset{(I)}{=} D_u\Big[\big(\mathrm{div}(-\mathcal{D}\nabla\psi^u),\ u\,\big)_{0,\Omega} + \big(\vec{n}\cdot\mathcal{D}\nabla\psi^u,\ u\,\big)_{0,\Gamma}\Big](\delta u)$$
$$\overset{(E.25)}{=} \big(\mathrm{div}(-\mathcal{D}\nabla\psi^u),\ \delta u\big)_{0,\Omega} + \underbrace{\big(\vec{n}\cdot\mathcal{D}\nabla\psi^u,\ \delta u\big)_{0,\Gamma\setminus\Gamma-}}_{u=\hat{u}_D \text{ on } \Gamma-} \tag{4.49}$$

$$D_u\Big[\big(\,\nabla\psi^u,\ -\vec{q}\,u\,\big)_{0,\Omega} + \big(\,\psi^u,\ \vec{n}\cdot\vec{q}\,u\,\big)_{0,\Gamma}\Big](\delta u)$$
$$\overset{(E.25)}{=} \big(\,-\vec{q}\,\nabla\psi^u,\ \delta u\,\big)_{0,\Omega} + \big(\,\vec{n}\cdot\vec{q}\,\psi^u,\ \delta u\,\big)_{0,\Gamma}$$
$$\overset{(4.1)}{=} \big(\,\mathrm{div}(-\vec{q}\,\psi^u) + (\tilde{w}_{\text{inj}} - \tilde{w}_{\text{ext}})\,\psi^u,\ \delta u\,\big)_{0,\Omega} + \underbrace{\big(\vec{n}\cdot\vec{q}\,\psi^u,\ \delta u\big)_{0,\Gamma\setminus\Gamma-}}_{u=\hat{u}_D \text{ on } \Gamma-} \tag{4.50}$$

$$D_u\Big[\big(\,\psi^u,\ \tilde{w}_{\text{ext}}\,u - \tilde{w}_{\text{inj}}\,\tilde{m}_1^{\text{inj}} - \theta t\,\big)_{0,\Omega}\Big](\delta u) = \big(\,\tilde{w}_{\text{ext}}\psi^u,\ \delta u\,\big)_{0,\Omega} \tag{4.51}$$

Summing up, we obtain

$$\big(\,\eta_\varepsilon^\ell + \mathrm{div}\big(-\mathcal{D}\nabla\psi^u - \vec{q}\,\psi^u\big) + \tilde{w}_{\text{inj}}\,\psi^u,\ \delta u\,\big)_{0,\Omega} + \big(\,\vec{n}\cdot\vec{q}\,\psi^u,\ \delta u\,\big)_{0,\Gamma\setminus\Gamma-} \overset{!}{=} 0 \tag{4.52}$$

if we additionally require the diffusive flux $\vec{n}\cdot(\mathcal{D}\nabla\psi^u)$ to vanish on all boundaries. In this weak formulation of a new BVP, the Lagrange multiplier ψ^u becomes the weak solution and the term δu plays the role of a test function. The convective term has a reversed Darcy velocity $-\vec{q}$. Hence, inflow and outflow boundaries swap their roles. ψ^u need not be prescribed on the outflow boundary. Therefore, it makes sense to define

$$U_D^+ := \{v \in H^1(\Omega) : \gamma_u^+ v = 0 \text{ on } \Gamma_+\} \tag{4.53}$$

(with a trace operator $\gamma_u^+ : H^1(\Omega) \longrightarrow L^2(\Gamma_+)$) and seek the solution $\psi^u \in U_D^+$ such that (4.52) holds for all $\delta u \in U_D^+$.

Hence, the adjoint equation (4.43) is necessarily fulfilled by the solution $\psi_\ell^{m_1} := \psi^u$ (called the **adjoint state**) of the BVP:

$$\boxed{\begin{array}{rcll} \mathrm{div}(-\mathcal{D}\nabla\psi_\ell^{m_1} - \vec{q}\,\psi_\ell^{m_1}) + \tilde{w}_{\text{inj}}\,\psi_\ell^{m_1} & = & -\eta_\varepsilon^\ell & \text{in } \Omega \\ \psi_\ell^{m_1} & = & 0 & \text{on } \Gamma_+ \\ \vec{n}\cdot(-\mathcal{D}\nabla\psi_\ell^{m_1}) & = & 0 & \text{on } \Gamma \end{array}} \tag{4.54}$$

Remark 4.2. *An injection well with rate $\tilde{w}_{inj}$ in the forward transport equation becomes an extraction well with pumping rate $-\tilde{w}_{inj} < 0$ in the adjoint transport equation, whereas an extraction well with $\tilde{w}_{ext}$ becomes a fresh water injection well.*

The adjoint equation (4.44) corresponding to $t = m_0^c$ reads:

$$L_t'(\delta t) \;=\; D_t\Big\langle\, \psi^t,\ M_0(Y,s,t)\,\Big\rangle_{\mathcal{H}_t^*,\mathcal{H}_t}(\delta t) + D_t\Big\langle\, \psi^u,\ M_1(Y,s,t,u)\,\Big\rangle_{\mathcal{H}_u^*,\mathcal{H}_u}(\delta t) \tag{4.55}$$

Following the same argumentation as for (4.52), we obtain the BVP:
Find $\psi^t \in U_D^+$ such that

$$\big(\,\mathrm{div}(-\mathcal{D}\nabla\psi^t - \vec{q}\,\psi^t) + \tilde{w}_{\text{inj}}\,\psi^t - \theta\psi^u,\ \delta t\,\big)_{0,\Omega} + \big(\vec{n}\cdot\vec{q}\,\psi^t,\ \delta t\,\big)_{0,\Gamma\setminus\Gamma_-} \overset{!}{=} 0 \quad \forall \delta t \in U_D^+ \tag{4.56}$$

Therefore, the solution $\psi_\ell^{m_0} := \psi^t$ of the BVP

$$\boxed{\begin{array}{rcll} \mathrm{div}(-\mathcal{D}\nabla\psi_\ell^{m_0} - \vec{q}\,\psi_\ell^{m_0}) + \tilde{w}_{\text{inj}}\,\psi_\ell^{m_0} & = & \theta\psi_\ell^{m_1} & \text{in } \Omega \\ \psi_\ell^{m_0} & = & 0 & \text{on } \Gamma_+ \\ \vec{n}\cdot(-\mathcal{D}\nabla\psi_\ell^{m_0}) & = & 0 & \text{on } \Gamma \end{array}} \tag{4.57}$$

fulfills (4.44).

The adjoint equation (4.45) corresponding to $s = \phi$ reads (where we use $\vec{q} = -K\nabla s$):

$$\begin{aligned} L_s'(\delta s) &= D_s\Big\langle \psi^s,\ G(Y,s)\Big\rangle_{\mathcal{H}_s,\mathcal{H}_s^*}(\delta s) \\ &\quad + D_s\Big\langle \psi^t,\ M_0(Y,s,t)\Big\rangle_{\mathcal{H}_t,\mathcal{H}_t^*}(\delta s) + D_s\Big\langle \psi^u,\ M_1(Y,s,t,u)\Big\rangle_{\mathcal{H}_u,\mathcal{H}_u^*}(\delta s) \end{aligned} \tag{4.58}$$

First term:

$$D_s\Big\langle \psi^s,\ G(Y,s)\Big\rangle_{\mathcal{H}_s,\mathcal{H}_s^*}(\delta s) = D_s\Big[\big(\psi^s, K\nabla s\,\big)_{0,\Omega}\Big](\delta s) - \underbrace{D_s\Big[\big(\,\psi^s,\ (\tilde{w}_{\text{inj}} - \tilde{w}_{\text{ext}})\,\big)_{0,\Omega}\Big](\delta s)}_{=0}$$

$$\overset{\text{(I)}}{=} D_s\Big[\big(\text{div}(-K\nabla\psi^s),\ s\,\big)_{0,\Omega}\Big](\delta s) - \underbrace{D_s\Big[\big(\,\vec{n}\cdot(-K\nabla\psi^s),\ s\,\big)_{0,\Gamma}\Big](\delta s)}_{s=\hat{s}_D \text{ on } \Gamma_D}$$

$$\overset{(E.25)}{=} \big(\,\text{div}(-K\nabla\psi^s),\ \delta s\,\big)_{0,\Omega} - \big(\,\vec{n}\cdot(-K\nabla\psi^s),\ \delta s\,\big)_{0,\Gamma_N} \tag{4.59}$$

Second term:

$$D_s\Big\langle \psi^t,\ M_0(Y,s,t)\Big\rangle_{\mathcal{H}_t,\mathcal{H}_t^*}(\delta s) \overset{(III)}{\overset{\downarrow}{\approx}} D_s\Big[\big(\,\psi^t,\ \text{div}(\vec{q}\,t)\,\big)_{0,\Omega}\Big](\delta s)$$

$$\overset{(4.1)}{=} D_s\Big[\big(\,\psi^t,\ -K\nabla s\,\nabla t + (\tilde{w}_{\text{inj}} - \tilde{w}_{\text{ext}})\,t\,\big)_{0,\Omega}\Big](\delta s) = D_s\Big[\big(\,-\psi^t K\nabla t,\ \nabla s\,\big)_{0,\Omega}\Big](\delta s)$$

$$\overset{\text{(I)}}{=} D_s\Big[-\big(\text{div}(-\psi^t K\nabla t),\ s\,\big)_{0,\Omega}\Big](\delta s) + \underbrace{D_s\Big[\big(\vec{n}\cdot(-\psi^t K\nabla t),\ s\,\big)_{0,\Gamma}\Big](\delta s)}_{s=\hat{s}_D \text{ on } \Gamma_D} \tag{4.60}$$

$$= -\big(\text{div}(-\psi^t K\nabla t), \delta s\,\big)_{0,\Omega} + \big(\vec{n}\cdot(-\psi^t K\nabla t),\ \delta s\,\big)_{0,\Gamma_N}$$

The third term is calculated analogously to the second term. If we choose ψ^s in such a way that it vanishes on the Dirichlet boundary Γ_D, then the sum of the three terms yields the BVP:

Find $\psi^s \in S_D$ such that

$$\begin{aligned} &\big(\,\text{div}(-K\nabla\psi^s) - \text{div}(-\psi^t K\nabla t - \psi^u K\nabla u),\ \delta s\,\big)_{0,\Omega} \\ &- \big(\big\{\vec{n}\cdot(-K\nabla\psi^s) - \vec{n}\cdot(-\psi^t K\nabla t - \psi^u K\nabla u)\big\},\ \delta s\,\big)_{\Gamma_N} \overset{!}{=} 0 \quad \delta s \in S_D\,. \end{aligned} \tag{4.61}$$

The adjoint equation (4.45) is fulfilled by the solution $\psi_\ell^\phi := \psi^s$ of the BVP:

$$\boxed{\begin{array}{rcll} \operatorname{div}(-K\nabla\psi_\ell^\phi) &=& \operatorname{div}(-\psi_\ell^{m_0}K\nabla m_0^c - \psi_\ell^{m_1}K\nabla m_1) & \text{in } \Omega \\ \psi_\ell^\phi &=& 0 & \text{on } \Gamma_D \\ \vec{n}\cdot(-K\nabla\psi_\ell^\phi) &=& \vec{n}\cdot(-\psi_\ell^{m_1}K\nabla m_0^c - \psi_\ell^{m_1}K\nabla m_1) & \text{on } \Gamma_N \end{array} \qquad (4.62)}$$

where we have used $t = m_0^c$, $u = m_1^c$, $\psi^t = \psi_\ell^{m_0}$ and $\psi^u = \psi_\ell^{m_1}$.

The sensitivity is given by

$$\begin{aligned} &L'_Y(\delta Y) = \\ &\quad D_Y\Big\langle \psi^s, G(Y,s)\Big\rangle_{\mathcal{H}_s,\mathcal{H}_s^*}(\delta Y) + D_Y\Big\langle \psi^t, M_0(Y,s,t)\Big\rangle_{\mathcal{H}_t,\mathcal{H}_t^*}(\delta Y) \\ &\qquad\qquad + D_Y\Big\langle \psi^u, M_1(Y,s,t,u)\Big\rangle_{\mathcal{H}_u,\mathcal{H}_u^*}(\delta Y) \\ &\overset{(III)}{\overset{\downarrow}{\approx}} D_Y\big(\nabla\psi^s,\ K\nabla s\big)_{0,\Omega}(\delta Y) + D_Y\big(\psi^t,\ \vec{q}\,\nabla t\big)_{0,\Omega}(\delta Y) + D_Y\big(\psi^u,\ \vec{q}\,\nabla u\big)_{0,\Omega}(\delta Y) \\ &\overset{(IV)}{=} \big(\nabla\psi^s\,(-\vec{q}),\ \delta Y\big)_{0,\Omega} - \big(\psi^t\,\nabla t\,(-\vec{q}),\ \delta Y\big)_{0,\Omega} - \big(\psi^u\,\nabla u\,(-\vec{q}),\ \delta Y\big)_{0,\Omega}\,. \end{aligned} \tag{4.63}$$

Finally, inserting back $s = \phi$, $t = m_0^c$, $u = m_1^c$ and $\psi^s = \psi_\ell^\phi$, $\psi^t = \psi_\ell^{m_0}$, $\psi^u = \psi_\ell^{m_1}$, the sensitivity of the state variable $u = m_1^c$ with respect to the parameter Y reads:

$$\boxed{\begin{aligned} D_Y[m_1^c(Y)](\delta Y)\Big|_{\vec{x}=\vec{x}_\ell} &= L'_Y(\delta Y) \\ &= \int_\Omega \Big\{\big(\nabla\psi_\ell^\phi - \psi_\ell^{m_0}\,\nabla m_0^c - \psi_\ell^{m_1}\,\nabla m_1^c\big)\cdot(-\vec{q}\,)\Big\}\cdot\delta Y\,d\Omega \end{aligned} \qquad (4.64)}$$

4.2.3 Discretized sensitivities

A cell-wise discretization (2.11) of Y into $\boldsymbol{y} \in \mathbb{R}^N$ yields a representation of the sensitivity (4.64) in the canonical basis of $\mathbb{R}^N$. We are interested in the gradient of m_1^c which is made of partial derivatives with respect to the components of $\boldsymbol{y}$. Therefore, choosing the constant direction

$$\delta Y(\vec{x}) = \begin{cases} \delta Y_j = 1 & \vec{x} \in t_j \\ 0 & \vec{x} \in \Omega \backslash t_j \end{cases} \tag{4.65}$$

where t_j is the j-th cell of the structured mesh $\mathcal{T}_h$ (2.10), we obtain

$$\boxed{\left.\frac{\partial m_1^c(\boldsymbol{y})}{\partial y_j}\right|_{\vec{x}=\vec{x}_\ell} = \int_{t_j} \left\{ \left(\nabla \psi_\ell^\phi - \psi_\ell^{m_0} \nabla m_0^c - \psi_\ell^{m_1} \nabla m_1^c\right) \cdot (K \nabla \phi) \right\} d\Omega} \tag{4.66}$$

for the j-th component of the discrete gradient with respect to $\boldsymbol{y}$. We call the expression

$$\left.\frac{\partial m_1^c(\boldsymbol{y})}{\boldsymbol{y}}\right|_{\vec{x}=\vec{x}_\ell}$$

the **sensitivity field of** m_1^c at the location $\vec{x}_\ell$ in order to avoid confusion with the gradient ∇m_1^c.

Remark 4.3. *When monitoring m_0^c, we set $\psi_\ell^{m_1} = 0$ in the BVPs (4.57) and (4.62) for the adjoint states $\psi_\ell^{m_0}$ and ψ_ℓ^ϕ. Furthermore, the right hand side of the PDE in (4.57) is replaced by the source term $-\eta_\varepsilon^\ell(\vec{x})$. The sensitivity field of m_0^c at the location $\vec{x}_\ell$ is computed via:*

$$\boxed{\left.\frac{\partial m_0^c(\boldsymbol{y})}{\partial y_j}\right|_{\vec{x}=\vec{x}_\ell} = \int_{t_j} \left\{ \left(\nabla \psi_\ell^\phi - \psi_\ell^{m_0} \nabla m_0^c\right) \cdot (K \nabla \phi) \right\} d\Omega} \tag{4.67}$$

Remark 4.4. *When monitoring ϕ, we set $\psi_\ell^{m_1} = 0$ and $\psi_\ell^{m_0} = 0$ in the BVP (4.62) for the adjoint state ψ_ℓ^ϕ and the source term $-\eta_\varepsilon^\ell(\vec{x})$ becomes the right hand side of the PDE in (4.62). The sensitivity field of ϕ at the location $\vec{x}_\ell$ is computed via:*

$$\boxed{\left.\frac{\partial \phi(\boldsymbol{y})}{\partial y_j}\right|_{\vec{x}=\vec{x}_\ell} = \int_{t_j} \left\{ \nabla \psi_\ell^\phi \cdot (K \nabla \phi) \right\} d\Omega \qquad (4.68)}$$

Cirpka and Kitanidis [2001] obtain $-\nabla\psi_\ell^\phi$ instead of $\nabla\psi_\ell^\phi$ under the integrals of (4.66)-(4.68) due to the fact that they used $\mathrm{div}(K\nabla\psi_\ell^\phi)$ in (4.62) instead of $\mathrm{div}(-K\nabla\psi_\ell^\phi)$.

In summary, the adjoint equation method for the computation of a discrete sensitivity field

$$\boldsymbol{h}_\ell := \left.\frac{\partial u(\boldsymbol{y})}{\boldsymbol{y}}\right|_{\vec{x}=\vec{x}_\ell} \in \mathbb{R}^N \qquad \ell = 1, ..., M \tag{4.69}$$

requires the solution of ν adjoint problems per measuring point and ν forward problems, where ν is the number of measurement types in the dependency chain ($\nu = 1$ for $u = \phi$, $\nu = 2$ for $u = m_0^c$ and $\nu = 3$ for $u = m_1^c$). In total, this amounts to the solution of $\nu \cdot (M + 1)$ problems.

4.2.4 Efficient computation of cross-covariance matrices

The periodic circulant embedding ($\Omega \subset \Omega'$) described in §2.3 can be re-used to speed up the computation of the large matrix×matrix product $\mathcal{R}_{yy}\mathcal{H}^T$ remarkably [Nowak et al., 2003]. Each of the M rows of the sensitivity matrix $\mathcal{H}$ is filled with a sensitivity field (4.69). In other words,

$$\mathcal{H}^T = [\boldsymbol{h}_1 | \dots | \boldsymbol{h}_M] \,. \tag{4.70}$$

We need to compute

$$\mathcal{R}_{yy}\mathcal{H}^T = [\mathcal{R}_{yy}\boldsymbol{h}_1 | \dots | \mathcal{R}_{yy}\boldsymbol{h}_M] \,. \tag{4.71}$$

The linear mapping

$$\begin{array}{rccc} \mathcal{P}: & \mathbb{R}^N & \longrightarrow & \mathbb{R}^{N'} \\ & \boldsymbol{h} = (h_1, ..., h_N)^T & \mapsto & (h_1, ..., h_N, \underbrace{0, ..., 0}_{P:=N'-N})^T \end{array} \quad , \tag{4.72}$$

which simply adds **zero padding** to the embedded vector $\boldsymbol{h} \in \mathbb{R}^N$, has the matrix representation

$$\mathcal{P} = \begin{bmatrix} \mathbf{Id} \\ \mathbf{0} \end{bmatrix} \quad \text{with} \quad \mathbf{Id} \in \mathbb{R}^{N\times N}, \mathbf{0} \in \mathbb{R}^{P\times N} \tag{4.73}$$

On the other hand, for a vector $\boldsymbol{h}' \in \mathbb{R}^{N'}$, the extraction of its first N components is obtained with $\mathcal{P}^T\boldsymbol{h}'$. Now, we can write for any vector $\boldsymbol{h} \in \{\boldsymbol{h}_1, ..., \boldsymbol{h}_M\}$:

$$\mathcal{R}_{yy} \cdot \boldsymbol{h} = \mathcal{P}^T \cdot \mathcal{S}_{yy} \cdot \mathcal{P} \cdot \boldsymbol{h} \tag{4.74}$$

$$\overset{(A.30)}{=} \mathcal{P}^T \cdot \mathcal{F}_{N'} \,\mathrm{diag}(\boldsymbol{\lambda})\, \mathcal{F}_{N'}^{-1} \cdot \mathcal{P} \cdot \boldsymbol{h} \tag{4.75}$$

The eigenvalues $\boldsymbol{\lambda} = (\lambda_0, ..., \lambda_{N'-1})^T$ of the circulant matrix $\mathcal{S}_{yy} \in \mathbb{R}^{N' \times N'}$ computed in (2.19) in step (3) of Algorithm 3.1 can be retrieved from storage.
A direct multiplication for the left hand side of (4.75) requires $\mathcal{O}(N^2)$ floating point operations. The right hand side requires two FFTs of lengths N' which has complexity $\mathcal{O}(N' \log(N')) = \mathcal{O}(N \log(N))$ since $N' \sim 2^d N$ (see Remarks A.2 and the definition of N' in Algorithm 2.2). Thus, the complexity for the computation of the M cross-covariance fields in (4.71) is reduced from $\mathcal{O}(MN^2)$ to $\mathcal{O}(MN\log(N))$.

4.3 Subroutines of the inversion algorithm

Algorithm 4.1 is invoked in step (12.3) of Algorithm 3.1 and in step (1) of Algorithm 4.2:

Algorithm 4.1 Solving forward problems and take simulated measurements

```
1:  Input: Current estimate of the parameter field $\boldsymbol{y} \in \mathbb{R}^N$
2:                                        ▷ // Make sure that $M = M_\phi + M_{m_0^c} + M_{m_1^c}$ !
3:  (1) Solve the forward BVP (4.1) for $\phi$
4:  for $\ell \in \{1, ..., M_\phi\}$ do
5:      (1.1) Take point measurement $f_\ell = \phi(\vec{x}_\ell)$;
6:  end for
7:
8:  (2) Solve the forward BVP (4.21) for $m_0^c$;        ▷ // only if $u = m_1^c$ or $u = m_0^c$
9:  for $\ell \in \{1, ..., M_{m_0^c}\}$ do
10:     (2.1) $\ell = \ell + M_\phi$;
11:     (2.2) Take point measurement $f_\ell = m_0^c(\vec{x}_\ell)$;
12: end for
13:
14: (3) Solve the forward BVP (4.21) for $m_1^c$;                  ▷ // only if $u = m_1^c$
15: for $\ell \in \{1, ..., M_{m_1^c}\}$ do
16:     (3.1) $\ell = \ell + M_{m_0^c} + M_\phi$;
17:     (3.2) Take point measurement $f_\ell = m_1^c(\vec{x}_\ell)$;
18: end for
19:
20: Output: Simulated measurements $\boldsymbol{f}(\boldsymbol{y}) \in \mathbb{R}^M$
```

Algorithm 4.2 is invoked in step (7) of Algorithm 3.1:

Algorithm 4.2 Sensitivities and cross-covariance matrices

```
1:  Input: Current estimate of the parameter field $\boldsymbol{y} \in \mathbb{R}^N$
2:
3:  (1) Call Algorithm 4.1 with input $\boldsymbol{y}$;        ▷ // Get simulated measurements $\boldsymbol{f}(\boldsymbol{y})$
4:
5:  for $\ell \in \{1, ..., M\}$ do
6:      (3.1) Solve the adjoint BVP (4.54) for $\psi_\ell^{m_1}$;        ▷ // only if $u = m_1^c$
7:      (3.2) Solve the adjoint BVP (4.57) for $\psi_\ell^{m_0}$;        ▷ // only if $u = m_0^c$ or $u = m_0^c$
8:      (3.3) Solve the adjoint BVP (4.62) for $\psi_\ell^{\phi}$;
9:      (4) Compute and store the sensitivity field
                $$\boldsymbol{h}_\ell := \frac{\partial u(\boldsymbol{y})}{\boldsymbol{y}}$$
10:             as in (4.66) if $u = m_1^c$, (4.67) if $u = m_0^c$, or (4.68) if $u = \phi$;
11:     (5) Compute and store the cross-covariance field $\mathcal{R}_{yy} \cdot \boldsymbol{h}_\ell$ as in (4.75);
12: end for
13:
14: Output: $\boldsymbol{f}(\boldsymbol{y}) = (f_1, ..., f_M)^T$ and the columns of $\mathcal{H}^T$ and $\mathcal{R}_{yy}\mathcal{H}^T$.
```

Chapter 5

Numerical Solution of Model and Adjoint Equations

All the boundary value problems presented in Chapter 4 can be classified into two types. We first summarize all occurring source terms and boundary conditions in an overview. Then, we consider two combinations of numerical schemes (FEM/SDFEM vs. CCFV/DG) for the discretization of these BVPs before showing methods to reduce numerical oscillations and methods to solve the occurring linear systems.

5.1 Types of equations

Type I:
The differential equation (4.1) and its adjoint counterpart (4.62) assume the form of the *steady-state diffusion equation*

$$\operatorname{div}\big(- K\nabla\phi\big) = \tilde{w} \quad \text{in} \quad \Omega \tag{5.1}$$

subject to the boundary conditions

$$\begin{array}{rcll} \phi & = & \hat{\phi}_D & \text{on} \quad \Gamma_D,\ \Gamma_D \neq \emptyset \\ \vec{n}\cdot(-K\nabla\phi) & = & 0 & \text{on} \quad \Gamma_N = \partial\Omega\backslash\Gamma_D\ . \end{array} \tag{5.2}$$

The source term and the Dirichlet boundary function are specified as follows:

	equation		source type	$\tilde{w}$	Dirichlet b.c.
(a)	(4.1)	(forward)	(S1) well	$\tilde{w}_{\text{inj}} - \tilde{w}_{\text{ext}}$	inhomogeneous
(b)	Remark 4.4	(adjoint)	(S2) point	$-\eta^{\ell}_{\varepsilon}(\vec{x})$	homog.: $\hat{\phi}_D = 0$
(c)	(4.62)	(adjoint)	(S3) derivative	$\operatorname{div}(-\psi^u K\nabla u)$	homog.: $\hat{\phi}_D = 0$

Table 5.1: Overview of source and boundary terms for the diffusion equation (5.1)

Type II:

The differential equations (4.21), (4.57) and (4.54) are special cases of the *steady-state convection-diffusion-reaction equation*

$$\operatorname{div}(-\mathfrak{D}\nabla u + \tau\vec{q}\,u) + \mu u = \tilde{s} \qquad \text{in} \quad \Omega\,. \tag{5.3}$$

The reaction coefficient $\mu \in \mathbb{R}$ is used here only as a sink term within extraction wells. The factor $\tau \in \{-1, +1\}$ distinguishes between the forward and adjoint equations. Under the assumption (4.13) that the flux is non-diffusive on all boundaries, the Neumann b.c. reads

$$\vec{n} \cdot (-\mathfrak{D}\nabla u) \;=\; 0 \;\; \text{on} \;\; \Gamma\;. \tag{5.4}$$

The Dirichlet b.c. and the source term are given by:

	equation		τ	source type	$\tilde{s}$	μu	Dirichlet b.c.
(a)	(4.21)	(forward)	1	(S4) injection	$\tilde{w}_{\text{inj}}\tilde{u}_{\text{inj}}$	$\tilde{w}_{\text{ext}}u$	$u = \hat{u}_D$ at Γ_-
(b)	(4.54)	(adjoint)	-1	(S2) point	$-\eta_\varepsilon^\ell(\vec{x})$	$\tilde{w}_{\text{inj}}u$	$u = 0$ at Γ_+
(c)	(4.57)	(adjoint)	-1	(S5) functional	θt	$\tilde{w}_{\text{inj}}u$	$u = 0$ at Γ_+
(d)	(4.21)	(forward)	1	(S4)+(S5) combined	$\tilde{w}_{\text{inj}}\tilde{u}_{\text{inj}}$, θt	$\tilde{w}_{\text{ext}}u$	$u = \hat{u}_D$ at Γ_-

Table 5.2: Overview of source, reaction and b.c. terms for the transport equation (5.3)

$\vec{q} = -K\nabla\phi$ is the Darcy velocity (4.5) and $\mathfrak{D}$ is the dispersion tensor given by (4.8).

Type II - non-conservative formulation:

For $\mu' := \mu + \tau \operatorname{div}(\vec{q}\,)$, equation (5.3) is equivalent to

$$\operatorname{div}(-\mathfrak{D}\nabla u) + \tau\vec{q}\,\nabla u + \mu' u = \tilde{s} \qquad \text{in} \quad \Omega\,. \tag{5.5}$$

The Dirichlet b.c. and the source term are given by:

	equation		τ	source type	$\tilde{s}$	$\mu' u$	Dirichlet b.c.
(a)	(4.21)	(forward)	1	(S4) injection	$\tilde{w}_{\text{inj}}\tilde{u}_{\text{inj}}$	$\tilde{w}_{\text{inj}}u$	$u = \hat{u}_D$ at Γ_-
(b)	(4.54)	(adjoint)	-1	(S2) point	$-\eta_\varepsilon^\ell(\vec{x})$	$\tilde{w}_{\text{ext}}u$	$u = 0$ at Γ_+
(c)	(4.57)	(adjoint)	-1	(S5) functional	θt	$\tilde{w}_{\text{ext}}u$	$u = 0$ at Γ_+
(d)	(4.21)	(forward)	1	(S4)+(S5) combined	$\tilde{w}_{\text{inj}}\tilde{u}_{\text{inj}}$, θt	$\tilde{w}_{\text{inj}}u$	$u = \hat{u}_D$ at Γ_-

Table 5.3: Overview of source, reaction and b.c. terms for the transport equation (5.5)

5.2 Discrete solution spaces

As in §2.2, we assume that the domain $\Omega \subset \mathbb{R}^d$, $d \in \{2,3\}$ is a rectangular cuboid. Let $\{\mathcal{T}_{h_\nu}\}_{\nu\in\mathbb{N}}$ be a family of structured or adaptively refined meshes (comprised of axis parallel cuboidal cells) that we get from a successive refinement of an initially structured mesh. Each $\mathcal{T}_{h_\nu}$ forms a partitioning of Ω into n_ν disjoint cells (mesh elements), this means

$$\begin{aligned} \mathcal{T}_{h_\nu} &= \{t_j^\nu\}_{j=0,\dots,n_\nu-1} \\ \overline{\Omega} &= \bigcup_{j=0}^{n_\nu-1} \overline{t_j^\nu}, \quad t_i^\nu \cap t_j^\nu = \emptyset \quad \forall i \neq j. \end{aligned} \tag{5.6}$$

The variable ν indicates the refinement level. As refinement proceeds, the meshsize

$$h_\nu = \max_{t\in\mathcal{T}_\nu^*}\left\{\max_{\vec{x},\vec{y}\in\bar{t}} \|\vec{x}-\vec{y}\|\right\} \tag{5.7}$$

tends to 0. $\mathcal{T}_\nu^*$ is understood to be the subset of $\mathcal{T}_{h_\nu}$ that contains only the finest level cells. To keep notation readable, we write h instead of h_ν, $\mathcal{T}_h$ instead of $\mathcal{T}_{h_\nu}$ and n instead of n_ν when it is clear or irrelevant which refinement level ν we are considering.

The hydraulic conductivity is resolved on the structured mesh $\mathcal{T}_{h_0}$. It is described by a cell-wise constant function as in §2.2.2:

$$K_h(\vec{x}) = K(\vec{x}_t) = \exp(Y(\vec{x}_t)) \qquad \forall \vec{x} \in t,\ \vec{x}_t \text{ is the center the cell } t \in \mathcal{T}_{h_0}\ . \tag{5.8}$$

Inside the wells, the hydraulic conductivity is supposed to be very high:

$$K_h(\vec{x}) = 1.0 \qquad \forall\, \vec{x} \in t, t \in \mathcal{T}_{h_0} \quad \text{with} \quad t \cap W_{\text{inj}} \neq \emptyset \quad \text{or} \quad t \cap W_{\text{ext}} \neq \emptyset\ . \tag{5.9}$$

The hydraulic head distribution and the Darcy velocity (4.5) are computed on the same mesh. The transport equation (5.3), whose convective part is prescribed by the Darcy velocity, may be solved either on the same mesh or on a hierarchy of adaptively refined meshes based on $\mathcal{T}_{h_0}$, i.e. a cell of a subsequently refined mesh $\mathcal{T}_{h_{\nu+1}}$ is always a subset of a cell in $\mathcal{T}_{h_\nu}$.

In our practical application, the estimated hydraulic conductivity fields (3.50) in the inversion algorithm have the smoothness of a Gaussian variogram model. The mesh-sizes are chosen in such a way that they resolve the correlation lengths well. Thus, the flow field on the mesh $\mathcal{T}_{h_0}$ can be regarded as sufficiently accurate and we consider adaptive mesh refinement only for the solution of the transport problem.

Finite elements on cuboids $t \in \mathcal{T}_h$ are based on the polynomial space

$$\mathbb{Q}_k^d = \left\{ p(\vec{x}) = \sum_{\substack{0 \le \alpha_i \le k \\ 1 \le i \le d}} \gamma_{\alpha_1,\dots,\alpha_d} \cdot x_1^{\alpha_1} \cdots x_d^{\alpha_d}, \quad \vec{x} \in t \right\} \tag{5.10}$$

with maximal degree k in each coordinate direction. Discontinuous Galerkin (DG) approximations are based on the **broken polynomial space**

$$W_{h,k} = W_{h,k}(\Omega, \mathcal{T}_h) = \left\{ u \in L^2(\Omega) : u|_t \in \mathbb{Q}_k^d \quad \forall\, t \in \mathcal{T}_h \right\}. \tag{5.11}$$

We restrict ourselves to the case where the maximal polynomial degree k is constant for all cells:

$$\dim(W_{h,k}) = n \cdot \dim(\mathbb{Q}_k^d) = n \cdot (k+1)^d\,. \tag{5.12}$$

The **continuous polynomial space**

$$V_h \;=\; V_h(\Omega, \mathcal{T}_h) \;=\; \left\{ u \in C^0(\overline{\Omega}) : u|_t \in \mathbb{Q}_1^d \quad \forall\, t \in \mathcal{T}_h \right\} \;\subset\; H^1(\Omega) \tag{5.13}$$

is used to describe the standard Galerkin FEM and the streamline diffusion method. For a structured mesh with $n = \prod_{i=1}^{d} n_i$ cells, its dimension is $\prod_{i=1}^{d} (n_i + 1)$.

All cell-wise or face-wise defined integrals of the form (E.3), that will occur in the following numerical schemes, are integrals over products of at least two polynomials of order k. A Gaussian quadrature rule of order $k+1$ guarantees the exact evaluation of polynomials of order $2k+1$.

5.3 FEM / SDFEM

Let $V_h^0 = \{u \in V_h \;:\; u_{|\partial\Omega} = 0\}$ and assume $\hat{\phi}_D$ to be a piecewise linear approximation of the Dirichlet b.c. in (5.2). The standard Galerkin **FEM method** (cf. [Elman et al., 2005]) for solving (5.1)&(5.2) reads:

Find $\phi_h \in V_h^{\mathrm{D}} = \{u \in V_h \;:\; u_{|\Gamma_D} = \hat{\phi}_D\}$ such that

$$\sum_{t \in \mathcal{T}_h} \left(K_h \nabla \phi_h, \nabla v_h \right)_{0,t} = \tilde{w}_h(v_h) \quad \forall\, v_h \in V_h^0\,. \tag{5.14}$$

The discrete source term on the right-hand side is

$$\tilde{w}_h(v_h) = \begin{cases} \sum\limits_{\substack{t\in\mathcal{T}_h \\ t\cap W_{\text{inj}}\neq\emptyset}} \left(\tilde{w}_{\text{inj}}, v_h\right)_{0,t} - \sum\limits_{\substack{t\in\mathcal{T}_h \\ t\cap W_{\text{ext}}\neq\emptyset}} \left(\tilde{w}_{\text{ext}}, v_h\right)_{0,t} & \text{(a)} \\ -\sum\limits_{t\in\mathcal{T}_h} \left(\eta_\varepsilon^\ell(\vec{x}), v_h\right)_{0,t} \approx -\sum\limits_{t\in\mathcal{T}_h} \left(\delta(\vec{x}-\vec{x}_\ell), v_h\right)_{0,t} = -v_h(\vec{x}_\ell) & \text{(b)} \\ \sum\limits_{t\in\mathcal{T}_h} \left(\, \psi_h^u K_h \nabla u_h,\; \nabla v_h \,\right)_{0,t} - \sum\limits_{f\in\mathcal{B}_h\cap\Gamma_N} \left(\, \vec{n}_f,\; \psi_h^u K \nabla u_h \,\right)_{0,t} & \text{(c)} \end{cases} \tag{5.15}$$

where $K_h \in W_{h_0,0}$ as defined in (5.8). ψ_h^u and u_h are the discrete solutions of the adjoint and forward transport equations respectively.

The discrete Darcy velocity can be computed by direct pointwise evaluation of gradients of the polynomial basis on each cell $t \in \mathcal{T}_{h_0}$:

$$\vec{q}_h = -K_h \nabla \phi_h. \tag{5.16}$$

The **SDFEM method** (cf. [Brooks and Hughes, 1982]) for solving (5.5) reads:

$$\boxed{\begin{aligned} &\text{Find } u_h \in V_h^- = \left\{u \in V_h \;:\; u_{|\Gamma_-} = \hat{u}_D\right\} \text{ such that} \\ &\sum_{t\in\mathcal{T}_h} \left\{ \left(\, \mathbf{D}\nabla u_h,\; \nabla v_h \,\right)_{0,t} + \left(\, \vec{q}_h \, \nabla u_h + \mu_h' u_h,\; v_h + \delta_t^{\text{SD}} \cdot \vec{q}_h \, \nabla v_h \,\right)_{0,t} \right\} \\ &\qquad\qquad = \sum_{t\in\mathcal{T}_h} \left\{ \left(\, \tilde{s}_h,\; v_h + \delta_t^{\text{SD}} \cdot \vec{q}_h \, \nabla v_h \,\right)_{0,t} \right\} \quad \forall\, v_h \in V_h^0. \end{aligned}} \tag{5.17}$$

In the adjoint case, $\vec{q}_h$ is replaced by $-\vec{q}_h$, Γ_- and Γ_+ swap positions and $\hat{u}_D = 0$. μ_h' and $\tilde{s}_h$ evaluate the corresponding terms listed in Table 5.3 at quadrature points. Note that the point source is discretized in the same way as in equation (5.15(b)). The matrix $\mathbf{D} \in \mathbb{R}^{d\times d}$ is the discretized version of the dispersion tensor $\mathcal{D}$ in (4.8) which is depending on $\vec{v}_h = \vec{q}_h/\theta$. The **stabilization parameter** determined by

$$\delta_t^{\text{SD}} = \frac{h}{2\|\vec{q}_h\|_2} \cdot \zeta(\mathcal{P}_h^t) \tag{5.18}$$

is used to tune the amount of artificial diffusion depending on the magnitude of the **mesh Péclet number** $\mathcal{P}_h^t$. If $\mathbf{D} = \varepsilon \cdot \mathbf{Id}$ ($\varepsilon > 0$), the general definition of the mesh Péclet number is

$$\mathcal{P}_h^t = \frac{1}{2} \cdot \frac{\|\vec{q}_h\|_2 \cdot h_t}{\varepsilon}. \tag{5.19}$$

For the Scheidegger dispersion tensor (4.9), the effective mesh Péclet number according to [Cirpka and Kitanidis, 2001] is

$$\mathcal{P}_h^t = \frac{1}{2} \cdot \frac{\|\vec{q}_h\|_2 \cdot h_t}{\alpha_\ell \cdot \|\vec{q}_h\|_2 + \theta D_m}. \tag{5.20}$$

There is a large variety of definitions for the function ζ in the literature. The original choice in [Brooks and Hughes, 1982] is

$$\zeta(\mathcal{P}_h^t) = \coth(\mathcal{P}_h^t) - 1/\mathcal{P}_h^t. \tag{5.21}$$

We go for the more efficient approximation (cf. [Elman et al., 2005])

$$\zeta(\mathcal{P}_h^t) = \max\left\{\, 0,\ 1 - 1/\mathcal{P}_h^t \,\right\}. \tag{5.22}$$

Remark 5.1.

- δ_t^{SD} *vanishes in the diffusion-dominated case* ($\mathcal{P}_h^t \leq 1$) *and SDFEM reduces to the standard Galerkin FEM.*
- *The stabilization in (5.17) is based on artificial diffusion added by the multiplication of the residual*

$$\nabla \cdot (-\mathbf{D}\nabla u_h) + \vec{q}_h \nabla u_h + \mu_h' u_h - \tilde{s}_h \tag{5.23}$$

with the term $\delta_t^{SD} \cdot \vec{q}_h\ \nabla v_h$ *. Since we consider only bilinear polynomials* ($\mathbb{Q}_1^d$) *here, the second-order derivative can be omitted.*

5.4 CCFV / DG

5.4.1 Preliminary definitions

The following notation is inspired by the presentation of Ern et al. [2008]. For a given mesh $\mathcal{T}_h$, each cell $t \in \mathcal{T}_h$ has a cell-center $\vec{x}_t$ and a d-dimensional cell-volume $|t|$. Given two neighboring cells t^- and t^+ in $\mathcal{T}_h$, an interior face or interface f is defined as the intersection of their boundaries $\partial t^- \cap \partial t^+$. To be more precise, we write $t_f^- = t^-$ and $t_f^+ = t^+$. The **unit normal vector** $\vec{n}_f$ to f is assumed to be oriented from t_f^- to t_f^+. In the same manner, f is called a boundary face if there exists a $t \in \mathcal{T}_h$ such that $f = \partial t \cap \partial\Omega$ and we write $t_f^- = t$. In this case, $\vec{n}_f$ is chosen to be the unit outer normal to $\partial\Omega$. We denote by $\mathcal{E}_h$ the **set of interior faces** and by $\mathcal{B}_h$ the **set of boundary faces**. For a face $f \in \mathcal{E}_h \cup \mathcal{B}_h$, we denote by $\vec{x}_f$ its midpoint (face center), whereas $\vec{x}_{f-}$ is the center of the cell t_f^-. For $f \in \mathcal{E}_h$, we denote by $\vec{x}_{f+}$ the center of t_f^+. $|f|$ is the $(d-1)$-dimensional volume of the face f.

A function $u \in W_{h,k}$ is in general double valued on internal faces $f \in \mathcal{E}_h$. There, we set

$$u_f^\pm(\vec{x}) = \lim_{\varepsilon \to 0\pm} u(\vec{x} + \varepsilon \vec{n}_f) \qquad \vec{x} \in f. \tag{5.24}$$

The jump across f and the arithmetic mean value on f are given by

$$[\![u]\!]_f = u_f^- - u_f^+ \quad \text{and} \quad \langle u \rangle_f(\vec{x}) = \frac{1}{2}\left(u_f^- + u_f^+\right) \tag{5.25}$$

respectively. Following convention, the definition of these terms is extended to the boundary $\partial\Omega$ by:

$$[\![u]\!]_f(\vec{x}) = \langle u \rangle_f(\vec{x}) = u(\vec{x}) \qquad \forall\, \vec{x} \in f, \quad f \in \mathcal{B}_h. \tag{5.26}$$

We will suppress the letter f in subscripts if there is no ambiguity.

5.4.2 Two-point flux cell-centered finite volume method (CCFV)

Using the function space $W_{h,0}$, the approximation of the boundary value problem (5.1) & (5.2) is defined as follows:

Find $\phi_h \in W_{h,0}$ such that

$$a^{\text{FV}}(\phi_h, v_h) = \ell^{\text{FV}}(v_h) \qquad \forall\, v_h \in W_{h,0}. \tag{5.27}$$

The bilinear form $a^{\text{FV}} : W_{h,0} \times W_{h,0} \longrightarrow \mathbb{R}$ is defined by

$$a^{\text{FV}}(\phi, v) \;=\; \sum_{f \in \mathcal{E}_h} q_h^\phi(\vec{x}_f) \cdot [\![v]\!]_f \cdot |f| \;+\; \sum_{f \in \mathcal{B}_h \cap \Gamma_D} q_h^\phi(\vec{x}_f) \cdot v(\vec{x}_{f-}) \cdot |f| \tag{5.28}$$

where

$$q_h^\phi(\vec{x}_f) := \begin{cases} -K_h^{\text{eff}}(\vec{x}_{f+}, \vec{x}_{f-}) \cdot \dfrac{\phi(\vec{x}_{f+}) - \phi(\vec{x}_{f-})}{\|\vec{x}_{f+} - \vec{x}_{f-}\|_2} & \text{for} \quad f \in \mathcal{E}_h, \\[3ex] -K_h(\vec{x}_{f-}) \cdot \dfrac{\hat{\phi}_D(\vec{x}_f) - \phi(\vec{x}_{f-})}{\|\vec{x}_f - \vec{x}_{f-}\|_2} & \text{for} \quad f \in \mathcal{B}_h \cap \Gamma_D. \end{cases} \tag{5.29}$$

is a two-point finite difference approximation of the normal component $\vec{q} \cdot \vec{n}_f(\vec{x}_f)$ of the Darcy velocity $\vec{q} = -K\nabla\phi$ through the face f. The discrete hydraulic conductivity $K_h \in W_{h_0,0}$ is defined as in (5.8) and

$$K_h^{\text{eff}}(\vec{x}_{f+}, \vec{x}_{f-}) = \frac{2 \cdot K_h(\vec{x}_{f+}) \cdot K_h(\vec{x}_{f-})}{K_h(\vec{x}_{f+}) + K_h(\vec{x}_{f-})} \tag{5.30}$$

is the **harmonic average** of $K_h(\vec{x}_{f+})$ and $K_h(\vec{x}_{f-})$. In the definition (5.29), we make use of the fact that the face f is perpendicular to the line connecting $\vec{x}_{f+}$ and $\vec{x}_{f-}$.

Special care must be taken in the discretization of the source term $\text{div}(-\psi^u K\nabla u)$ from Table 5.1 (c). In this term, u is either the zeroth or first order temporal moment of concentration and ψ^u one of their corresponding adjoint states. While in the FEM / SDFEM combination, the discrete solutions of both equation types (Type I and II from §5.1) are from the same function space V_h, the situation in this section is more complicated. The solution ϕ_h to (5.1) is sought in $W_{h,0}$. However, both u and ψ^u are approximated by functions u_h and ψ_h^u in $W_{h,k} (k > 0)$ (see §5.4.4) or in V_h (see §5.6) respectively. The term $\text{div}(-\psi^u K\nabla u)$ has a similar structure as the differential operator of (5.1) itself. We choose the following approximation:

$$q_h^u(\vec{x}_f) := \begin{cases} -\langle \psi_h^u \cdot \rangle_f \; K_h^{\text{eff}}(\vec{x}_{f+}, \vec{x}_{f-}) \cdot \dfrac{u_h(\vec{x}_{f+}) - u_h(\vec{x}_{f-})}{\|\vec{x}_{f+} - \vec{x}_{f-}\|_2} & \text{for} \quad f \in \mathcal{E}_h, \\[2em] -\psi_h^u(\vec{x}_{f-}) \cdot K_h(\vec{x}_{f-}) \cdot \dfrac{\hat{u}_D(\vec{x}_f) - u_h(\vec{x}_{f-})}{\|\vec{x}_f - \vec{x}_{f-}\|_2} & \text{for} \quad f \in \mathcal{B}_h \cap \Gamma_D. \end{cases} \tag{5.31}$$

Hence the linear functional $\ell^{\text{FV}} : W_{h,0} \longrightarrow \mathbb{R}$ is given by

$$\ell^{\text{FV}}(v) = \begin{cases} \sum\limits_{\substack{t \in \mathcal{T}_h \\ t \cap W_{\text{inj}} \neq \emptyset}} \tilde{w}_{\text{inj}}(\vec{x}_t) \cdot v(\vec{x}_t) \cdot |t| - \sum\limits_{\substack{t \in \mathcal{T}_h \\ t \cap W_{\text{ext}} \neq \emptyset}} \tilde{w}_{\text{ext}}(\vec{x}_t) \cdot v(\vec{x}_t) \cdot |t| & \text{(a)} \\[2em] -\sum\limits_{t \in \mathcal{T}_h} \left(\eta_\varepsilon^\ell(\vec{x}),\; v \right)_{0,t} \approx -\sum\limits_{t \in \mathcal{T}_h} \left(\delta(\vec{x} - \vec{x}_\ell),\; v \right)_{0,t} = -v(\vec{x}_\ell) & \text{(b)} \\[2em] \sum\limits_{f \in \mathcal{E}_h} q_h^u(\vec{x}_f) \cdot [\![v]\!]_f \cdot |f| + \sum\limits_{f \in \mathcal{B}_h \cap \Gamma_D} q_h^u(\vec{x}_f) \cdot v(\vec{x}_{f-}) \cdot |f| & \text{(c)} \end{cases} \tag{5.32}$$

This approximation yields a cell-wise constant solution for which the Dirichlet boundary values $\hat{\phi}_D$ are satisfied weakly.

5.4.3 Flux reconstruction

So far, only the normal flux component $q_h^\phi(\vec{x}_e)$ from (5.29) is available on the midpoints of the faces of a cell $t \in \mathcal{T}_h$. But we need to evaluate the Darcy velocity on internal points of a cell. The simplest $H(\text{div})$-conforming flux reconstruction is achieved using Raviart-Thomas elements of order 0 [Brezzi and Fortin, 1991; Raviart and Thomas, 1977],

$$\begin{aligned}\mathbb{RT}_0(t) = \Big\{ \vec{\tau}(\vec{x}) \ : \tau_i = a_i + b_i(x_i - x_i^*), \quad \vec{x} \in t, \\ 1 \le i \le d, \quad a_i, b_i \in \mathbb{R}, \vec{x}^* \in t \text{ fixed} \Big\},\end{aligned} \tag{5.33}$$

for which the polynomial space is defined by

$$\mathbb{RT}_0(\Omega, \mathcal{T}_h) = \Big\{ \vec{\tau} \in [L^2(\Omega)]^d \ : \ \vec{\tau}_{|t} \in \mathbb{RT}_0(t) \quad \forall t \in \mathcal{T}_h \Big\}. \tag{5.34}$$

The discrete Darcy velocity

$$\vec{q}_h \in \mathbb{RT}_0(\Omega, \mathcal{T}_h) \tag{5.35}$$

can be evaluated component-wise by a linear interpolation between the normal fluxes on opposing face midpoints. For a structured mesh in which all cells are axis-parallel quadrilaterals in 2-D (or axis-parallel cuboids in 3-D) with edge lengths h_i, we can define the $2d$ local degrees of freedom through the normal fluxes given at the midpoints of the $2d$ faces:

$$\begin{aligned} a_1 &= q_h^\phi(x_f^{\text{west}}), & b_1 &= \frac{q_h^\phi(x_f^{\text{east}}) - q_h^\phi(x_f^{\text{west}})}{h_1}, \\ a_2 &= q_h^\phi(x_f^{\text{south}}), & b_2 &= \frac{q_h^\phi(x_f^{\text{north}}) - q_h^\phi(x_f^{\text{south}})}{h_2}, \\ a_3 &= q_h^\phi(x_f^{\text{bottom}}), & b_3 &= \frac{q_h^\phi(x_f^{\text{top}}) - q_h^\phi(x_f^{\text{bottom}})}{h_3}. \end{aligned} \tag{5.36}$$

Interpreting these normal fluxes as face-averaged normal fluxes between two neighboring cells, it can be shown that the reconstructed Darcy velocity $\vec{q}_h \in \mathbb{RT}_0(\Omega, \mathcal{T}_h)$ is indeed in $H(\text{div}, \Omega)$ and satisfies the projection condition:

$$\int_f \left(\vec{q}_h - \vec{q} \right) \cdot \vec{n}_f \, ds = 0 \tag{5.37}$$

Compared to the direct evaluation (5.16), this reconstructed Darcy velocity field is point-wise divergence-free if the groundwater equation is free of any source or sink terms ($\tilde{w}_{\text{inj}} = \tilde{w}_{\text{ext}} = 0$).

5.4.4 Discontinuous Galerkin method (DG)

The symmetric weighted interior penalty (SWIP) method presented in [Ern et al., 2008] is a robust discontinuous Galerkin method accounting for anisotropy and discontinuity in the diffusion tensor of (5.3).

For the discretization of the diffusive term, the authors have introduced a scalar- and double-valued weighting function ω on internal faces. The two values ω^- and ω^+ are constructed based on the double-valued diffusion tensor $\mathbf{D}$ with $\mathbf{D}^-$ and $\mathbf{D}^+$ defined element-wise following (5.24). Using the normal component of $\mathbf{D}^\mp$ across the face, namely $\delta^\mp = \vec{n}_f \cdot \mathbf{D}^\mp \cdot \vec{n}_f$, the weighting factors are defined as

$$\omega^- = \frac{\delta^+}{\delta^- + \delta^+} \qquad \text{and} \qquad \omega^+ = \frac{\delta^-}{\delta^- + \delta^+}. \tag{5.38}$$

Both factors are non-negative and add up to unity $\omega^- + \omega^+ = 1$.
For $v \in W_{h,k}$, the weighted average of the diffusive flux is defined as

$$\left\langle \mathbf{D}\nabla v \right\rangle^\omega = \omega^-(\mathbf{D}\nabla v)^- + \omega^+(\mathbf{D}\nabla v)^+. \tag{5.39}$$

On the boundary face $f \in \mathcal{B}_h$, we set $\omega = 1$ and $\left\langle \mathbf{D}\nabla v \right\rangle^\omega = \mathbf{D}\nabla v$.

For the convective term, we choose an **upwind flux** formulation that is equivalent to the presentations in [Georgoulis et al., 2009], in §4.2 of [Rivière, 2008] or in §4.6.2 of [Pietro and Ern, 2012]. For an internal face $f \in \mathcal{E}_h$ lying between two neighboring cells t_f^- and t_f^+, recall that the unit normal vector $\vec{n}_f$ is assumed to be oriented from t_f^- to t_f^+. For a boundary face $f \in \mathcal{B}_h$, $\vec{n}_f$ is the unit outer normal. By u^{upwind} we denote the upwind value of a function $u \in W_{h,k}$. For $\vec{x} \in f, f \subset \partial t \backslash \Gamma_-$, it is defined by

$$u^{\text{upwind}}(\vec{x}) = \begin{cases} u^+(\vec{x}) & \text{if} \quad \vec{q}_h(\vec{x}) \cdot \vec{n}_f < 0 \\ u^-(\vec{x}) & \text{if} \quad \vec{q}_h(\vec{x}) \cdot \vec{n}_f \geq 0 \end{cases}$$

The higher order DG approximation[1] of the boundary value problem (5.3)&(5.4) reads[2]:

Find $u_h \in W_{h,k}$ such that

$$a^{\text{DG}}(u_h, v_h) = \ell^{\text{DG}}(v_h) \quad \forall\, v_h \in W_{h,k}. \tag{5.40}$$

[1] DG(k) indicates that the polynomial basis is from $\mathbb{Q}_k^d$.

[2] The colors in the following terms are red for the convection terms, blue for the diffusion terms, green for the reaction term and grey for the source term.

The bilinear form is defined by

$$
\begin{aligned}
a^{\mathrm{DG}}(u,v) \; = & \sum_{t\in\mathcal{T}_h} \Big\{ (\mathbf{D}\nabla u, \nabla v)_{0,t} - (u, \vec{q}_h \cdot \nabla v)_{0,t} + (\mu_h u, v)_{0,t} \Big\} \quad \textit{(cell terms)} \\
& + \sum_{f\in\mathcal{E}_h} \Big\{ \big(\, \gamma[\![u]\!],\, [\![v]\!] \,\big)_{0,f} + \big(\, |\vec{n}_f \cdot \vec{q}_h|\, u^{\mathrm{upwind}},\, [\![v]\!] \,\big)_{0,f} \quad \textit{(interior face fluxes)} \\
& \qquad - \big(\, \langle \vec{n}_f \cdot \mathbf{D}\nabla u \rangle^{\omega},\, [\![v]\!] \,\big)_{0,f} - \big(\, \langle \vec{n}_f \cdot \mathbf{D}\nabla v \rangle^{\omega},\, [\![u]\!] \,\big)_{0,f} \Big\} \\
& + \sum_{f\in\mathcal{E}_h\cap\Gamma_+} \big(\, \vec{n}_f \cdot \vec{q}_h\, u,\, v \,\big)_{0,f} \quad \textit{(convective outflow)} \\
& + \sum_{f\in\mathcal{B}_h\cap\Gamma_-} \Big\{ \big(\, \gamma(u-\hat{u}_D),\, v \,\big)_{0,f} - \big(\, u-\hat{u}_D,\, \vec{n}_f \cdot \mathbf{D}\nabla v \,\big)_{0,f} \Big\} \quad \textit{(Dirichlet B.C.)}
\end{aligned}
\tag{5.41}
$$

The linear functional is given by

$$
\ell^{\mathrm{DG}}(v) \; = \sum_{t\in\mathcal{T}_h} (\tilde{s}_h, v)_{0,t} - \sum_{f\in\mathcal{B}_h\cap\Gamma_-} \big(\, \vec{n}_f \cdot \vec{q}_h\, \hat{u}_D,\, v \,\big)_{0,f} \quad \textit{(source term, Dirichlet B.C.)}
\tag{5.42}
$$

In the adjoint case, $\vec{q}_h$ is replaced by $-\vec{q}_h$, Γ_- and Γ_+ swap positions and $\hat{u}_D = 0$. μ_h and $\tilde{s}_h$ evaluate the corresponding terms listed in Table 5.2 at quadrature points. The parameter γ penalizing discontinuity in the solution is given in [Bastian, 2011] by

$$
\gamma = C_\gamma \cdot \frac{D_{\mathrm{eff}} \cdot k(k+d-1)}{h_f} \qquad \forall\, f \in \mathcal{E}_h \cup (\mathcal{B}_h \cap \Gamma_-)
\tag{5.43}
$$

where $C_\gamma > 0$ is a constant to be chosen sufficiently large ($C_\gamma = 10$ is usually enough). With this definition of γ the user-chosen constant does not play a big role anymore in the convection-dominated case.
Remember that for the discrete problem (5.40), $\mathcal{T}_h$ can be a non-conformingly refined cuboidal mesh. For a face lying between two possibly non-matching elements, we set $h_f = \min\{h_f^-, h_f^+\}$ where $h_f^{\mp} = \mathrm{diam}(f \cap \partial t_{\mp})$ are face diameters. On internal faces, the effective diffusivity is defined by the harmonic average

$$
D_{\mathrm{eff}} = \frac{2\delta^-\delta^+}{\delta^- + \delta^+}
\tag{5.44}
$$

of the normal component of the diffusion tensor across the face. On boundary faces, we set directly

$$
D_{\mathrm{eff}} = \vec{n}_f \cdot \mathbf{D} \cdot \vec{n}_f .
\tag{5.45}
$$

5.5 Adaptive mesh refinement

We are interested in reducing numerical oscillations globally. Adaptive mesh refinement is a particularly promising way of achieving this goal.

Residual-based a-posteriori error estimators offer the advantage of small evaluation cost because they are based on local residual terms. We consider two existing h-adaptive versions for the above mentioned discretization schemes of the transport problem. Let u_h be the SDFEM solution (5.17) or the DG solution (5.40) respectively. The residual-based error estimator described by Verfürth [1998, 2005], developed for the finite element and the SDFEM discretization of the steady-state convection-diffusion equation, is based on the following local error indicator: For each element $t \in \mathcal{T}_h$, the local error indicator η_t^2 is given by the sum of two terms,

$$\eta_t^2 = \eta_{R_t}^2 + \eta_{R_f}^2, \tag{5.46}$$

an element residual term[1]

$$\eta_{R_t}^2 = \frac{h_t^2}{\varepsilon} \left\| s_h + \varepsilon \Delta u_h - \vec{q}_h \cdot \nabla u_h - \mu_h' u_h \right\|_{L^2(t)}^2 \tag{5.47}$$

and a face residual term

$$\eta_{R_f}^2 = \frac{1}{2} \sum_{f \in \partial t \setminus \Gamma} \frac{h_f}{\varepsilon} \left\| \, [\![\vec{n} \cdot (\varepsilon \nabla u_h)]\!] \, \right\|_{L^2(f)}^2.$$

To these two terms, Schötzau and Zhu [2009] added a third term measuring jumps of the solution on internal or inflow boundary faces

$$\begin{aligned} \eta_{J_f}^2 = & \frac{1}{2} \sum_{f \in \partial t \setminus \Gamma} \left(\frac{\gamma \varepsilon}{h_f} + \frac{h_f}{\varepsilon} \right) \left\| [\![u_h]\!] \right\|_{L^2(f)}^2 \\ & + \sum_{f \in \partial t \cap \Gamma_-} \left(\frac{\gamma \varepsilon}{h_f} + \frac{h_f}{\varepsilon} \right) \left\| (u_h - \hat{u}_D) \right\|_{L^2(f)}^2 \end{aligned} \tag{5.48}$$

to construct a residual-based error estimator for the interior penalty DG discretization scheme. Hence, the local error indicator η_t^2 for each element $t \in \mathcal{T}_h$ is given by the sum of three terms,

$$\eta_t^2 = \eta_{R_t}^2 + \eta_{R_f}^2 + \eta_{J_f}^2. \tag{5.49}$$

In our concrete application, we choose $\varepsilon = \min_{1 \le i \le d} \{\mathbf{D}_{ii}\}$ to avoid underestimation.

[1] Note that Δu_h can be omitted for the polynomial degree $k = 1$.

The a-posteriori error estimator (in both cases) is defined by

$$\eta = \left(\sum_{t \in \mathcal{T}_h} \eta_t^2 \right)^{1/2} . \tag{5.50}$$

Schötzau and Zhu have shown that their error estimator is robust in convection-dominated regimes, effective in locating characteristic and boundary layers and that the error in the energy norm converges with optimal order as soon as refinement reaches a state when the local mesh Péclet number is of order 1. An extension to hp-adaptivity can be found in [Zhu and Schötzau, 2011]. The performance of Verfürth's error estimator is assessed in a comparative study by John [2000].

Alternative error estimators for DG discretization schemes of the convection-diffusion equation can be found in [Ern et al., 2010] and [Georgoulis et al., 2009].

A robust error estimator is one aspect of adaptivity. Another important aspect is a marking strategy that achieves an equitable distribution of error contributions. An error-fraction based refinement strategy has the following characteristics: Given a fixed refinement fraction $p_r[\%]$ and the list of all cells sorted by the magnitude of the local error indicators

$$\eta_{t_{j1}}^2 \leq \eta_{t_{j2}}^2 \leq \ldots \leq \eta_{t_{jn}}^2 ,$$

the goal is to mark the cells with the largest local errors for refinement such that their contribution to the total error is $p_r[\%]$. To be more precise, we need to find the largest $\eta^\star$ such that

$$\sum_{t \in \mathcal{T}_h : \eta_t \geq \eta^\star} \eta_t^2 \geq \frac{p_r}{100} \cdot \eta^2 \tag{5.51}$$

using the bisection method and mark the top contributors $t \in \mathcal{T}_h$ with $\eta_t \geq \eta^\star$ for refinement. This strategy is readily parallelizable: the sum on the left hand side of (5.51) and the total error η^2 get their contributions from all processes. The very same strategy can be used to mark the mesh cells with the lowest error contribution for coarsening, given a fixed coarsening fraction $p_c[\%]$: Find the smallest $\rho^\star$ such that

$$\sum_{t \in \mathcal{T}_h : \eta_t \leq \rho^\star} \eta_t^2 \leq \frac{p_c}{100} \cdot \eta^2 \tag{5.52}$$

and mark all cells $t \in \mathcal{T}_h$ for which $\eta_t \leq \rho^\star$ for coarsening. Note that the choice of the fixed parameters p_r and p_c are dependent on the error distribution and have a strong influence on the efficiency of the scheme.

For higher order polynomials, the second order derivative in the element residual term (5.47) would be required. In the small 2-D test problems, this is neglected. In the solute transport simulations, we apply adaptivity only to the DG(1) discretization. For a comparative study based on small 2-D test problems, a sequential version of the adaptive SDFEM code is sufficient.

Remember that in the practical application, $\mathcal{T}_0$ is the mesh on which

- the conductivity field K is resolved (5.8),
- the flow equation (4.1) is solved as in §5.4.2 and
- the Darcy flux (4.5) is evaluated as in §5.4.3.

The mesh $\mathcal{T}_0$ should be sufficiently fine such that the main features of the solution are visible and a partitioning of the mesh among all available processes is possible. The **stopping criterion** for refinement is reached as soon as either of the following conditions is fulfilled:

(i) The estimated error is below some prescribed tolerance: $\eta \leq TOL$.

(ii) Provided that the range of the true solution u is given by $[0, \hat{u}]$, we may choose a tolerance of $p_{\text{osc}} = 5\%$ for the maximal under- and overshoots by

$$\max \left\{ \frac{|u_{\min}|}{\hat{u}}, \frac{u_{\max} - \hat{u}}{\hat{u}} \right\} < p_{\text{osc}}. \tag{5.53}$$

(iii) The mesh refinement level L or the total number of unknowns has exceeded a certain limit.

The h**-adaptive mesh refinement algorithm** for the solution of the convection-diffusion equation can be formulated as follows.

Algorithm 5.1 h-adaptive refinement

```
Input: Appropriate values for $p_r$ and $p_c$.
(1) Start with mesh level $L = 0$.
(2) Compute the solution $u_{h_0}$ of (5.40) on $\mathcal{T}_{h_0}$.
(3) Compute the error estimator $\eta$ as in (5.50) for $u_0$.
while $\eta > TOL$ do
    (4) Apply the marking strategy (5.51).
    (5) Refine the mesh and set $L = L + 1$.
    if $L > L_{\max}$ then
        break;                ▷ // maximal number of refinement steps exceeded
    end if
    (2') Compute the solution $u_{h_L}$ of (5.40) on $\mathcal{T}_{h_L}$.
    if (5.53) holds then
        break;                ▷ // overshoots and undershoots are small enough
    end if
    (3') Compute the error estimator $\eta$ as in (5.50) for $u_{h_L}$.
end while
Output: $u_{h_L}$
```

In each refinement step, the linear system for (5.40) can be solved independently of solutions from the previous refinement step since the problem is linear and stationary.

5.6 Diffusive L^2-projection

In this section, we present another method to reduce numerical oscillations. Due to its simplicity we consider this a post-processing step for the DG solution. Given the DG solution $u_{\text{DG}} \in W_{h,k}$ on the coarse level $h = h_0$, our goal is to find an approximation of u_{DG} in the space V_h of continuous Galerkin finite elements that preserves the profile of the DG solution, but with a significant reduction of spurious oscillations. The L^2-projection is a good candidate. It is well-known to give a good on average approximation of a function and it does not require the approximated function to be continuous. Furthermore, an extra term imitating a small amount of diffusive flux can be added. This way, the L^2-projection can be interpreted as the solution of a diffusion-reaction equation without boundary constraints. This leads to the following variational problem:

Find $u_h \in V_h$ such that

$$\big(\varepsilon_h \nabla u_h, \nabla v_h\big)_{0,\Omega} + \big(u_h, v_h\big)_{0,\Omega} = \big(u_{\text{DG}}, v_h\big)_{0,\Omega} \qquad \forall\, v_h \in V_h. \tag{5.54}$$

Hereby, we choose the extra diffusion $\varepsilon_h = \frac{1}{2}h^2$ in such a way that the diffusivity of characteristic layers are in the order of magnitude of the meshsize $\sim O(\sqrt{\varepsilon_h}) = O(h)$.

Remark 5.2. *Taking a point measurement of a DG solution on intersectional entities between two or more neighboring cells requires the averaging over as many different values. Taking a point measurement of the L^2-projected solution $u_h \in V_h$ simply means a direct evaluation of u_h. This is an important aspect for the subroutine 4.1 of the inversion scheme.*

5.7 Efficient solution of the arising linear systems

The parallel overlapping linear solvers available in `DUNE` comprise

- CG (conjugate gradient method),
- GMRES(k) (restarted generalized minimal residual method, where k is the number vectors in an orthogonal sequence that need to be stored temporarily), and
- BiCGSTAB (biconjugate gradient stabilized method).

Each of them can be preconditioned with

- SSOR (symmetric successive over-relaxation method),
- ILU(k) (incomplete LU factorization, where the sparsity pattern of L and U in $A = LU$ are chosen to be the same as for A^{k+1}),
- AMG (algebraic multi-grid method) with SSOR pre-/postsmoothing.

With multigrid methods as preconditioners one can achieve convergence rates which are independent of the resolution of the discretization. The classical geometric multigrid methods rely on the uniform coarsening of the underlying grid. They do not make use of information contained in the conductivity field K. However, modelling flow in a heterogeneous porous medium, K may be highly variable or even have large jumps. Algebraic multigrid methods are able to detect such jumps and adapt their coarsening scheme such that the solver remains robust. Iterative methods using other preconditioners are more likely to run into problems since the condition of the stiffness matrix does not only depend on the resolution of the discretization but also on the ratio of the conductivity jumps. The coarsening strategy for the parallel algebraic multigrid method described by Blatt [2010] works with aggregation. It uses a strength of connection measure that is based on a heuristic greedy aggregation algorithm. The AMG preconditioner is designed for the solution of problems of the type (4.1) with a highly discontinuous coefficient K.

5.7.1 Flow equation and diffusive L^2-projection

The linear systems arising from the discrete elliptic problems (5.14) or (5.27) and (5.54) are all of the size $O(n^2)$, symmetric positive definite and can be solved very efficiently using the combination CG with AMG.

5.7.2 Transport equation

By contrast, the stiffness matrix of the discrete transport equation is non-symmetric. BiCGSTAB or alternatively GMRES are used in our numerical tests. For the SDFEM discretization (5.17), the matrix size is also $O(n^2)$. In the more diffusive case of heat transport, parallel AMG may be used as a preconditioner. In the convection-dominated case, the SSOR or ILU(0) preconditioners are used.

As mentioned in the introduction, for the discretization of a first order hyperbolic problem using an upwind scheme, the order in which the unknowns are indexed, plays an important role for the performance and stability of an iterative solver. The main purpose of ordering unknowns in flow direction can already be found in [Reed and Hill, 1973]. The downwind numbering algorithms described in the works of Bey and Wittum

[1997] and Hackbusch and Probst [1997] handle arbitrary velocity fields. Steady-state groundwater flow (with a scalar conductivity field) is a potential flow and therefore always cycle-free. Since the velocity field is induced by the hydraulic head, the latter can be used directly as the sorting key for the unknowns.

For the DG discretization, it is advisable to collect the unknowns of the solution vector $\vec{u}_h$ block-wise where each vector block $\vec{u}^{(t)}$ holds the unknowns of $\{u_1^{(t)}, \dots, u_{n_{\text{local}}}^{(t)}\}$ of a single mesh cell t with n_{local} denoting the dimension of the local polynomial space. The stiffness matrix $\mathbf{A}_h$ becomes a block matrix with constant block-size $n_{\text{local}} \times n_{\text{local}}$. The arising linear system

$$\mathbf{A}_h \, \vec{u}_h = \vec{b}_h \tag{5.55}$$

is of the size $O(n^2 \cdot n_{\text{local}}^2)$ and can be solved efficiently using a block version of BiCGSTAB or GMRES combined with SSOR or ILU(0) preconditioning, after a renumbering of mesh cells: In the hyperbolic limit the bilinear form of the DG discretization is reduced to the terms listed in the first two lines of (5.41). If the mesh cells are sorted according to the hydraulic head distribution ϕ the stiffness matrix $\mathbf{A}_h$ obtains the shape of a block-triangular matrix. In this case, the symmetric block Gauss-Seidel method for (5.55) becomes a direct solver because it converges after one step.

If the groundwater flow and the solute transport equations are solved on the same mesh, this procedure is straightforward. Otherwise, if adaptive refinement is applied only to the solution of the transport problem, as mentioned in subsection 5.2, the hydraulic head ϕ must be reconstructed on the locally refined sub-cells. Given the discrete Darcy velocity $\vec{q}_h$ in the form (5.35), the hydraulic head can be locally reconstructed as a quadratic function

$$\tilde{\phi}_{|t} = \sum_{j=1}^{d} \left(a_j x_j^2 + b_j x_j \right) + c_0 \tag{5.56}$$

satisfying

$$\tilde{\phi}_{|t} = \phi_{|t} \tag{5.57}$$

on the cell center and the discrete form of Darcy's law

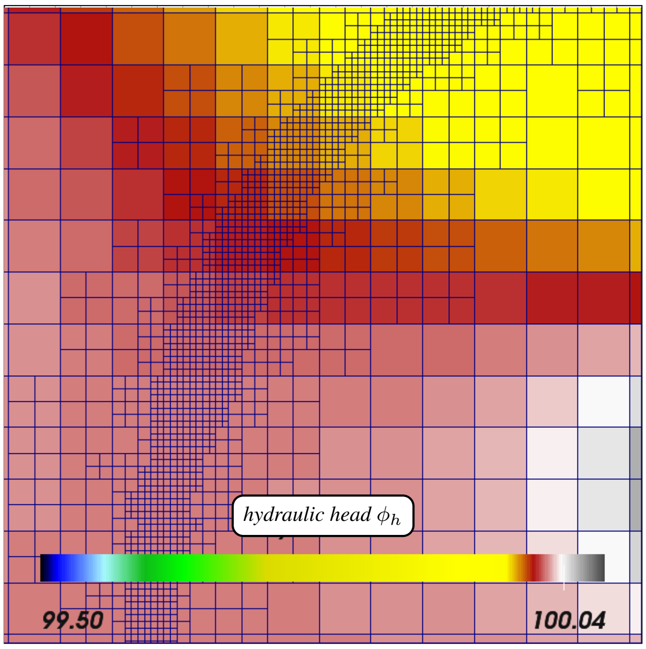
$$-K \nabla \tilde{\phi}_{|t} = \vec{q}_h \tag{5.58}$$

on the $2d$ face centers of a coarse mesh cell $t \in \mathcal{T}_0$. This yields $2d + 1$ equations for the $2d + 1$ coefficients of $\tilde{\phi}_{|t}$. The locally refined sub-cells can then be sorted according to $\tilde{\phi}_{|t}$ evaluated at the centers of the sub-cells (Figure 5.1).

Due to the large problem size in 3-D, parallelization is necessary for efficiency. After each refinement step, the parallel partition-blocks[1] of the coarse mesh $\mathcal{T}_0$ (and with it all locally refined sub-cells) may be altered to achieve a similar amount of refined sub-cells on every processor partition (**dynamic load-balancing**). To ensure that the renumbering procedure still works after each repartitioning step, the two quantities ϕ and K have to

[1] In general, the shape of these blocks is not cuboidal.

be made available on each processor partition of $\mathcal{T}_0$ for a new reconstruction of $\vec{q}_h$ and $\tilde{\phi}$.



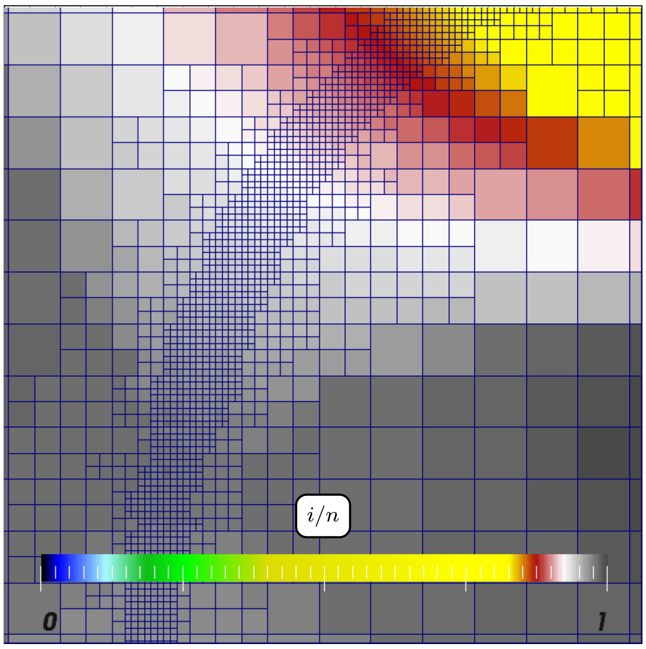


Figure 5.1: Downwind numbering of locally refined mesh cells: The hydraulic head ϕ is constant on each coarse mesh cell (upper plot). For the renumbering of mesh cells according to the hydraulic head, it has to be resolved on the locally refined sub-cells. The cell index i in the lower plot is sorted according to $\tilde{\phi}$. n is the total number of all cells of the locally refined mesh.

Chapter 6

Implementation

This chapter gives an overview of the external libraries used and a short insight into some implementation aspects (explained for the 3-D case).

6.1 The DUNE software framework

DUNE makes extensive use of templates to implement static polymorphism, which has significant advantages in terms of run-time performance and type safety, when compared to dynamic polymorphism [Vandevoorde and Josuttis, 2002]. The set of core modules (**version 2.3.1**) is itemized here with an incomplete list of features:

- `dune-common`: base class templates, exception handler, MPI helper class, dense matrix and vector classes
- `dune-geometry`: generic reference elements and quadrature rules
- `dune-grid`: parallel structured grid `YASP`, interfaces to the unstructured grids `UG` [Bastian et al., 1997] and `DUNE-ALUGrid` [Dedner et al., 2014], VTK output methods
- `dune-istl`: generic sparse matrix/vector classes, parallel linear solvers and preconditioners, parallel AMG, interface to the direct sparse solver `SuperLU` [Li, 2005]
- `dune-localfunctions`: shape functions for conforming and non-conforming finite elements

On top of these modules, the finite element discretization module `dune-pdelab` (**version 2.0.0**)[1] serves as a library tailored for the implementation of grid-based numerical schemes and solvers for linear and non-linear systems of PDEs, stationary as well as

[1]requires the template library `dune-typetree` for statically typed trees of objects

time-dependent. The linear algebra backend facilitates exchanging linear solvers available in `dune-istl` with minimal effort. An implementation of the Newton method enables the solution of non-linear systems of equations. A generic implementation of explicit and implicit one-step-methods allows for the method-of-lines solution of time-dependent systems of PDEs (first-order in time) using different Butcher schemes. The modular design of `DUNE` ensures a great flexibility that allows for the development of completely new modules solving application problems. A long list of applications has been realized using `dune-pdelab`[1]. `dune-gesis` is a further contribution.

6.2 The `DUNE` module `dune-gesis`

6.2.1 Directory structure

All source files containing the definition of data structures and classes are organized in the sub-directories of `dune/gesis/`. Their contents can be summarized as follows:

```
/
├─bin/
├─dune/
│  └─gesis/
│     ├─BVPs/
│     │  ├─adaptive/
│     │  ├─DG/
│     │  ├─FEM/
│     │  ├─obs/
│     │  ├─projectors/
│     │  ├─sourceterms/
│     │  └─wells/
│     ├─common/
│     │  └─io/
│     ├─driver/
│     ├─QLGA/
│     └─yfield/
├─src/
└─tools/
```

`BVPs/`: parameter interface classes with which the coefficients of the PDEs, the types and values of boundary conditions and the diverse source-terms listed in §5.1 are specified;

`adaptive/`: mesh refinement strategy and error estimator;

`DG/`: reordering of grid cells, the equation solvers and local operators for the CCFV / DG approach;

`FEM/`: equation solvers and local operators for the FEM / SDFEM approach;

`obs/`: class handling measurements (taking, storing, redistributing, reading existing);

`projectors/`: diffusive L^2-projection;

`wells/`: assigning grid cells to well locations;

`common/`: general purpose classes and functions;

`io/`: everything related to file I/O (definition of file locations, input file parser, HDF5 routines, VTK output);

`driver/`: driver function initializing the computational grid, selecting the finite element basis and setting up the grid function spaces required for the solution of the

[1]http://www.dune-project.org/pdelab/

forward problems;

yfield/: FFT-based random field generator implementing Algorithm 2.2

QLGA/: computation of objective function, sensitivities, cross-covariance matrices (FFT-based) and the estimated variance; inversion kernel implementing Algorithm 3.1;

The directory src/ contains the 2-D and 3-D versions of the main applications. All available versions are listed in the file Makefile.am.

The directory bin/ is reserved for the build results and supposed to be the working directory for the user. Two examples of user-specified input files containing data for a complete inversion with synthetic fields are given in InputFile2d.xml and InputFile3d.xml.

The directory tools/ contains Python scripts for the visualization of HDF5 files and for the creation of histogram plots.

6.2.2 External libraries

Boost Property Tree

XML[1] (*eXtensible Markup Language*) is a general purpose and easy to learn textual data description language. Every XML document forms a tree that starts at the root node. This node is the parent of other nodes, called children nodes, which themselves may be either a leaf node (which contains only textual or binary data) or the parent node of a sub-tree. Every node may have a set of attributes filled with textual data. The definition of the tree structure including the node and attribute names gives the document the required semantics. Any kind of human-readable information can be represented using an XML document. XML is a widely accepted standard for passing information. The C++ library Boost.PropertyTree[2] provides an XML parser and a data structure that stores an arbitrarily deeply nested tree of values. After parsing, the tree hierarchy with all its nodes and attributes is mapped to this data structure.

The measurement data usually occurs as a long list where each line consists of column- or comma-separated data, holding e.g. the position x, y, z, the value of some measured quantity, its absolute and relative errors. For each set of measurements, this information can be stored in a separate text file. In XML however, this text can be stored as one long string or a CDATA section (interpreted by the parser as character data, not as markup) within a leaf node. Both options are available. dune-gesis reads and parses the input XML-file as one single string buffer on the root process. When run in parallel, file I/O happens only on the root process and this buffer is broadcasted to all other processes before being parsed by all processes.

[1] http://www.w3schools.com/xml/xml_whatis.asp, checked 11/09/2014
[2] http://www.boost.org

HDF5

Working with high resolutions in 3-D, the total size of the sensitivity matrix and the cross-covariance matrix can easily exceed the available RAM. The efficient storage and retrieval of these intermediate results play an important role in the inversion scheme. HDF5[1] (Hierarchical Data Format) is a database format designed for the efficient management of high volume and complex datasets prevalent in scientific computing. The data is structured in an arbitrary tree starting at the root group. A group is a container structure that can inhabit other groups or datasets. A dataset is a multi-dimensional array of a homogeneous datatype. It is comparable to the file-system made of directories (groups) and files (datasets). Data is stored in compressed binary format.

The HDF5 library supports MPI-based partial I/O on different processes using the concept of hyperslab selection. This feature is predestined for data assigned to a structured mesh. In our application, a HDF5 dataset may be the log conductivity field, a sensitivity field or the finite element solution of the transport equation. In all cases, it is not necessary, but it turns out to be very practical to preserve the 3-D structure of each of these fields. A hyperslab is a portion of the whole dataset. When working with P processes in parallel, the computational grid may be partitioned into P hyperslabs (3-D blocks) which may be overlapping or non-overlapping.

Special care must be taken here for reasons of performance: 1.) It is essential that InfiniBand is activated for file access to the hard-disc from all computing nodes. 2.) When working with overlapping hyperslabs and a high number of cores, competing processes may hinder each other during data access on the overlapping regions.

The class `HDF5Tools` in `dune-gesis` provides grid-based or direct sequential or parallel methods for reading from or writing to a HDF5 file. A 3-D array may be written to a HDF5 file preserving its structure or as an 1-D array. If written in structure-preserving mode, the HDF5 files can be read on a differently partitioned parallel grid. This is the basis for the parallel-in-parallel approach investigated in the article [Schwede et al., 2012], in which the parallel computation of the sensitivities introduces an additional layer of concurrency, reducing the overhead of inter-process communication required for domain decomposition in the parallel solution of the adjoint problems. Another advantage of the structure preserving mode is that all intermediate results can be reused in a succeeding computation on a different machine with a different number of cores. And last but not least, a structure preserving HDF5 file can be directly visualized.

FFTW

FFTW[2] (Fastest Fourier Transform in the West) is known as the fastest free software library (written in C) that computes the DFT of real- or complex-valued arrays of arbi-

[1] `http://www.hdfgroup.org/why_hdf`, checked 11/09/2014
[2] `www.fftw.org`

trary length n in $O(n \log n)$ time [Frigo and Johnson, 2005]. MPI distributed memory transforms are supported since the version 3.3 (from 2011). In all the simulations presented in this work, this version linked with the flag `-lfftw3_mpi` was applied. `dune-gesis` works also with the latest version 3.3.4 (2014). FFTW uses a 1-D block distribution of data, distributed along the first dimension[1]. In the implementation of the spectral method (4.75) for the computation of cross-covariance matrices, a redistribution of the sensitivity field data was established using a structure preserving HDF5 storage:

1. The sensitivity field $\boldsymbol{h} \in \mathbb{R}^N$ is computed on the `YASP` grid, underlying a 2-D block MPI distribution of data (see §6.2.3). The result is stored as a 3-D array with $N = n_1 \cdot n_2 \cdot n_3$ in structure-preserving mode to one HDF5 file.

2. The size of the extended field is known and given by $N' = n'_1 \cdot n'_2 \cdot n'_3$ (Algorithm 2.2). Knowing the number of processes in use, the FFTW routine

   ```
   fftw_mpi_local_size_3d
   ```

   provides the information what portion of the extended array is occupied by the current process.

3. The vector $\boldsymbol{h} \in \mathbb{R}^N$, now available in the 1-D block distribution required for FFTW, has to be embedded into a larger array $\boldsymbol{h}' \in \mathbb{R}^{N'}$ using zero-padding before the parallel FFT applies.

4. From the result, the first N components are extracted and stored in structure-preserving mode to a new HDF5 file.

6.2.3 The main performance optimization steps

Domain decomposition

In 3-D groundwater flow simulations, it is often necessary to use a higher resolution in the vertical direction. Therefore, the computational grid cells will become stretched in the x, y-directions and shortened in the z-direction. The number of parallel partition blocks per dimension can be adjusted in the `YASP` grid. We apply a 2-D partitioning only in the x, y-directions, avoiding a deterioration of the linear solver convergence.

[1] http://www.fftw.org/doc/MPI-Data-Distribution.html, checked 11/09/2014

Caching the stiffness matrix

A large fraction of the solution time of a BVP is spent on assembling the stiffness matrix[1] (including the setup of the sparsity pattern). In the computation of the many adjoint states for one measurement type, the stiffness matrix for each of the BVPs (4.54), (4.57) and (4.62) can be "recycled" for different measuring points $\vec{x}_\ell$, because only the right hand side of the PDE varies with each $\vec{x}_\ell$. The forward equation solver classes in `dune-gesis` assemble the stiffness matrix only once and store it as a class member. In the case where AMG is used as the preconditioner, the re-use of the coarsening hierarchy provided by the AMG solver backend is activated.

Renumbering of mesh cells

The new template class `ReorderedGridView` is derived from the base type

```
GRID::LeafGridView
```

where `GRID` can be any of the grid managers `YASP`, `UG` or `ALUGrid`. The grid-view provides ordered access to the grid entities (e.g. cells and faces) via the `indexSet()` method which returns the index-set of the grid. This method is overridden to return the new template class `ReorderedIndexSet` that is derived from the class

```
Dune::IndexSet <GRID, OrigIndexSet, OrigIndexType>
```

(using CRTP[2]) where `OrigIndexSet` is the original (usually lexicographically ordered) index-set and `OrigIndexType` is usually an integer type.
`ReorderedIndexSet` encapsulates the generation of an STL vector container of cell indices, sorted after the cell values of the hydraulic head ϕ (which is passed to the new classes as another template argument).

Caching the Darcy velocity

Once computed for one flow field, the discrete Darcy velocity $\vec{q}_h$ (5.35) is required for the solution of the transport equations for m_0^c and m_1^c, or for all their adjoint states and discrete sensitivities. Caching the values of $\vec{q}_h$ saves a considerable amount of computing time. In `dune-gesis`, a cache handler class evaluates $\vec{q}_h$ using (5.36) and stores the d components of $\vec{q}_h$ inside an STL map for every occurrence of a new location. When assembling the stiffness matrix for the computation of m_1^c, the same quadrature points as for the assembly of the stiffness matrix for computing m_0^c, are revisited. In the second run, the assembly takes less than 40% of the time of the first run[3].

[1] See e.g. Tables 7.9 or 7.11 for some figures.

[2] Curiously Recurring Tempate Pattern [Vandevoorde and Josuttis, 2002]

[3] Compare the DG matrix assembly time for m_0^c in Table 7.9 with the DG matrix assembly time for m_1^c in Table 7.10. The difference in the mesh Péclet numbers does not play a role in this comparison.

Chapter 7

Numerical Studies

In §7.1 and §7.2, we start with two singularly perturbed problems on the unit square for which analytical solutions exist in the domain of interest. Convergence tests can be performed using global and adaptive refinement. For the first problem, we demonstrate the influence of the ordering of unknowns on the performance of the iterative solvers for linear systems arising from a DG(1) discretization as described in section 5.7. The second problem has a less regular solution. In §7.3, we take a closer look on the quality of the diffusive L^2-projection compared to the SDFEM solution on a structured mesh. In §7.4 and §7.5, we show a 2-D and a 3-D example of the coupled groundwater flow and transport problem. Since analytical solutions are not available, numerical solutions computed on adaptively refined meshes are taken as reference solutions for assessing the quality of different solution methods computed on a coarse structured mesh.

For computations on structured meshes in 2-D and in 3-D, we use the `YASP` grid, an implementation of a structured parallel mesh available in the `dune-grid` module. For sequential adaptive refinement in 2-D, we use the `UG` grid [Bastian et al., 1997], and for parallel adaptive refinement with dynamic load-balancing in 3-D, we use the `DUNE ALUGrid` module [Dedner et al., 2014].

In all computations, the best available linear solver / preconditioner combination (in terms of robustness and speed) is chosen for each linear system arising from a finite element discretization of a stationary problem.

All time measurements are based on the **wall-clock time**, i.e. the difference between the time at which a certain task finishes and the start time of that task. It may include time that passes while waiting for resources to become available.

All 2-D computations are performed in sequential mode on a laptop with an *Intel®Core™2 Duo CPU (P9500, 2.53 GHz)* and 4 GB total memory. All 3-D computations are performed on a multi-core architectures with large memory and high-speed network communication links. Tables F.1 and F.2 give an overview of the used hardware.

7.1 An example with a regular solution

Let $(x, y) \in \overline{\Omega} = [0, 1]^2$ and consider the boundary value problem (from [John et al., 1997])

$$
\begin{aligned}
-\varepsilon \Delta u + \vec{q} \cdot \nabla u + \mu \cdot u &= \tilde{s}_\varepsilon \quad \text{in } \Omega & (7.1)\\
u &= 0 \quad \text{on } \partial\Omega & (7.2)
\end{aligned}
$$

where $\vec{q} = (2, 3)^T$, $\mu = 2$ and the source term $\tilde{s}_\varepsilon(x, y)$ is chosen such that

$$
u(x, y) = \frac{16}{\pi} x(1 - x)y(1 - y)\left(\frac{\pi}{2} + \arctan\left[\frac{2}{\sqrt{\varepsilon}}\xi(x, y)\right]\right) \tag{7.3}
$$

with

$$
\xi(x, y) = 0.25^2 - (x - 0.5)^2 - (y - 0.5)^2. \tag{7.4}
$$

For our tests, we choose $\varepsilon = 10^{-5}$. Note that in this example, the internal layer is generated by a source term which itself depends on ε. For $\varepsilon \ll 1$, an accurate representation of the source term requires a fine mesh, because, in a finite element discretization of $\tilde{s}_\varepsilon$, the error in the quadrature-rule might become dominating on a coarse mesh. Since we investigate the convergence behavior for global and adaptive refinement, this is not a severe problem.

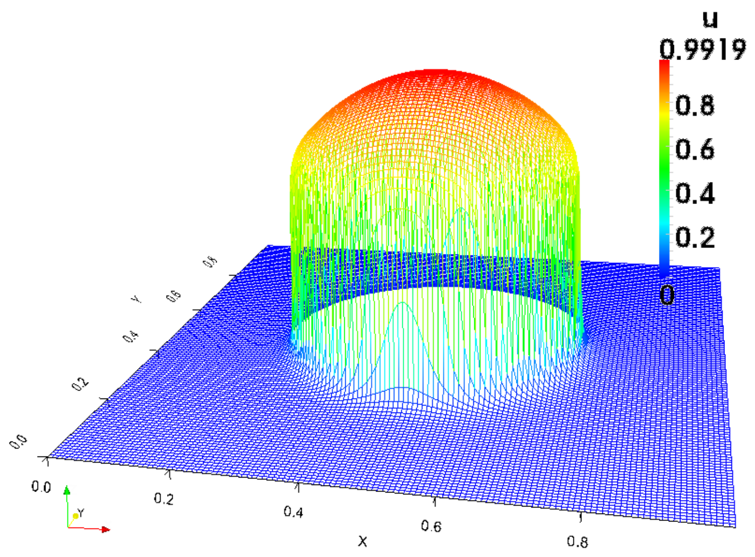


Figure 7.1: Profile of the analytical solution u for $\varepsilon = 10^{-5}$.

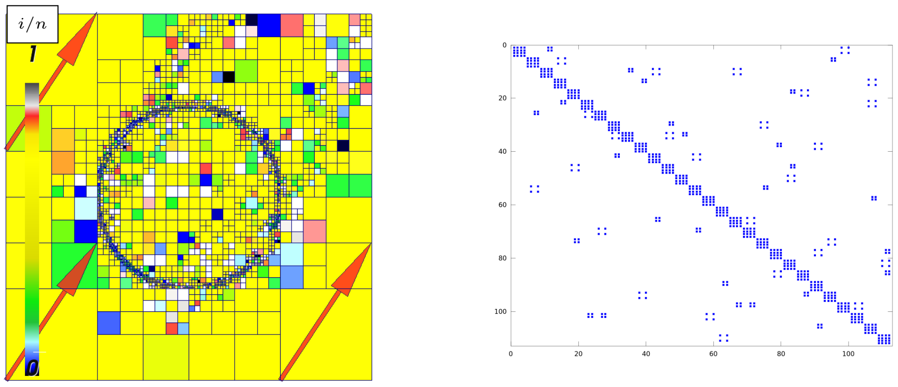
Linear solver performance and accuracy

Starting on a coarse structured mesh with $h_0 = 1/8$, we perform an adaptive refinement loop three times, based on three different ways of cell numbering as depicted in Figures 7.2(a)-(c). For the SDFEM discretization, the different cell numbering has no influence on the speed of the linear solver. For the DG(1) discretization, Table 7.1 confirms that an optimal numbering of degrees of freedom (following the velocity field) results in a faster solution of the arising linear system. The time for renumbering the grid cells is comparable to the time for one step of the iterative linear solver.

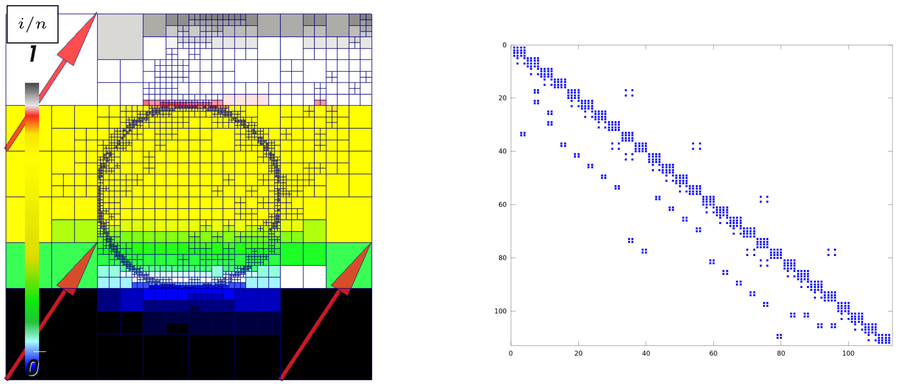
		random		horizontal		downstream		
L	DOF	IT	$TIT[s]$	IT	$TIT[s]$	IT	$TIT[s]$	$T_{\text{sort}}[s]$
9	6,856	8	0.005	4	0.004	1	0.008	0.003
10	10,528	9	0.008	5	0.007	1	0.008	0.006
11	19,672	13	0.015	6	0.014	2	0.010	0.012
12	34,540	17	0.028	7	0.026	2	0.028	0.022
13	85,468	25	0.081	9	0.079	3	0.068	0.059
14	150,016	34	0.152	10	0.138	4	0.127	0.123
15	278,836	48	0.325	12	0.262	5	0.246	0.284
16	544,300	73	0.770	14	0.516	7	0.496	0.661

Table 7.1: Performance of the linear solver (BiCGSTAB + SSOR with reduction 10^{-8}) for different cell numbering strategies applied to the DG(1) method: L = refinement level, DOF = degrees of freedom, IT = number of iterations for the linear solver, TIT = time per iteration, T_{sort} = time for renumbering the grid cells.

(a) random



(b) horizontal



(c) downstream

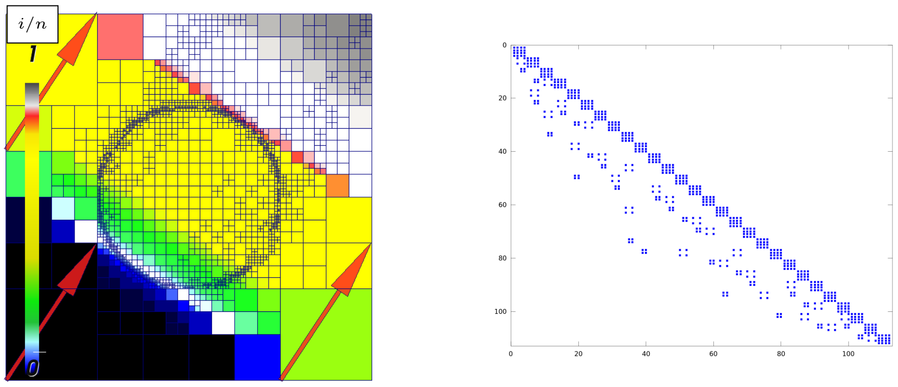


Figure 7.2: Different cell numbering strategies and corresponding matrix patterns for the DG(1) method. The block matrix for $\mathbb{Q}_1$ elements in 2-D is made of 4×4 - blocks. i = cell index, n = total number of mesh cells.

On coarse meshes, the matrix pattern can assume a block-triangular form (Figure 7.2(c)). In these cases, the iteration number is indeed 1. As refinement proceeds, the meshsizes and therefore the mesh Péclet numbers decrease and we diverge from the hyperbolic limit. Although this increases the iteration numbers, they stay at a low level for the optimal numbering. Not only the number of iterations is reduced but also the required time per iteration. This may be due to the fact that the non-zero matrix entries occur in a more compact format such that cache effects contribute to the performance boost. The linear solver used is BiCGSTAB with SSOR. Similar results are obtained with the combinations BiCGSTAB + ILU(0), GMRES + SSOR and GMRES + ILU(0).
We measure accuracy with respect to computing time. The SDFEM method with bilinear elements is compared to the DG(k) methods (with globally constant polynomial degree $k \in \{1, 2, 3\}$) in a convergence test with uniform and adaptive refinement. Accuracy is measured in the L^2-norm of the error taken over the domain of interest: $\|u - u_h\|_{L^2(\Omega)}$. The solution time is the sum of the system assembly time and the linear solver time. For the DG(k) methods, we apply an optimal numbering of mesh cells to speed up the linear solver. The time required to sort the mesh cells is negligible compared to the solution time.
The solution time is linearly proportional to the number of unknowns. For a comparable number of degrees of freedom (DOF), the iterative linear solvers perform better for the DG-based methods.

From Figure 7.3 we can make the following observations:

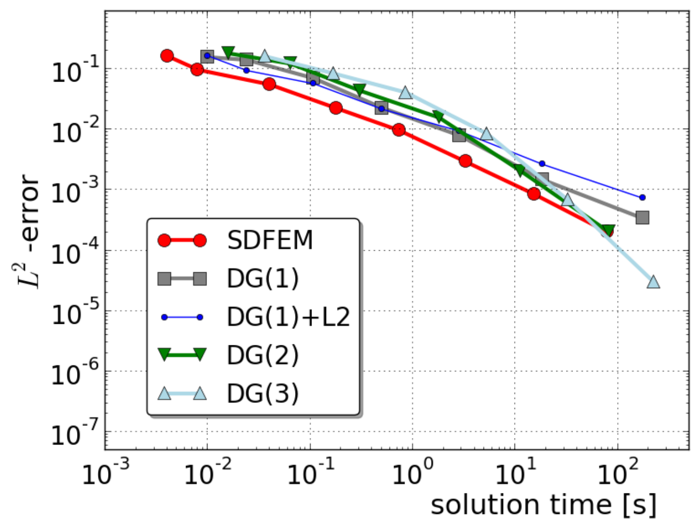
1. For the same computing time, SDFEM is more accurate than DG(1).

2. The accuracy of the higher order DG methods overtakes the accuracy of SDFEM at a certain refinement level. This happens as soon as the steep gradient is resolved and optimal convergence order is achieved.

3. With adaptive refinement, higher accuracy is achieved at an earlier stage.

The blue curve (DG(1)+L2) in the first plot displays the accuracy of the post-processed DG(1) solution on a structured mesh. Although it is close to the DG(1) curve, in this case, the diffusive L^2-projection adds an extra amount of error. Since solution time for the post-processing step is a small fraction of the solution time for the transport problem, it is neglected in this plot.

(a) global refinement



(b) adaptive refinement

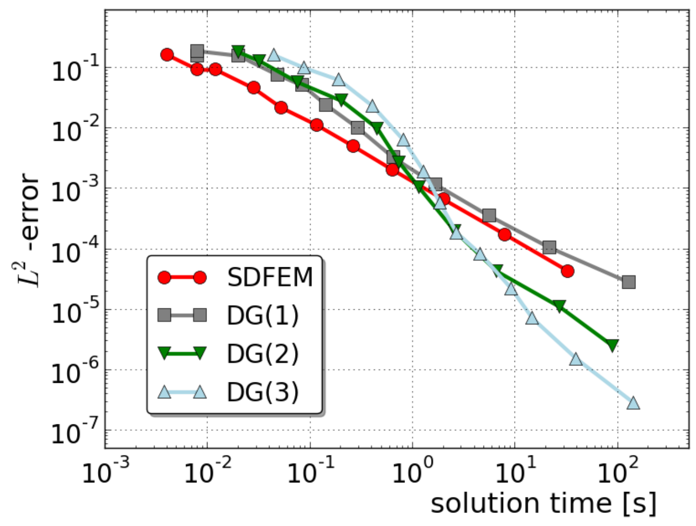


Figure 7.3: Example by John: (a) global refinement on a structured mesh and (b) adaptive refinement on an unstructured mesh (UG), both starting on a coarse mesh with meshsize $h = 1/8$. The refinement and coarsening fractions for the adaptive refinement algorithm are $p_r = 95[\%]$ and $p_c = 0.5[\%]$. Linear solver used: BiCGSTAB + SSOR with reduction 10^{-8}. Solution time = system assembly time + linear solver time.

7.2 An example with a less regular solution

Let $(x, y) \in [0,1]^2$. We consider the boundary value problem

$$-\varepsilon \Delta u + \vec{q}\,\nabla u = 0 \quad \text{in } [0,1]^2 \tag{7.5}$$

with constant velocity $\vec{q} = \frac{\sqrt{2}}{2}(1,1)^T$ and discontinuous boundary conditions

$$u(x,0) = 1 \qquad \text{and} \qquad u(0,y) = 0. \tag{7.6}$$

Obviously, the solution has a jump at the origin and is therefore not H^1-regular. This jump causes a characteristic boundary layer along the direction $\vec{q}$. This example is close to a real-world example in the sense that the discontinuity may be used to describe a binary state and the direction of the velocity is not aligned to the mesh. Theorem 1 of [López and Sinusía, 2004] provides an asymptotic expansion of the solution u on the subset $\Omega = [0,1]^2 \setminus \mathbf{U}_0$ where $\mathbf{U}_0 = \{\vec{y} \in \mathbb{R}^2 : \|\vec{y}\|_2 < r_0\}$ is a ball with radius $r_0 > 0$ and center $(0,0)$. Introducing polar coordinates through $x = r\sin\varphi$ and $y = r\cos\varphi$, we get

$$u = u_0(r,\varphi) + \frac{e^{wr(\sin(\varphi+\beta)-1)}}{\pi\sqrt{2wr}} u_1(r,\varphi) \tag{7.7}$$

where $\beta = \pi/4$, $w = \|\vec{q}\|_2/(2\varepsilon)$ and

$$u_0(r,\varphi) = \frac{1}{2}\begin{cases} \operatorname{erfc}\left(\sqrt{(1-\sin(\varphi+\beta))wr}\right) & \text{if } \varphi < \beta, \\ 1 & \text{if } \varphi = \beta, \\ 2 - \operatorname{erfc}\left(\sqrt{(1-\sin(\varphi+\beta))wr}\right) & \text{if } \varphi > \beta. \end{cases} \tag{7.8}$$

The function $u_1(r,\varphi)$ has an asymptotic expansion from which we use only the first term,

$$u_1(r,\varphi) = \begin{cases} \Gamma(1/2)\left\{\left(\frac{\cos(\varphi-\beta)}{\cos(\varphi+\beta)} - \frac{\cos(\varphi+\beta)}{\cos(\varphi-\beta)}\right) - \dfrac{1}{2\sin(\frac{1}{2}(\frac{\pi}{2}-\varphi-\beta))}\right\} & \text{if } \varphi \neq \pi/4, \\ 0 & \text{if } \varphi = \pi/4, \end{cases} \tag{7.9}$$

hereby neglecting higher order terms of ε.

For our tests, we choose $\varepsilon = 10^{-5}$ and $r_0 = 5 \times 10^{-5}$. The small area $\mathbf{U}_0(r_0)$ around the critical location $(0, 0)$, where the numerical errors are largest and a different asymptotic expansion is necessary, is left out of consideration.

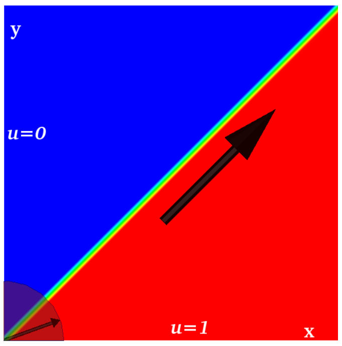


Figure 7.4: Reference solution for $\varepsilon = 10^{-5}$ on the domain of interest $\Omega = [0, 1]^2 \setminus \mathbf{U}_0$ ($r_0 = 5 \times 10^{-5}$).

Linear solver performance and accuracy

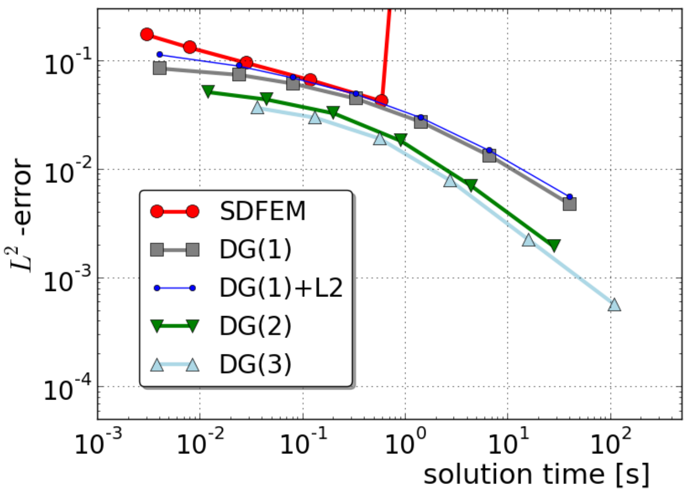
The same numerical experiments as in §7.1 are conducted on this example. The influence of renumbering cells on the linear solver performance for the DG(1) discretization are very similar to the results presented in §7.1 Table 7.1.

From Figure 7.5 we can see that the convergence behavior for the approximation of this less regular solution is different from the observations made in §7.1:

1. For the same solution time, the accuracy of DG(1) and SDFEM are comparable.
2. Higher order DG methods are more accurate than SDFEM right from the beginning.
3. With adaptive refinement, higher accuracy is achieved at an earlier stage.

The blue curve (DG(1)+L2) in the first plot is closer to the DG(1) curve than in the example of §7.1 (Figure 7.3). Furthermore, the SDFEM plot (red curve in Figure 7.5) stops after 4 refinement steps ($16, 641$ unknowns). In the next refinement step ($66, 049$ unknowns), the iterative linear solver BiCGSTAB with SSOR converges, but the solution is wrong. In the 6-th step ($263, 169$ unknowns), the iterative linear solver does not converge, although the linear system can still be solved using the direct solver `SuperLU`. This is most likely due to the fact that the large sparse system has become very ill-conditioned. However, direct solvers are not an option for large practical problems. `SuperLU` has reached the memory limit of the laptop already in the next refinement level where the number of unknowns is $1, 050, 625$.

(a) global refinement



(b) adaptive refinement

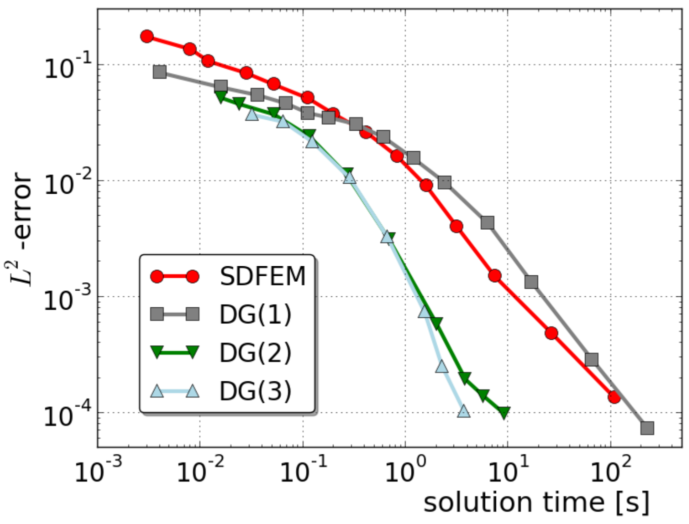


Figure 7.5: Example by López and Sinusía: (a) global refinement on a structured mesh and (b) adaptive refinement on an unstructured mesh (UG), both starting on a coarse mesh with meshsize $h_0 = 1/8$. The refinement fraction for adaptive refinement is $p_r = 95[\%]$ (no coarsening). The solution time = matrix assembly time + linear solver time. Linear solver used: BiCGSTAB with SSOR with a reduction of 10^{-8}.

7.3 Post-processed DG(1) versus SDFEM

Using the test problem from subsection 7.2, we take a closer look at the quality of the solution with respect to smearing effects and numerical over- and undershoots around the characteristic layer. We compare the post-processed DG(1) method with the SD-FEM method on structured meshes. Figure 7.6 shows the 3-D profile of four different numerical solutions on the whole domain $[0, 1]^2$. Figure 7.7 uses cross-sectional plots over the diagonal line between $(0, 1)$ and $(1, 0)$ to zoom into the steep front.

Observations from Figures 7.6 and 7.7:

Near the discontinuity in the boundary condition:

1. On the same refinement level, DG(1) exhibits larger over- and undershoots than SDFEM (see Figures 7.6 (a)+(c)).

2. The diffusive L^2-projection has a dampening effect on the DG(1) solution, reducing the amount of large over- and undershoots significantly (see Figure 7.6(c)+(d)) without smearing out the steep front beyond a mesh cell (see Figure 7.7 solution plots).

Globally:

3. While large over- and undershoots in the DG(1) solution are reduced efficiently, small over- and undershoots are merely dampened by the diffusive L^2-Projection (see Figure 7.7).

4. On the same refinement level, both DG(1) and DG(1)+L2 capture the location of the steep front more accurately than SDFEM throughout the domain (see Figures 7.6(a)+(c) and 7.7).

5. SDFEM on refinement level $L + 1$ captures the steep front as well as DG(1) or DG(1)+L2 on level L (comparable number of DOF) (see Figure 7.6(b)+(c) and Figure 7.7: blue line in (a) vs. red line in (b))

To achieve a comparable number of DOF as for the DG(1) method, the SDFEM method requires one extra level of global mesh refinement. The resulting matrix assembly time for SDFEM on the refined mesh is higher than for DG(1) on the coarse mesh.

(a)

$u_h = u_{sd}$

$h = 1/32$

$N = 1089$

$\epsilon = 0.096$

$u_{\max} = -1.045$

$u_{\min} = -0.047$

$T_{\text{sol}} = 0.028s$

(b)

$u_h = u_{sd}$

$h = 1/64$

$N = 4225$

$\epsilon = 0.066$

$u_{\max} = -1.046$

$u_{\min} = -0.048$

$T_{\text{sol}} = 0.12s$

(c)

$u_h = u_{dg}$

$h = 1/32$

$N = 4096$

$\epsilon = 0.062$

$u_{\max} = -1.249$

$u_{\min} = -0.249$

$T_{\text{sol}} = 0.08s$

(d)

$u_h = u_{cg}$

$h = 1/32$

$N = 1089$

$\epsilon = 0.069$

$u_{\max} = -1.042$

$u_{\min} = -0.042$

$T_{\text{sol}} = 0.10s$

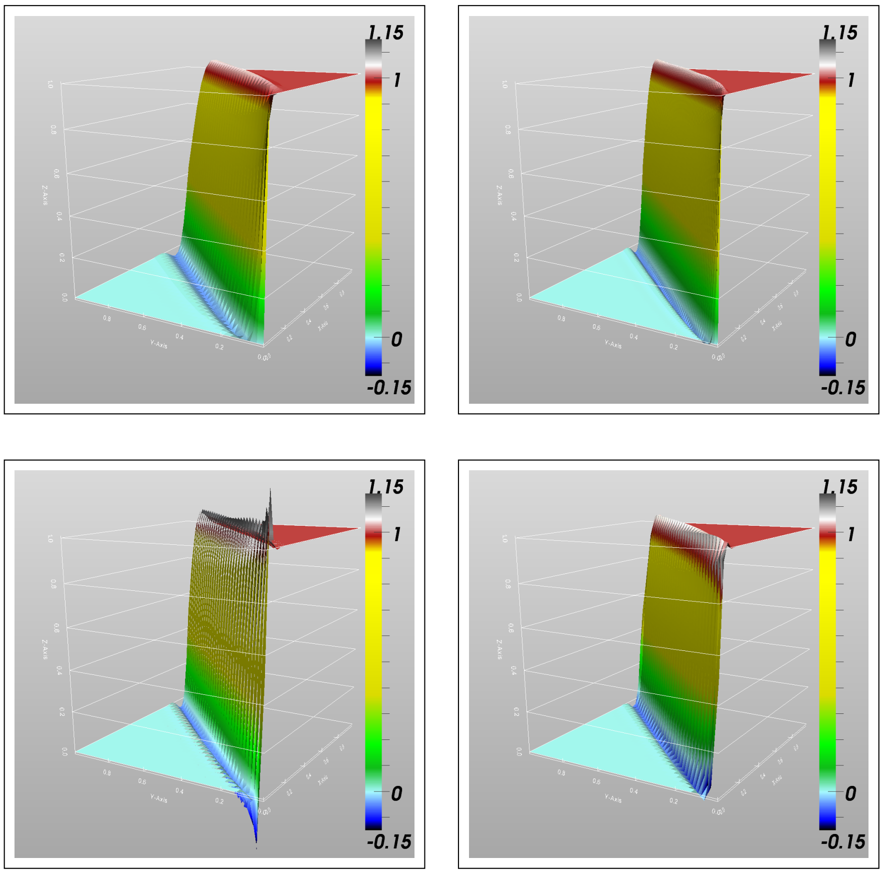


Figure 7.6: Warped plots (all from the same perspective) of the numerical solution for different methods: (a) and (b): u_{sd} is the SDFEM solution, (c): u_{dg} is the DG(1) solution, (d): u_{cg} is the diffusive L^2-projection of u_{dg} (DG(1)+L2). h is the uniform meshsize, N is the dimension of the solution space and $\epsilon = \|u - u_h\|_{L^2(\Omega)}$. $u_{\max}$ and $u_{\min}$ are the maximal and minimal values of the numerical solution u_h. T_{sol} is the solution time (matrix assembly + linear solver).

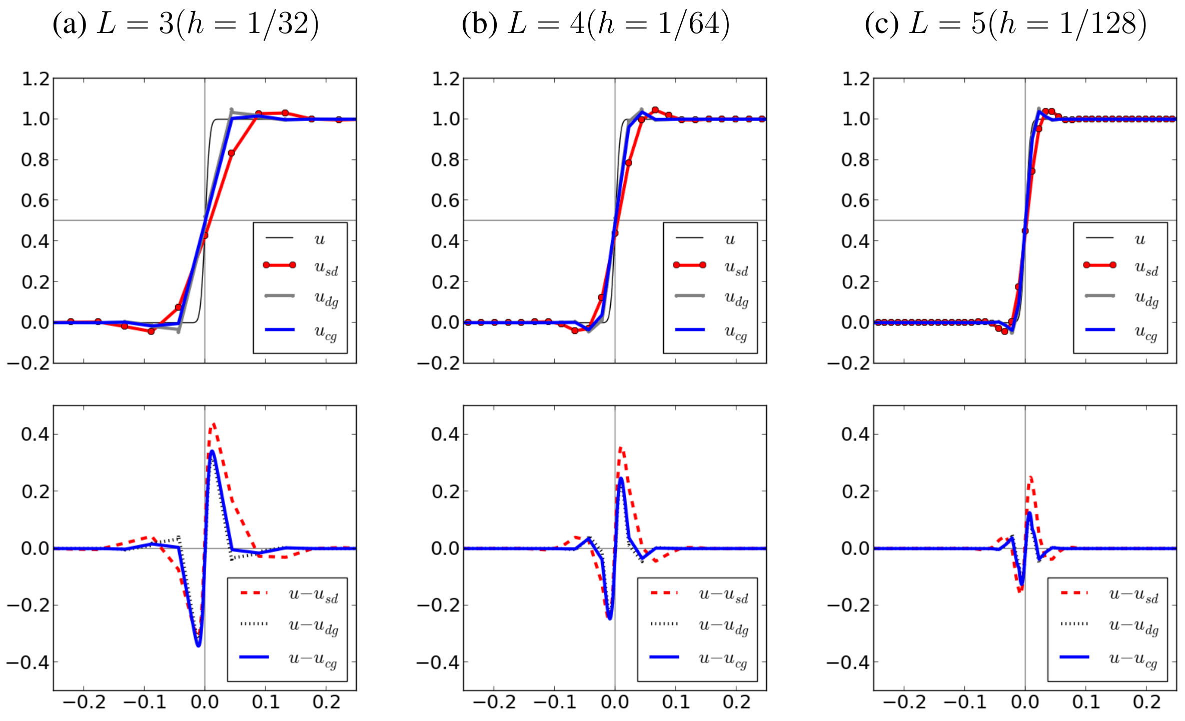


Figure 7.7: Zoomed plots of the solution and the absolute error for different methods along the diagonal line connecting (1,0) and (0,1). L is the global refinement level. u is the true solution resolved on a very fine mesh ($h = 10^{-5}$).

7.4 Forward transport in 2-D

In the following, we demonstrate the applicability of the presented DG methods to more realistic scenarios. We solve the groundwater flow equation (4.1) for the hydraulic head distribution ϕ and evaluate the velocity field (4.5) on the structured mesh $\mathcal{T}_{h_0}$. The solute transport equation (5.3) is then solved

- using adaptive DG(1) on a hierarchy of adaptively refined meshes $\{\mathcal{T}_\nu^{\text{adapt}}\}_{\nu\in\mathbb{N}}$, or
- using DG(k) with diffusive L^2-projection on the same mesh $\mathcal{T}_{h_0}$, or
- using SDFEM on the same mesh $\mathcal{T}_{h_0}$ or on a globally refined mesh $\mathcal{T}_1^{\text{global}}$.

The Gaussian field Y depicted in Figure 7.8(a) has the physical size of $100\times100[m^2]$, the resolution of the structured mesh $\mathcal{T}_0$ is 100×100 cells. Y is described by its mean value $\beta = -6.0$, its variance $\sigma^2 = 1.0$ and the correlation lengths $(\ell_x, \ell_y) = (10, 10)[m]$. The hydraulic head is prescribed on the left ($\phi\big|_{x=0} = 100[m]$) and on the right ($\phi\big|_{x=100} = 99.5[m]$) boundaries. The induced pressure gradient drives the main flow. The injection well parameters are $\tilde{w}_{\text{inj}} = 5 \times 10^{-4}[m^3/s]$, $\tilde{c}_{\text{inj}} = 1[g/m^3]$ and $T_{\text{inj}} = 100[s]$. The stationary solution values range between 0 and $100[gs/m^3]$. The molecular diffusion is $D_m = 2 \times 10^{-9}[m^2/s]$, the longitudinal and transversal dispersivities are $\alpha_l = 10^{-3}[m]$ and $\alpha_t = 10^{-4}[m]$.

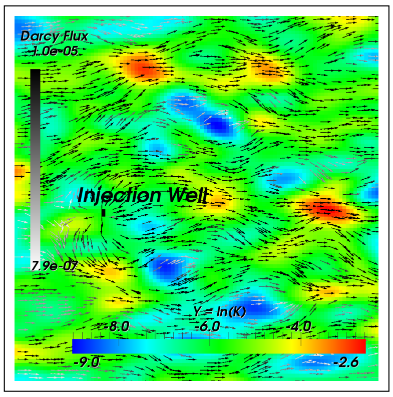
Theoretically, u_h tends to the true solution u as the meshsize h tends to 0. At least, all numerical oscillations should disappear if the meshsizes near the characteristic layer become so small that the effective mesh Péclet numbers[1] are smaller than 1. Working with $\mathcal{O}(\alpha_t) \sim 10^{-4}$ and a base level meshsize $\mathcal{O}(h) \sim 1$ we would get $\mathcal{O}(\mathcal{P}_h^t) \sim 10^3$. It would require 10 levels of global refinement to achieve $\mathcal{O}(\mathcal{P}_h^t) \sim 1$, but already after 7 levels of global refinement, our mesh would have more than 10^8 cells. The problem size would become extremely large for a 2-D simulation.

Adaptive mesh refinement is the only chance to keep the problem size orders of magnitude lower while reducing mesh Péclet numbers at the steep fronts. We choose the stopping criterion (5.53) from section 5.5 with a tolerance of $p_{\text{osc}} = 1\%$. The result is achieved after 5 steps of adaptive refinement (Table 7.2 and Figure 7.8(b)). This solution is taken as the reference solution for assessing the quality of different methods on the structured mesh (Table 7.3 and Figure 7.8).

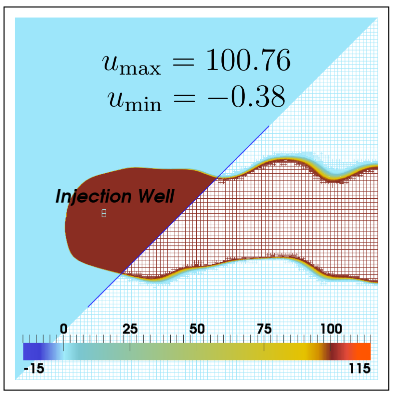
[1]Replace α_ℓ by α_t in (5.20) to consider longitudinal effects.

2-D solute transport: $u = m_0^c$

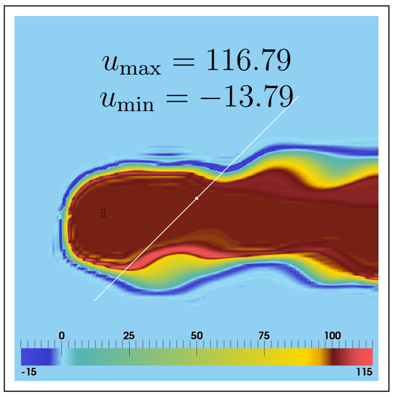
(a)



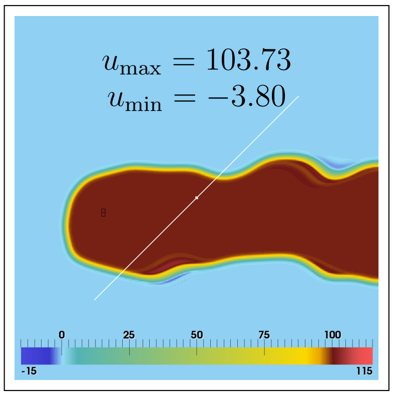
(b) adaptive DG(1), level 7



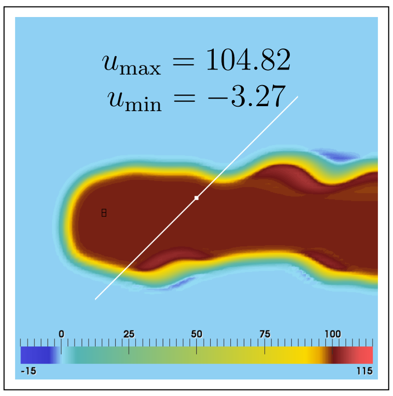
(c) SDFEM



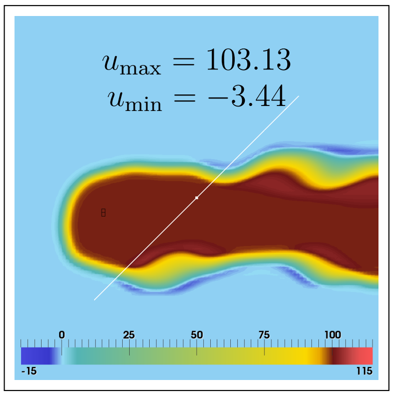
(d) Post-processed DG(3)



(e) Post-processed DG(2)

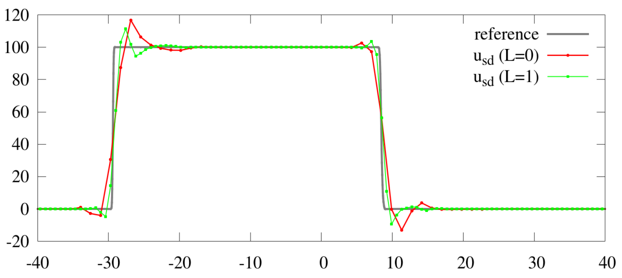


(f) Post-processed DG(1)

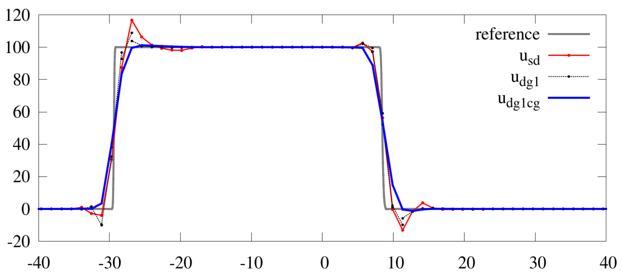


Cross sectional plots along the diagonal line indicated above:

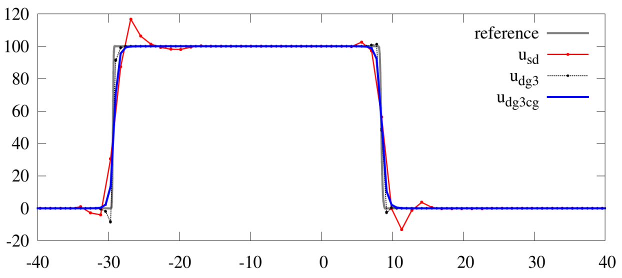
(g) SDFEM



(h) post-processed DG(1)



(i) post-processed DG(3)



(j) post-processed DG(2)

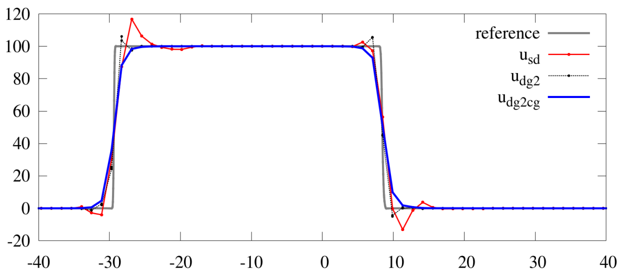


Figure 7.8: (a) Gaussian field $Y = \ln(K)$ and velocity field $\vec{q}_h$ on $\mathcal{T}_0$. (b) Adaptive DG(1) solution with $p_{\text{osc}} \leq 1\%$ is taken to be the reference solution for the stationary transport problem. (c) - (f) Comparing different solutions on the coarse structured mesh $\mathcal{T}_0$ with 100×100 cells. (g) - (j): SDFEM solution u_{sd} on $\mathcal{T}_0$ and on $\mathcal{T}_1^{\text{global}}$ compared to DG(k) solutions (u_{dgk}) on $\mathcal{T}_0$ and post-processed DG(k) solutions (u_{dgkcg}) on $\mathcal{T}_0$ along the indicated cut-line (for $k = 1, 2, 3$). $u_{\max}$ and $u_{\min}$ are the maximal and minimal values of the numerical solution.

		Adaptive DG(1)						
L	DOF	max. $\mathcal{P}_h^t$	$M[s]$	$T[s]$	IT	TIT	$u_{\min}$	$u_{\max}$
0	40,000	4675.89	0.91	0.04	4	0.011	134.47	-33.11
1	64,000	2232.44	1.49	0.08	4	0.018	132.26	-31.77
2	102,364	1105.47	2.42	0.25	8	0.033	127.38	-24.68
3	162,964	545.85	3.93	0.75	14	0.050	112.48	-12.10
4	259,372	272.94	6.29	2.29	26	0.076	104.99	-3.39
5	413,860	136.48	10.26	7.29	52	0.110	100.76	-0.38

Table 7.2: Adaptive mesh refinement on UG, with $p_r = 20[\%]$ and $p_c = 10[\%]$. Renumbering mesh cells on mesh level $L = 5$ takes 0.16 sec. Linear solver used: BiCGSTAB + ILU(0) with reduction 10^{-8}. M = matrix assembly time, IT = linear solver iterations, T = linear solver time

Observations:

1. On the coarse mesh $\mathcal{T}_0$, DG(k) solutions with $k = 1, 2, 3$ may locally exhibit stronger over- and undershoots than the SDFEM solution, but these can be reduced very efficiently with a diffusive L^2-projection (see the runtimes of M and T for the diffusive L^2-projection in Table 7.3 and Figure 7.8(h)-(j)).

2. The over- and undershoots of the SDFEM solution may oscillate into the domain surrounding the steep front (see Figure 7.8(c)) whereas over- and undershoots of the DG solutions do not show this behavior.

3. On the same mesh $\mathcal{T}_0$, higher order polynomials can be used to improve the quality of the DG solution with respect to the sharpness of the steep front. The areas with over- and undershoots are shrunk. A diffusive L^2-projection preserves this behavior and reduces over- and undershoots (see Figure 7.8(d)-(f)).

4. The SDFEM method applied to a globally refined mesh $\mathcal{T}_1^{\text{global}}$ requires a comparable number of unknowns as the DG(1) method on $\mathcal{T}_0$. The steep front is resolved similarly well, but the matrix assembly and the linear solution takes longer ($L = 1$ in Table 7.3 and Figure 7.8).

Global h- or p-refinement on a structured mesh						
Method	DOF	$M[s]$	IT	$T[s]$	u_{min}	u_{max}
SDFEM ($L = 0$)	10,201	0.2	12	0.02	-13.79	116.79
SDFEM ($L = 1$)	40,401	0.8	28	0.26	-15.29	113.43
DG(1)	40,000	0.6	4	0.04	-33.11	134.47
diffusive L^2-proj.	10,201	+0.05	1	+0.02	-3.44	103.13
DG(2)	90,000	1.7	4	0.13	-34.04	135.88
diffusive L^2-proj.	10,201	+0.09	1	+0.02	-3.27	104.82
DG(3)	160,000	4.5	4	0.26	-36.69	135.94
diffusive L^2-proj.	10,201	+0.16	1	+0.02	-3.80	103.73

Table 7.3: Computations on a structured mesh: SDFEM on mesh level $L = 0$ and $L = 1$ versus post-processed DG methods on mesh level $L = 0$ with different polynomial orders. Renumbering mesh cells on $L = 0$ for the DG methods takes 0.004 sec. Linear solver used for solving the transport equation: BiCGSTAB + ILU(0) with reduction 10^{-8}. Linear solver used for the diffusive L^2-projection: BiCGSTAB + AMG with reduction 10^{-8}. M = matrix assembly time, IT = linear solver iteration number, T = linear solver time.

7.5 Forward transport in 3-D

A 3-D simulation with a similar setting is started on a coarse mesh $\mathcal{T}_{h_0}$ with $32 \times 32 \times 32$ cells. The domain extensions are $(L_x, L_y, L_z) = (10, 10, 10)[m]$. The geostatistical field parameters for the Gaussian model are $\beta = -6.0$, $\sigma^2 = 1.0$ and $(\ell_x, \ell_y, \ell_z) = (2, 2, 1)[m]$. The hydraulic head is prescribed on the left boundary by $\phi\big|_{x=0} = 10[m]$ and on the right boundary by $\phi\big|_{x=10} = 9.8[m]$. An injection well is placed at the position $(x, y) = (2.1, 5.1)[m]$ and its z-range is $[0... - 5][m]$. The injection well parameters are $\tilde{w} = 1 \times 10^{-2}[m^3/s]$, $\tilde{c} = 1[g/m^3]$ and $T_{\text{inj}} = 100[s]$. The values of the stationary solution range between 0 and $100[gs/m^3]$.

Doing the same analysis as for the 2-D case, we find that with global refinement, we would end up with more than 10^{13} cells after 10 refinement steps in order to achieve $\mathcal{O}(\mathcal{P}_h^t) \sim 1$. The adaptive DG(1) solution in Table 7.4 after 9 steps shows a sharp resolution of the steep front (Figure 7.9(c)). However, there are thin layers where the over- and undershoots exceed 25%. This is still far away from being an acceptable reference solution. To achieve our targeted reduction of under- and overshoots below $p_{\text{osc}} = 5\%$, further refinement steps with increasing memory consumption and solution time are necessary.

We make very similar observations as in the 2-D case:

1. The over- and undershoots generated by the DG(k) solutions with $k = 1, 2, 3$ can be reduced very efficiently with a diffusive L^2-projection (see the runtimes of M and T for the diffusive L^2-projection in Table 7.5).

2. The over- and undershoots of the SDFEM solution may oscillate into the domain surrounding the steep front (see Figure 7.9(d)) whereas over- and undershoots of the DG solutions do not show this behavior.

3. On the same mesh $\mathcal{T}_0$, higher order polynomials can be used to improve the quality of the DG solution with respect to the sharpness of the steep front. The areas with over- and undershoots are shrunk. A diffusive L^2-projection preserves this behavior and reduces over- and undershoots (see Figure 7.9(e)+(f)).

4. The SDFEM method applied to a globally refined mesh $\mathcal{T}_1^{\text{global}}$ requires a comparable number of unknowns as the DG(1) method on $\mathcal{T}_0$. The steep front is resolved similarly well (not shown here), but the matrix assembly and the linear solution takes longer ($L = 1$ in Table 7.3).

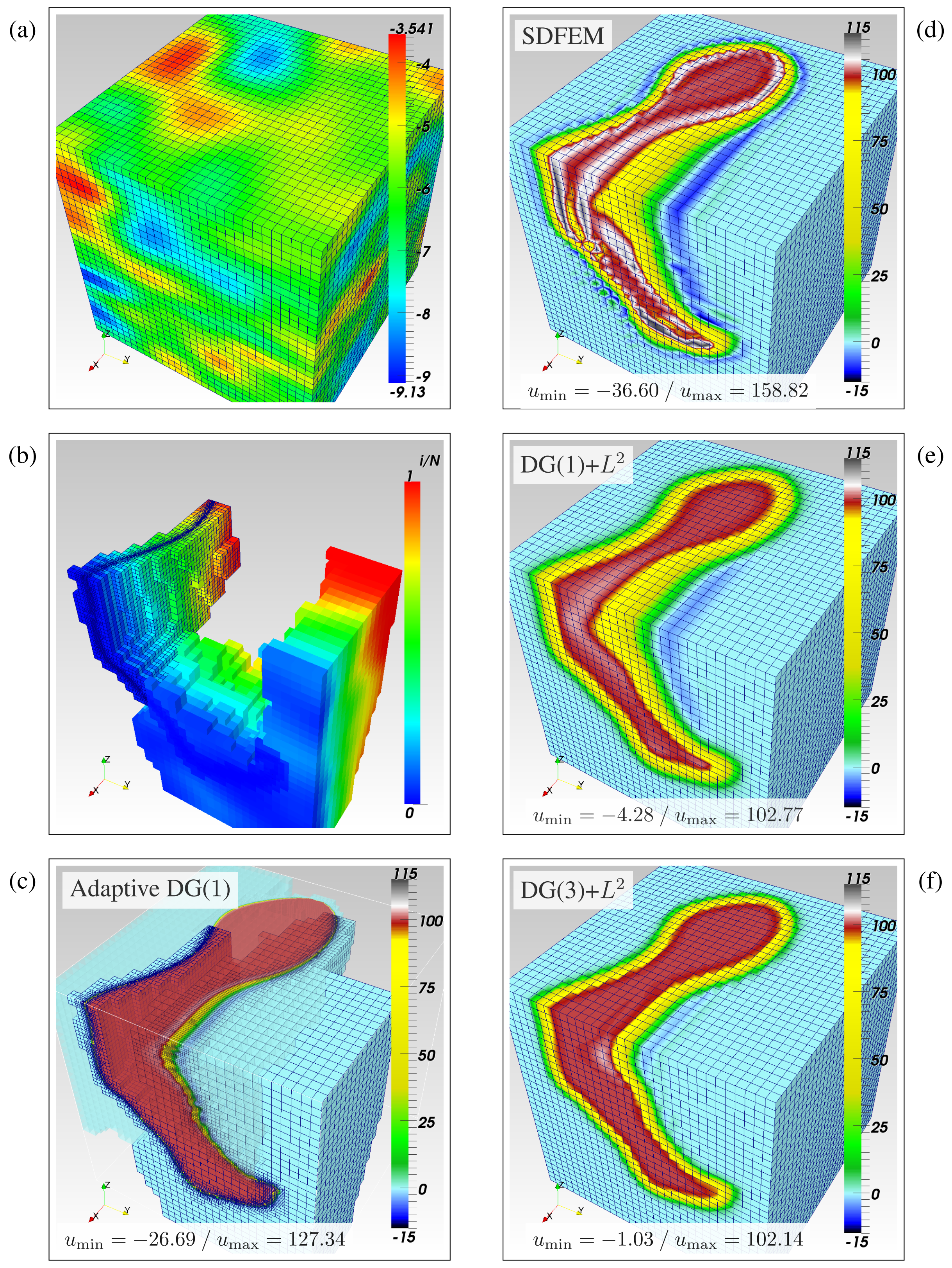

(a)
-3.541
-4
-5
-6
-7
-8
-9
-9.13
(b)
i/N
1
0
(c)
Adaptive DG(1)
$u_{\min} = -26.69$ / $u_{\max} = 127.34$
(d)
SDFEM
$u_{\min} = -36.60$ / $u_{\max} = 158.82$
(e)
DG(1)+L^2
$u_{\min} = -4.28$ / $u_{\max} = 102.77$
(f)
DG(3)+L^2
$u_{\min} = -1.03$ / $u_{\max} = 102.14$
115
100
75
50
25
0
-15

Figure 7.9: (a): Gaussian field $Y = \ln(K)$ on $\mathcal{T}_0$ with $32 \times 32 \times 32$ cells. (b)-(c): Illustrating parallel adaptive refinement with dynamic load-balancing (step 9). A reduction of under- and overshoots below 5% is possible, see Table 7.4. (d)-(f): Comparing different solutions on the coarse mesh $\mathcal{T}_0$. $u_{\max}$ and $u_{\min}$ are the maximal and minimal values of the displayed numerical solution.

Adaptive DG(1)						
L	DOF	IT	$T[s]$	$u_{\min}$	$u_{\max}$	$\mathcal{P}_h^t$
0	346,560	8	0.7	-38.8	136.59	1562
1	399,024	11	1.1	-38.7	142.70	1560
2	537,240	12	1.6	-44.8	144.25	1560
3	814,152	12	3.2	-50.7	141.17	1560
4	1,410,136	18	8.2	-50.2	142.59	389
5	2,740,704	17	15	-56.3	142.78	194
6	5,275,496	18	31	-43.1	147.91	194
7	9,273,552	24	88	-38.8	151.62	97
8	15,368,112	28	152	-28.9	132.08	97
9	23,806,944	35	349	-26.7	127.34	48
10	39,897,040	52	743	-24.0	121.37	48
11	59,903,672	64	1425	-18.7	117.17	24
12	70,498,808	67	1935	-15.2	114.24	24
13	95,837,712	87	3926	-15.5	112.58	24
14	132,771,680	95	5724	-11.7	113.12	24
15	188,298,456	110	8854	-10.5	110.78	24
16	242,272,328	128	13489	-6.88	106.72	24
17	321,793,400	167	28075	-6.59	106.73	24
18	348,929,992	185	30403	-5.30	105.37	24
19	433,481,584	200	38251	-5.23	104.85	12
20	474,804,264	238	61351	-4.98	104.57	12

Table 7.4: 3-D parallel adaptive refinement on `ALUGrid`, with $p_r = 70[\%]$ and $p_c = 5[\%]$. Renumbering mesh cells on level 20 takes 24.5 sec. Linear solver used: BiCGSTAB + SSOR with reduction 10^{-8}. L = refinement level, IT = linear solver iterations, T = linear solver time. The computation is performed on `quadxeon4`. $P = 16$ cores are used for the computation, more than 66% of the RAM is required for the linear solver in step 20 alone.

Global h- or p-refinement on a structured mesh						
Method	DOF	$M[s]$	IT	$T[s]$	$u_{\min}$	$u_{\max}$
SDFEM ($L = 0$)	58,806	0.5	14	0.11	-36.6	158.8
SDFEM ($L = 1$)	363,350	2.7	22	1.12	-36.5	139.1
DG(1)	376,832	2.3	7	0.38	-37.9	136.4
diffusive L^2-proj.	58,806	+0.2	2	+0.1	-4.3	102.8
DG(2)	1,271,808	22.5	8	3.8	-50.8	146.7
diffusive L^2-proj.	58,806	+0.5	2	+0.1	-3.07	103.4
DG(3)	3,014,656	190.1	9	38.4	-47.1	152.0
diffusive L^2-proj.	58,806	+1.2	2	+0.1	-1.0	102.1

Table 7.5: 3-D parallel computations with $P = 8$ cores on `fna` (see Table F.2) using a structured mesh with partitioning $(P_x, P_y, P_z) = (1, 8, 1)$ and overlap $= 1$. SDFEM on mesh levels $L = 0$ and $L = 1$ compared to DG methods on the coarse mesh level $L = 0$ with different polynomial orders. Parallel renumbering of mesh cells for the DG methods on level $L = 0$ takes 0.004 sec. Linear solver used for solving the transport equation: BiCGSTAB + ILU(0) with reduction 10^{-8}. Linear solver used for the diffusive L^2-projection: BiCGSTAB + AMG with reduction 10^{-8}. M = matrix assembly time, IT = number of iterations, T = linear solver time.

7.6 Simulations in a nested cells environment

With respect to the efficiency (solution time) and the quality of the solution (maximal amplitude of the over- and undershoots and smearing effects at the steep fronts) for the convective-dominant transport problem, the observations made in the previous four sections §7.2– §7.5 favor the combination CCFV / DG(1) + diffusive L^2-projection over the FEM / SDFEM approach. A further comparison in this section will definitely help make the decision.

Without doubt, the best possible solution in terms of the L^2-error is achieved with adaptive mesh refinement. However, numerical oscillations can be reduced to an acceptable level only if the mesh cells at the steep front become so small that their local mesh Péclet numbers approach 1 (diffusion-dominant problem). This comes at a very high price, especially in 3-D. A "perfect" solution in this sense may not be necessary for a robust inversion scheme that can cope with noisy data. This will be verified in two separate test scenarios, in 2-D using adaptive mesh refinement (§7.6.2) and in 3-D by adding extra noise to the data (§7.6.6). Therefore, we decide to apply the diffusive L^2-projection rather than local mesh refinement to dampen the numerical over- and undershoots in the DG(1) solution of the transport equation during the inversion process. Another benefit of this decision is that the same structured mesh as the one on which the parameter field is resolved can used for the DG(1) discretization.

We conclude this chapter by applying the proposed methods to a real-world example as applied in field applications. In all upcoming 3-D computations, the structured parallel mesh (`YASP` grid) with an overlap of one cell is used.

7.6.1 The setting

The nested cell setup introduced by Luo et al. [2006] is a four well-system made of two injection and two extraction wells with adjustable pumping rates. The outer pair spanning the outer cell generates a flow field that acts as a shield,

1. protecting the inner cell, where measurements are being taken, from the influence of ambient flow,
2. minimizing tracer leakage and
3. devising a control mechanism for the size of the recirculation zone and residence times within the inner cell.

Provided that each pumping well can flexibly be utilized as an injection or extraction well, this setup can be employed in four principle directions using eight pumping wells as depicted by the triangles in Figure 7.10.

Geometry of the domain Ω	
extensions	$(\ L_x,\ L_y,\ L_z\ ) = (\ 51.2,\ 51.2,\ 5\ )[m]$
Geostatistical parameters of the original log conductivity field (prior knowledge)	
prior variance of Y_{orig}	$\sigma^2 = 1.0$
#zones	$N_\beta = 1$
prior mean of the trend	$\beta^* = -6.0$
uncertainty of the trend	$\sigma_\beta^2 = 0.1$ or $\sigma_\beta = 0.316$
correlation lengths:	$(\ \ell_x,\ \ell_y,\ \ell_z\ ) = (\ 1.0,\ 1.0,\ 0.5\ )[m]$
variogram model:	spherical
Transport parameters	
porosity	$\theta = 0.3$
molecular diffusion coefficient	$D_m = 10^{-9}[m^2/s]$
longitudinal dispersivity	$\alpha_\ell = 10^{-3}[m]$
transversal dispersivity	$\alpha_t = 10^{-4}[m]$
Flow parameters (ambient flow combined with nested cells)	
boundary conditions for ϕ:	Dirichlet B.C. (4.2) at west and east: $\phi\big\|_{x=0} = 100.00[m]$, $\phi\big\|_{x=51.2} = 99.90[m]$,
	no-flow Neumann B.C. (4.3) at all other boundaries
setup #1 (Figure 7.10)	outer injection well at $(10, 25, [-0.5 ... -4.5])[m]$ with $\tilde{w}_{\text{inj}} = 0.008[m^3/s]$
	inner injection well at $(17, 25)[m]$, screening sections
	at $z \in [-0.5 ... -1.5][m]$ with $\tilde{w}_{\text{inj}} = 0.002[m^3/s]$
	at $z \in [-2.0 ... -3.0][m]$ with $\tilde{w}_{\text{inj}} = 0.002[m^3/s]$ filled with tracer
	at $z \in [-3.5 ... -4.5][m]$ with $\tilde{w}_{\text{inj}} = 0.002[m^3/s]$
	inner extraction well at $(33, 25, [-0.5 ... -4.5])[m]$ with $\tilde{w}_{\text{ext}} = 0.009[m^3/s]$
	outer extraction well at $(40, 25, [-0.5 ... -4.5])[m]$ with $\tilde{w}_{\text{ext}} = 0.010[m^3/s]$
setup #2/#3/#4	= rotation of setup #1 by $90°/180°/270°$ around $(25, 25, z)[m]$
Measurement locations and types	
point measurements at $(\ x,\ y,\ z\ )$	with $x, y \in \{\ 20.5,\ 23.5,\ 26.5,\ 29.5\ \}[m]$ and $z \in \{\ -1,\ -2,\ -3,\ -4\ \}[m]$
measurement types	ϕ (= piezometric head) and m_1^c ($\sim$ arrival time)
measurement error of ϕ	(3.21) with $\epsilon_\ell^{(\text{abs})} = 0.005[m]$
measurement error of m_1^c	(3.21) with $\epsilon_\ell^{(\text{rel})} = 5[\%]$ and a fixed value of $\epsilon_\ell^{(\text{abs})}$
total #measurements	$M = 4$ setups $\times$ 64 points $\times$ 2 types $= 512$ measurements
Thresholds fine-tuning the inversion algorithm 3.1	
$k_{\max} = 20, \quad \alpha_{\min} = 0.001, \quad \Delta_J = 2\% \cdot M, \quad \Delta_{\boldsymbol{y}} = 0.01$	

Table 7.6: 3-D nested cells environment with four different setups

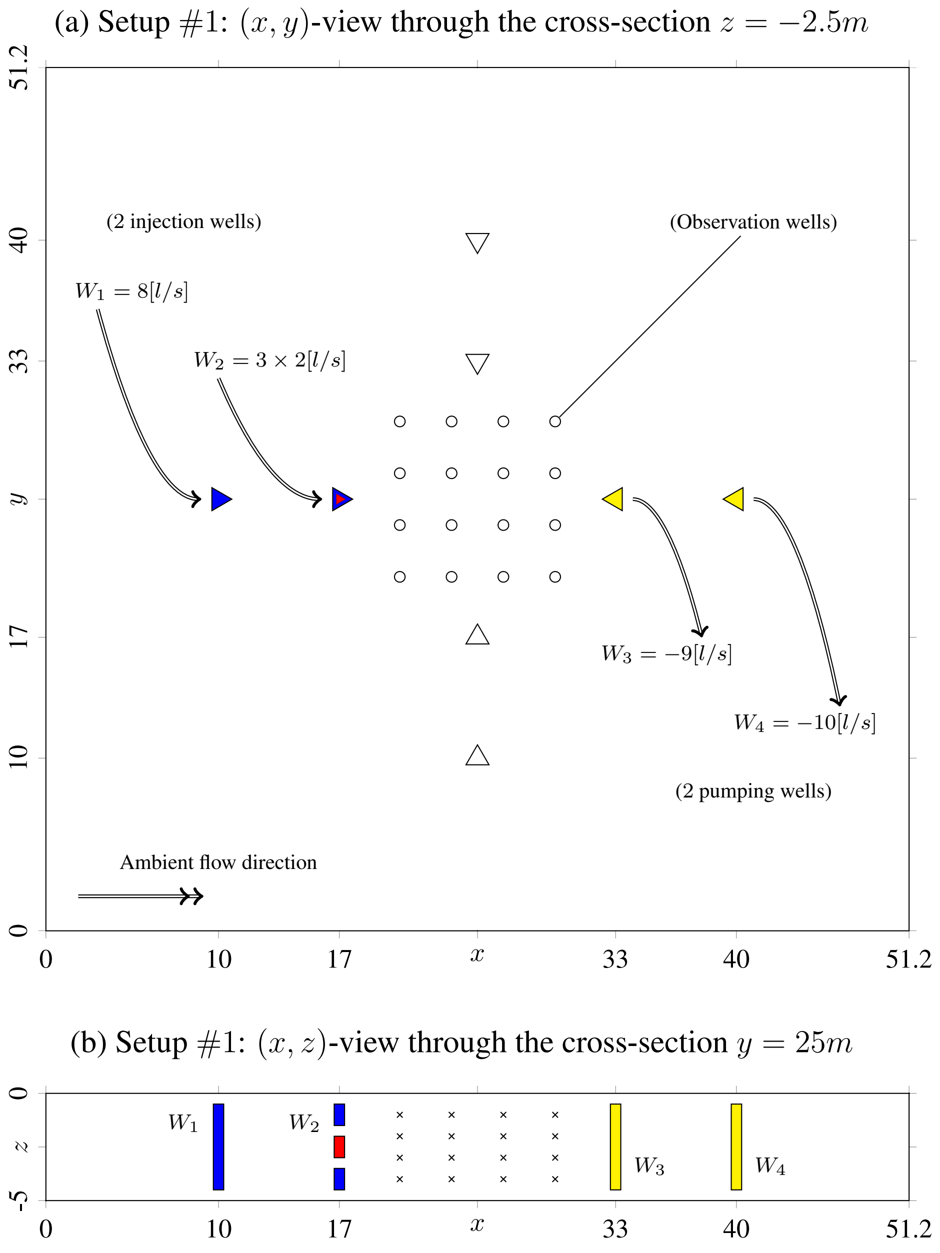


Figure 7.10: All 24 boreholes house a vertical well-pipe, partially filled with inflatable packers so that discrete depth intervals can be defined for different purposes. Color code: fresh water injection (blue), tracer injection (red), pumping (yellow). (a) Injection and pumping wells can be switched. The other three setups are obtained by a successive rotation of setup $\#1$ by an angle of 90° around the central axis $(25, 25, z)[m]$. The observation wells are identical in all four setups. (b) Well screens and z-locations of observation points ($\times$).

A QLGA-based inversion using heated water as a tracer was carried out in such a setting by Schwede et al. [2014]. The SDFEM method was used to discretize the heat transport equation. Solute transport in groundwater is much more convection-dominant than heat transport.
In this section, we investigate **in five test scenarios** for a convection-dominant test case listed in Table 7.6

(1) the influence of adaptivity on the solution of the inverse problem (2-D),

(2) the performance of FEM/SDFEM versus CCFV/DG(1) for the solution of the coupled forward problems (ϕ, m_0^c, m_1^c) (3-D),

(3) the parallel scalability of the forward solvers based on CCFV/DG(1) (3-D),

(4) the applicability of CCFV/DG(1) as a forward solver in the context of a massively parallel fully coupled inversion (3-D),

(5) and, last but not least, the stability of the inversion scheme with respect to disturbances in the data (3-D).

7.6.2 Scenario 1: Influence of adaptivity in a fully coupled 2-D inversion

We consider the setting from Table 7.6 and Figure 7.10 *without the third space dimension* z. The domain $\Omega = [0, 51.2] \times [0, 51.2]$ is discretized using an initially structured mesh $\mathcal{T}_{h_0}$ with 128×128 cells. The randomly generated field in Figure 7.11(a) is taken to be the original Y-field. For all four setups, we solve the forward problems to generate measurement data. The flow equation (4.1)[1] for ϕ is solved using the CCFV method on the 'coarse' mesh $\mathcal{T}_{h_0}$. The forward steady-state transport equations (4.21) for the temporal moments m_0^c and m_1^c are solved in two different ways:

(I) We use the combination DG(1) + diffusive L^2-projection on the 'coarse' mesh $\mathcal{T}_{h_0}$ (`YASP`).

(II) We use adaptive DG(1) on a sequence of locally refined meshes $\{\mathcal{T}_{h_\nu}^{\text{adapt}}\}_\nu$ (`UG` in sequential mode only). The residual based error estimator (5.49) is applied to $u := m_1^c$ with error-fraction marking strategy. Both equations for m_0^c and m_1^c are solved on the same mesh on each refinement level. The local refinement is proceeded until the maximal over- and undershoots in the discrete solutions are below 5%.

[1]Unlike in the 3-D setting, the Dirichlet boundary conditions are given by $\phi|_{x=0} = 100[m]$ and $\phi|_{x=51.2} = 99.88[m]$. The inner/outer well injection/extraction rates are given by $\tilde{w}_{\text{inj}} = +0.8/+2.4[l/s]$. and $\tilde{w}_{\text{ext}} = -0.8/-2.4[l/s]$.

			Setup #1		Setup #2		Setup #3		Setup #4	
			Case I	Case II	Case I	Case II	Case I	Case II	Case I	Case II
	x	y	m_1^c		m_1^c		m_1^c		m_1^c	
1	20.5	20.5	$\mathbf{1.31 \cdot 10^4}$	$-6.24 \cdot 10^{-2}$	$1.60 \cdot 10^6$	$1.76 \cdot 10^6$	$1.03 \cdot 10^7$	$1.04 \cdot 10^7$	$6.22 \cdot 10^6$	$6.21 \cdot 10^6$
2	23.5	20.5	$\mathbf{3.46 \cdot 10^4}$	$7.09 \cdot 10^0$	$1.36 \cdot 10^6$	$1.41 \cdot 10^6$	$6.40 \cdot 10^6$	$6.01 \cdot 10^6$	$4.38 \cdot 10^6$	$5.59 \cdot 10^6$
3	26.5	20.5	$\mathbf{1.00 \cdot 10^4}$	$-7.43 \cdot 10^0$	$1.15 \cdot 10^6$	$1.20 \cdot 10^6$	$4.75 \cdot 10^6$	$4.71 \cdot 10^6$	$4.28 \cdot 10^6$	$4.09 \cdot 10^6$
4	29.5	20.5	$\mathbf{3.32 \cdot 10^3}$	$3.76 \cdot 10^0$	$1.59 \cdot 10^6$	$2.04 \cdot 10^6$	$2.16 \cdot 10^6$	$2.50 \cdot 10^6$	$3.35 \cdot 10^6$	$6.80 \cdot 10^6$
5	20.5	23.5	$5.75 \cdot 10^5$	$4.86 \cdot 10^5$	$2.83 \cdot 10^6$	$3.74 \cdot 10^6$	$2.69 \cdot 10^6$	$2.69 \cdot 10^6$	$5.06 \cdot 10^6$	$5.40 \cdot 10^6$
6	23.5	23.5	$1.13 \cdot 10^6$	$1.12 \cdot 10^6$	$2.33 \cdot 10^6$	$2.07 \cdot 10^6$	$2.41 \cdot 10^6$	$2.17 \cdot 10^6$	$2.40 \cdot 10^6$	$1.83 \cdot 10^6$
7	26.5	23.5	$1.82 \cdot 10^6$	$2.42 \cdot 10^6$	$1.93 \cdot 10^6$	$1.82 \cdot 10^6$	$2.44 \cdot 10^6$	$2.30 \cdot 10^6$	$2.89 \cdot 10^6$	$3.03 \cdot 10^6$
8	29.5	23.5	$3.03 \cdot 10^6$	$3.76 \cdot 10^6$	$3.50 \cdot 10^6$	$4.68 \cdot 10^6$	$1.86 \cdot 10^6$	$2.45 \cdot 10^6$	$\mathbf{1.35 \cdot 10^5}$	$-1.68 \cdot 10^1$
9	20.5	26.5	$7.43 \cdot 10^5$	$7.06 \cdot 10^5$	$3.01 \cdot 10^6$	$4.73 \cdot 10^6$	$5.74 \cdot 10^6$	$5.48 \cdot 10^6$	$1.90 \cdot 10^6$	$1.82 \cdot 10^6$
10	23.5	26.5	$2.49 \cdot 10^6$	$2.50 \cdot 10^6$	$4.34 \cdot 10^6$	$4.76 \cdot 10^6$	$3.88 \cdot 10^6$	$3.38 \cdot 10^6$	$2.36 \cdot 10^6$	$2.41 \cdot 10^6$
11	26.5	26.5	$2.96 \cdot 10^6$	$2.75 \cdot 10^6$	$3.22 \cdot 10^6$	$2.89 \cdot 10^6$	$1.82 \cdot 10^6$	$1.75 \cdot 10^6$	$1.84 \cdot 10^6$	$1.59 \cdot 10^6$
12	29.5	26.5	$3.62 \cdot 10^6$	$3.21 \cdot 10^6$	$4.64 \cdot 10^6$	$4.38 \cdot 10^6$	$5.89 \cdot 10^5$	$4.77 \cdot 10^5$	$\mathbf{5.99 \cdot 10^5}$	$-1.77 \cdot 10^1$
13	20.5	29.5	$\mathbf{9.01 \cdot 10^6}$	$4.15 \cdot 10^5$	$\mathbf{3.59 \cdot 10^5}$	$-4.91 \cdot 10^1$	$\mathbf{1.15 \cdot 10^6}$	$-7.32 \cdot 10^1$	$1.21 \cdot 10^6$	$2.41 \cdot 10^6$
14	23.5	29.5	$2.88 \cdot 10^6$	$2.80 \cdot 10^6$	$6.38 \cdot 10^6$	$6.31 \cdot 10^6$	$\mathbf{2.66 \cdot 10^6}$	$4.25 \cdot 10^6$	$2.00 \cdot 10^6$	$2.01 \cdot 10^6$
15	26.5	29.5	$4.36 \cdot 10^6$	$4.92 \cdot 10^6$	$4.32 \cdot 10^6$	$3.95 \cdot 10^6$	$\mathbf{1.47 \cdot 10^6}$	$2.30 \cdot 10^6$	$\mathbf{8.48 \cdot 10^5}$	$8.97 \cdot 10^5$
16	29.5	29.5	$5.60 \cdot 10^6$	$6.15 \cdot 10^6$	$6.59 \cdot 10^6$	$7.63 \cdot 10^6$	$\mathbf{5.96 \cdot 10^5}$	$1.01 \cdot 10^6$	$\mathbf{3.29 \cdot 10^5}$	$-1.80 \cdot 10^1$

Table 7.7: The measurements of m_1^c for all 4 setups are listed here for both cases, (I): DG(1) + diffusive L^2-projection on the coarse mesh, and (II): adaptive DG(1) solution which is regarded as the reference solution. Large deviations from the reference are highlighted in bold font.

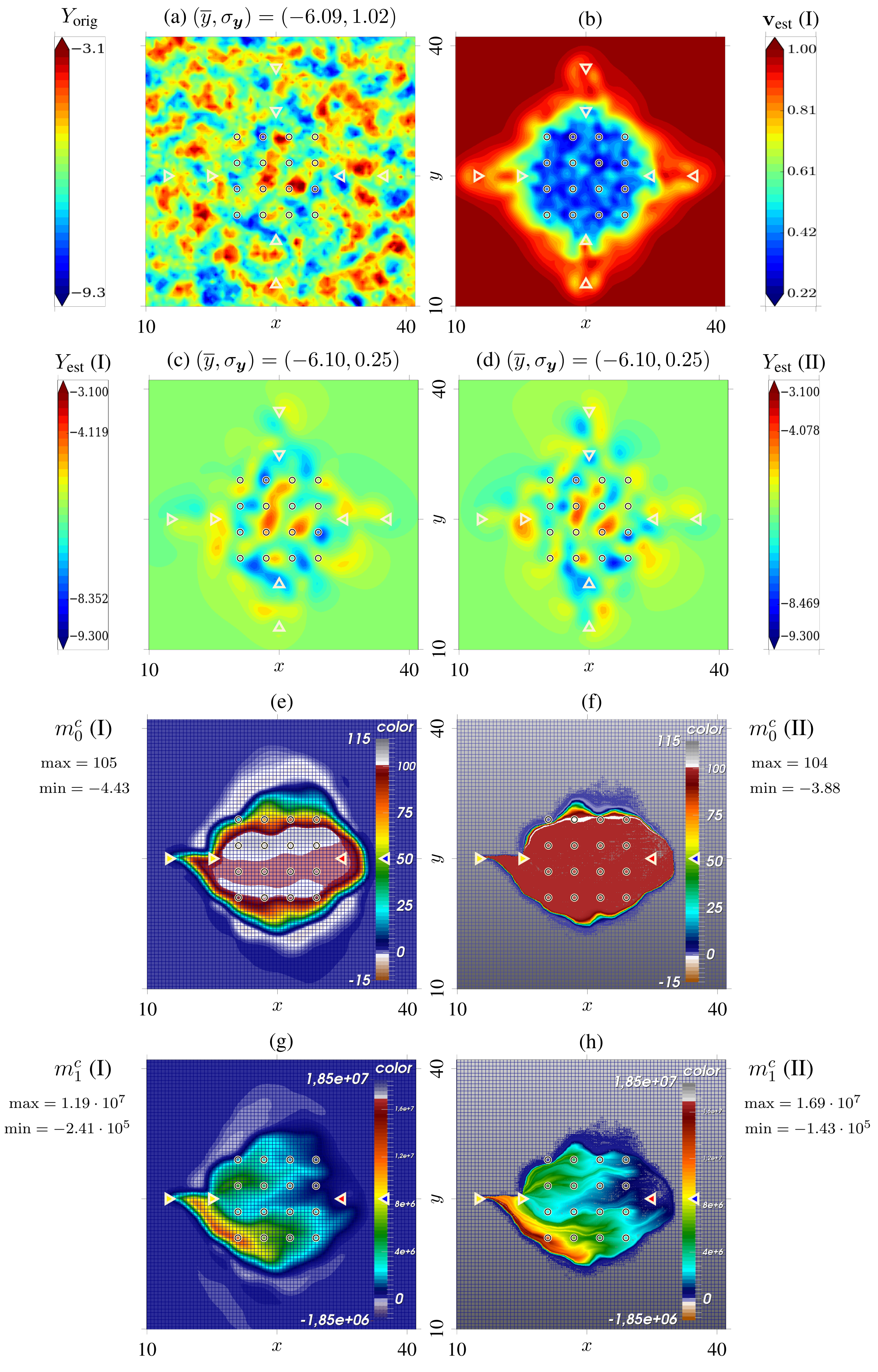


Figure 7.11: 2-D inversion: Investigating the influence of adaptive refinement on the final result of a parameter estimation process. The temporal moments displayed are taken from setup #3.

The two different sets of measurement data for m_1^c, each combined with the measurements for ϕ (identical in both cases), are then used to run a fully coupled inversion on the 'coarse' mesh $\mathcal{T}_{h_0}$ (`YASP` in parallel mode) where the combination CCFV / DG(1) + diffusive L^2-projection is used to solve the flow and transport equations.

	Case I			Case II		
GN step	$\mathscr{J}^d$	$\mathscr{J}^y$	$\mathscr{J}$	$\mathscr{J}^d$	$\mathscr{J}^y$	$\mathscr{J}$
0	$3.60 \cdot 10^4$		$3.60 \cdot 10^4$	$4.40 \cdot 10^4$		$4.40 \cdot 10^4$
1	$2.40 \cdot 10^4$	$5.69 \cdot 10^0$	$2.40 \cdot 10^4$	$3.35 \cdot 10^4$	$5.54 \cdot 10^0$	$3.35 \cdot 10^4$
2	$7.57 \cdot 10^3$	$1.24 \cdot 10^2$	$7.69 \cdot 10^3$	$1.80 \cdot 10^4$	$6.93 \cdot 10^1$	$1.81 \cdot 10^4$
3	$3.72 \cdot 10^3$	$5.30 \cdot 10^2$	$4.25 \cdot 10^3$	$1.09 \cdot 10^4$	$5.24 \cdot 10^2$	$1.14 \cdot 10^4$
4	$2.12 \cdot 10^3$	$6.48 \cdot 10^2$	$2.77 \cdot 10^3$	$7.81 \cdot 10^3$	$9.23 \cdot 10^2$	$8.73 \cdot 10^3$
5	$8.61 \cdot 10^2$	$2.52 \cdot 10^2$	$1.11 \cdot 10^3$	$5.63 \cdot 10^3$	$1.03 \cdot 10^3$	$6.66 \cdot 10^3$
6	$3.65 \cdot 10^2$	$2.01 \cdot 10^2$	$5.66 \cdot 10^2$			
7	$7.67 \cdot 10^1$	$1.23 \cdot 10^2$	$2.00 \cdot 10^2$			
8	$5.61 \cdot 10^1$	$1.07 \cdot 10^2$	$1.63 \cdot 10^2$			
9	$4.38 \cdot 10^1$	$1.01 \cdot 10^2$	$1.44 \cdot 10^2$			

Table 7.8: The 95-th percentile criterion (3.79) is fulfilled only in case I as soon as the objective function $\mathscr{J}$ falls below the threshold $\mathscr{J}^* := \frac{1}{2} \cdot \chi^2_{0.05}(385) = 215.88$. The computations run in parallel on 4 cores of `dnk`. Duration: 47 sec. per Gauss-Newton step.

Observations:

Fig.7.11(e)+(g) show the L^2-projected solution of m_0^c and m_1^c (case I), Fig.7.11(f)+(h) show the adaptive solution of m_0^c and m_1^c (case II) for Setup #3. The adaptive solutions may be regarded as the reference forward solutions for these transport problems.

As expected, the estimated field in Figure 7.11(c) shows a very good coincidence with the original, because the very same forward solvers used for the generation of the measurement data were used for the inversion scheme (case I).

The observations made for the case II supports the claim that the accuracy of adaptively computed solutions may not be necessary for the inversion procedure. The estimated field (Figure 7.11(d)) reveals that the main structures and also the variability of the original field still can be recovered within the measuring zone although the coarse grid solution may produce large deviations from the reference solution at single measuring points (Table 7.7).

Figure 7.11(b) shows the corresponding estimated variance (3.66). It is comparable to the estimated variance for the case I which is not depicted here.

Only the criterion (3.79) is not fulfilled. But as we will see in §7.6.6, increasing the value of the measurement error for m_1^c (which would have been reasonable here) can help reducing the value of the objective function.

In the remaining subsections, we solve 3-D problems on the structured `YASP` grid.

7.6.3 Scenario 2: FEM/SDFEM vs. CCFV/DG(1) as forward solvers

The domain Ω is discretized using a structured mesh $\mathcal{T}_{h_1}$ where the cell numbers are given by $(\ n_x,\ n_y,\ n_z\ ) = (\ 512,\ 512,\ 100\ )$. Each cell has the constant lengths $(\Delta x, \Delta y, \Delta z) = (\ 0.10,\ 0.10,\ 0.05\ )[m]$.

Setup	#3		#1	
Solving for ϕ :				
numerical scheme	FEM	CCFV	FEM	CCFV
# parallel unknowns	28,800,756	27,667,600	28,800,756	27,667,600
matrix assembly time $[s]$	9.135	7.044	9.15	5.31
linear solver time $[s]$	15.80	6.80	15.79	7.05
# iterations	23	23	23	23
time per iteration	0.68	0.295	0.68	0.31
linear solver/pre-conditioner	CG with AMG, reduction 10^{-10}		CG with AMG, reduction 10^{-10}	
Solving for m_0^c with effective mesh Péclet number $\mathcal{P}_h^t \approx 490$:				
numerical scheme	SDFEM	DG(1)	SDFEM	DG(1)
# parallel unknowns	28,800,756	221,340,800	28,800,756	221,340,800
matrix assembly time $[s]$	19.85	81.15	19.98	82.91
linear solver time $[s]$	32.12	195.0	---	81.56
# iterations	98	38	---	23
time per iteration	0.32	5.14	---	3.55
linear solver/pre-conditioner	GMRES(100) with ILU(0), red. 10^{-7}		GMRES(100) with ILU(0), red. 10^{-8}	

Table 7.9: 3-D nested cells environment, comparing forward solvers: Parallel computations on `dnk` (see Table F.2) using $P = 64$ cores on four nodes with 1 task per core and 2 tasks per memory channel.

Observations:

1. CCFV requires less time than FEM for the stiffness matrix assembly.
2. The linear system for CCFV can be solved more than twice as fast as the linear system for FEM.
3. The SDFEM linear system can be solved only for the Setup #3. In all other cases, the linear solver does not converge even after 5000 iterations.

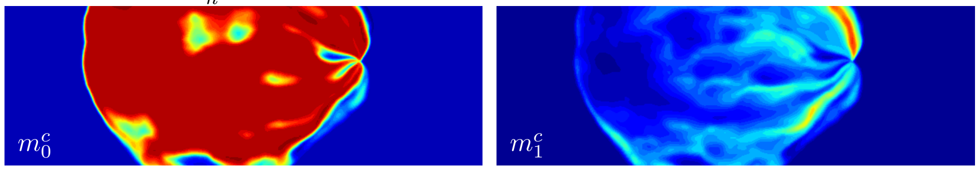
This makes the combination CCFV / DG(1) postprocessed with the diffusive L^2-projection our **method of choice** for the forward solution of stationary transport problems with high mesh Péclet numbers. However, for this high-resolution mesh, DG(1) scheme requires more than 200 million unknowns. The memory consumption of the linear solver reaches about 96% of the RAM that are available on the computing nodes.

We want to confirm that our implementation of the SDFEM method can be applied to less convection-dominant problems in the same setting. Therefore, we set $\alpha_\ell = 10^{-1}$ and successively increase the amount of $\alpha_t \in \{10^{-4}, 10^{-3}, 10^{-2}\}$. For $\alpha_t = 10^{-2}$ (when $\mathcal{P}_h^t \approx 4.9$), this may be comparable to the solution of the more diffusive heat transport equation. Only in this case, the SDFEM linear system can be solved by GMRES(100) with ILU(0) for all four setups. Over- and undershoots are less than 0.5%. Table 7.10 shows that the linear solution of the DG system takes much longer than in the convection-dominant case. A further test comparing preconditioners reveals that for such a linear system, GMRES(100) with ILU(0) is not only faster, but also more robust than BiCGSTAB with AMG.

Setup	#3			#1		
Solving for m_1^c with effective mesh Péclet number $\mathcal{P}_h^t \approx 4.9$:						
numerical scheme	SDFEM	SDFEM	DG(1)	SDFEM	SDFEM	DG(1)
# parallel unknowns	28,800,756	28,800,756	221,340,800	28,800,756	28,800,756	221,340,800
matrix assembly time $[s]$	13.09	13.17	32.91	13.23	19.94	33.11
linear solver time $[s]$	38.84	26.87	1697.32	23.38	---	1394.72
# iterations	135	34	594	82	---	420
time per iteration	0.29	0.79	2.86	0.29	---	3.32
linear solver	GMRES(100) with reduction 10^{-7}			GMRES(100) with reduction 10^{-7}		
pre-conditioner	ILU(0)	AMG	ILU(0)	ILU(0)	AMG	ILU(0)

Table 7.10: 3-D nested cells environment: Transport simulation with low mesh Péclet numbers, comparing SDFEM vs. DG(1) and ILU(0) vs. AMG. $P = 64$ cores as above.

DG(1)+L^2 solutions for $\mathcal{P}_h^t \approx 490$:



SDFEM solutions for $\mathcal{P}_h^t \approx 4.90$:

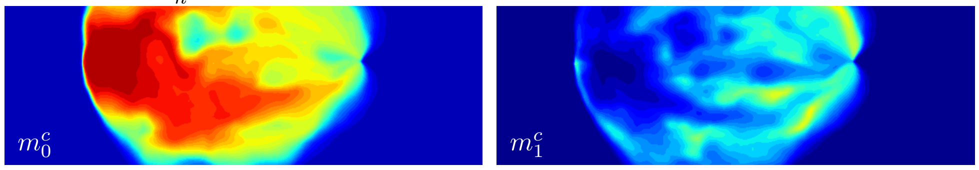


Figure 7.12: Quality of different m_0^c and m_1^c solutions for different Péclet numbers (Setup #1): Plots for the same quantities use the same color code. 1st *row:* high Péclet number solutions with sharp internal layers (steep gradients). 2nd *row:* low Péclet number solutions with smeared layers (smooth gradients). This reflects the fact that experiments with a solute tracer deliver more accurate measurements than a heat tracer.

7.6.4 Scenario 3: Parallel scalability tests for the forward solvers

Strong scaling

For a fixed mesh resolution, the coupled forward problems (ϕ, m_o^c) for Setup $\#1$ of Table 7.6 are solved on one and the same log conductivity field using an increasing number of cores. Due to a finer resolution and a lower number of cells in the z-direction, the parallel partitioning $P = P_x \times P_y \times 1$ of the domain is restricted to the (x, y)-plane.

	Solving (4.1) for ϕ			Solving (4.21) for m_0^c				Diffusive L^2-projection (5.54)		
	CCFV($\mathbb{P}_0$) / CG + AMG			DG($\mathbb{Q}_1$) / BiCGSTAB + ILU(0)				FEM($\mathbb{Q}_1$) / CG + AMG		
	$M[s]$	$\#IT$	$TIT[s]$	$T_{\text{sort}}[s]$	$M[s]$	$\#IT$	$TIT[s]$	$M[s]$	$\#IT$	$TIT[s]$
$\boldsymbol{P} = \mathbf{1}$ *sequential task*										
	10.073	24	1.0365	3.817	585.09	7	10.77	36.485	4	2.431
$\boldsymbol{P} = \mathbf{4} = 2 \times 2$ *parallel tasks on* 2 *nodes (1 task per core,* 1 *task per memory channel)*										
	2.867	23	0.246	0.795	159.284	9	2.709	9.906	6	0.598
S	3.51		4.21	4.80	3.67		3.98	3.68		4.07
$\boldsymbol{E}$	0.88		**1.05**	1.20	0.92		**0.99**	0.92		**1.02**
$\boldsymbol{P} = \mathbf{16} = 4 \times 4$ *parallel tasks on* 4 *nodes (1 task per core,* 1 *task per memory channel)*										
	0.758	23	0.0658	0.129	41.315	10	0.699	2.515	6	0.172
S	13.29		15.75	29.56	14.16		15.41	14.51		14.13
$\boldsymbol{E}$	0.83		**0.98**	1.85	0.89		**0.96**	0.91		**0.88**
$\boldsymbol{P} = \mathbf{32} = 4 \times 8$ *parallel tasks on* 4 *nodes (1 task per core,* 1 *task per memory channel)*										
	0.388	22	0.0385	0.061	22.303	12	0.412	1.294	6	0.104
S	25.96		26.92	62.57	26.23		26.14	28.20		23.38
$\boldsymbol{E}$	0.81		**0.84**	1.95	0.82		**0.82**	0.88		**0.73**
$\boldsymbol{P} = \mathbf{64} = 8 \times 8$ *parallel tasks on* 4 *nodes (1 task per core,* **2** *tasks per memory channel)*										
	0.215	24	0.0308	0.029	11.18	13	0.31	0.671	6	0.089
S	46.85		33.65	131.62	52.33		34.74	54.37		27.31
$\boldsymbol{E}$	0.73		**0.53**	2.06	0.82		**0.54**	0.85		**0.43**

Table 7.11: 3-D nested cells environment, strong scalability test for CCFV/DG(1) with a fixed number of $256 \times 256 \times 50 = 3,276,800$ elements. *Symbols:* T_{sort} = *time for renumbering the mesh elements,* M = *matrix assembly time,* $\#IT$ = *number of iterations,* TIT = *time per iteration of the linear solver,* S = *speed-up and* E = *parallel efficiency. All computations run on* `dnk` *(see Table F.2).*

Weak scaling

For an increasing number of cores, the mesh resolution is increased such that the number of unknowns per task remains at the same level. (ϕ, m_o^c) for Setup #1 of Table 7.6 are computed on different Y-fields generated from the same geostatistical parameters.

	Solving (4.1) for ϕ				Solving (4.21) for m_0^c			Diffusive L^2-projection (5.54)		
	CCFV($\mathbb{Q}_1$) / CG + AMG				DG($\mathbb{Q}_1$) / BCGS + ILU(0)			FEM($\mathbb{Q}_1$) / BCGS + AMG		
	$M[s]$	$\#IT$	$TIT[s]$	$T_{\text{sort}}[s]$	$M[s]$	$\#IT$	$TIT[s]$	$M[s]$	$\#IT$	$TIT[s]$
$N = 102,400$ / $\boldsymbol{P = 1}$ *sequential task*										
	DOF $= 102,400$				DOF $= 819,200$			DOF $= 111,537$		
	0.312	17	0.0267	0.0548	18.079	3	0.279	1.084	2	0.129
$N = 409,600$ / $\boldsymbol{P = 4}$ *parallel tasks on* 2 *nodes* (1 *task per core,* 1 *task per memory channel*)										
	DOF $= 422,500$				DOF $= 3,380,000$			DOF $= 453,024$		
	0.365	19	0.0282	0.0548	20.064	7	0.291	1.213	3	0.1449
$\boldsymbol{E}$	0.85		**0.95**	1.0	0.90		**0.96**	0.89		**0.89**
$N = 1,648,360$ / $\boldsymbol{P = 16}$ *parallel tasks on* 4 *nodes* (1 *task per core,* 1 *task per memory channel*)										
	DOF $= 1,747,240$				DOF $= 13,977,920$			DOF $= 1,860,129$		
	0.383	23	0.0326	0.0547	21.202	8	0.357	1.296	3	0.158
$\boldsymbol{E}$	0.81		**0.82**	1.0	0.85		**0.78**	0.84		**0.82**
$N = 3,276,800$ / $\boldsymbol{P = 32}$ *parallel tasks on* 4 *nodes* (1 *task per core,* 1 *task per memory channel*)										
	DOF $= 3,537,000$				DOF $= 28,296,000$			DOF $= 3,771,348$		
	0.388	22	0.0385	0.0629	22.303	12	0.471	1.307	3	0.209
$\boldsymbol{E}$	0.80		**0.69**	0.871	0.81		**0.59**	0.83		**0.62**
$N = 6,532,092$ / $\boldsymbol{P = 64}$ *parallel tasks on* 4 *nodes* (1 *task per core,* **2** *tasks per memory channel*)										
	DOF $= 7,112,448$				DOF $= 56,899,584$			DOF $= 7,573,504$		
	0.41	25	0.0648	0.0943	22.009	14	0.739	1.331	3	0.359
$\boldsymbol{E}$	0.76		**0.41**	0.581	0.82		**0.38**	0.81		**0.36**
DOF	$\approx 1.06 \cdot 10^5$ per task				$\approx 8.5 \cdot 10^5$ per task			$\approx 1.15 \cdot 10^5$ per task		

Table 7.12: 3-D nested cells environment, weak scalability test for CCFV/DG(1) with a per-task-average of $\approx 10^5$ mesh cells: *Symbols:* N *= number of cells,* T_{sort} *= time for renumbering the mesh elements,* M *= matrix assembly time,* $\#IT$ *= number of iterations,* TIT *= time per iteration of the linear solver and* E *= parallel efficiency. All computations run on* `dnk` *(see Table F.2).*

Observations

Both scalability tests (which start with $P = 1$) show that the matrix assembly for all numerical schemes and the renumbering of mesh cells scale well. The efficiency of the linear solvers for the different schemes remains at a high level as long as the full capacities of the memory channels can be exploited. It goes down as soon as more than one task have to share one channel. This is due to the fact that iterative schemes are based on matrix$\times$vector multiplications where a high throughput is required. In other tests on the cluster `hpc4` where a higher number of cores is available, the efficiency of the linear solver for the DG discretization drops for another reason: The cell-wise renumbering of the unknowns following the hydraulic head play a pre-conditioning role for the linear solver. For an ideal parallel scalability, the shape of the partitions of the parallel mesh would have to be constructed and ordered according to the hydraulic head. However, such an extension of the capabilities of the mesh is beyond the scope of this work. For our purposes, working with a relatively small number of cores ($P \leq 64$) and using the straight partition blocks of the structured mesh (`YASP` grid) is sufficient.

7.6.5 Scenario 4: Fully-coupled inversion with different strategies

The Y-field generated for the simulations in §7.6.3 on the fine mesh $\mathcal{T}_{h_1}$ is taken as the original parameter field on which measurements of ϕ and m_1^c are taken for the four nested cells setups presented in Table 7.6. Since the ratio between correlation length and meshsize is $10 : 1$, we may try to run the inversion on a coarse mesh $\mathcal{T}_{h_0}$ with double meshsize where this ratio is $5 : 1$. The combinations of numerical schemes and linear solvers verified in §7.6.4 (for the parallel solution of the steady-state groundwater flow and convection-dominant transport problems) are now employed to the framework of geostatistical inversion. We apply combinations of two types of inversions, namely

type 1: based on simulated measurements of ϕ,

type 2: based on simulated measurements of ϕ and m_1^c,

to devise different strategies

St1: inversion of type 1 on the mesh $\mathcal{T}_{h_0}$ initialized with $\boldsymbol{y}_0 = \mathcal{X}\beta^*$,

St2: inversion of type 2 on the mesh $\mathcal{T}_{h_0}$ initialized with the computed estimate of St1,

St3: inversion of type 2 on the mesh $\mathcal{T}_{h_0}$ initialized with $\boldsymbol{y}_0 = \mathcal{X}\beta^*$,

St4: inversion of type 2 on the mesh $\mathcal{T}_{h_1}$ initialized with the computed estimate of St2, which must be projected onto the fine mesh. One cell of $\mathcal{T}_{h_0}$ contains eight refined cells of $\mathcal{T}_{h_1}$. On these refined cells, the initial estimate for St4 assume the same value given by the parent cell in $\mathcal{T}_{h_0}$.

In the end, we simulate the forward problems for the final estimate on the fine mesh.

The absolute error $\epsilon_\ell^{(\text{abs})}[m_1^c]$ in (3.21) can be used to emphasize the importance (or weight) of a measurement at a certain location $\vec{x}_\ell$. The practitioner working in the field has to provide appropriate values. In our numerical experiments, we can vary its values within a wide range to see its effect. We either choose the same fixed value for all setups or, for each setup S, we define it by

$$\epsilon_\ell^{(\text{abs})}[m_1^c] := p_\epsilon \times \max_{d_\ell^{(\text{obs})} \in M_1^S} \left\{ d_\ell^{(\text{obs})} \right\} \tag{7.10}$$

where M_1^S is the list of all original m_1^c-measurements of the setup S. In the test cases (A)-(D) listed in Table 7.14, we choose $p_\epsilon = 5\%$ and obtain four different values for $\epsilon_\ell^{(\text{abs})}$ between $2.95 \cdot 10^5$ and $5.7 \cdot 10^5$.

The workload in all computations is distributed among $P = 64$ parallel tasks, using

- either 4 nodes of `hpc4` with 16 cores per node (1 task per memory channel)
- or 4 nodes of `dnk` with 8 cores per node (2 tasks per memory channel).

mesh	cell numbers	simulated measurements	S_ϕ	S_1	`hpc4`	`dnk`
$\mathcal{T}_{h_0}$	$256 \times 256 \times 50$	ϕ	256	0	90 min.	30 min.
$\mathcal{T}_{h_0}$	$256 \times 256 \times 50$	ϕ and m_1^c	256	256	8 hours	2 hours
$\mathcal{T}_{h_1}$	$512 \times 512 \times 100$	ϕ and m_1^c	256	256	>3 days	22 hours

Table 7.13: 3-D inversion: Average runtime for a single step of the Gauss-Newton iteration scheme, comprising the computation of all involved sensitivity and cross-covariance fields and the evaluation of the cost function for trial estimates in the line search algorithm. $S_{\{\phi,1\}}$ = number of sensitivity fields for $\{\phi, m_1^c\}$.

test case	strategy (required #steps)	`hpc4`	`dnk`	$\overline{y}$	σ_y	$\mathcal{J}$ (3.28)
(A)	St1(6)	9 hours	3 hours	-5.22	0.07	37.85 (ϕ only)
(B)	St1(6) + St2(5)	49 hours	13 hours	-5.23	0.112	137.0 (ϕ, m_1^c)
(C)	St3(7)	56 hours	14 hours	-5.23	0.114	145.0 (ϕ, m_1^c)
post-processing the coarse solution on the fine mesh:						
(D)	St4(+1)	---	+22 hours	-5.23	0.108	108.75 (ϕ, m_1^c)

Table 7.14: Total inversion runtimes for different strategies on the coarse mesh. $\overline{y}$ is the arithmetic mean and σ_y is the standard deviation taken from the visible part of the Y-field shown in Figure 7.13. For the original Y-field, $\overline{y} = -5.44$ and $\sigma_y = 1.0$. The means $\overline{y}$ of the estimated fields are all within the $2\sigma_\beta$-interval about the given trend $\beta = -6.0$ (see Table 7.6).

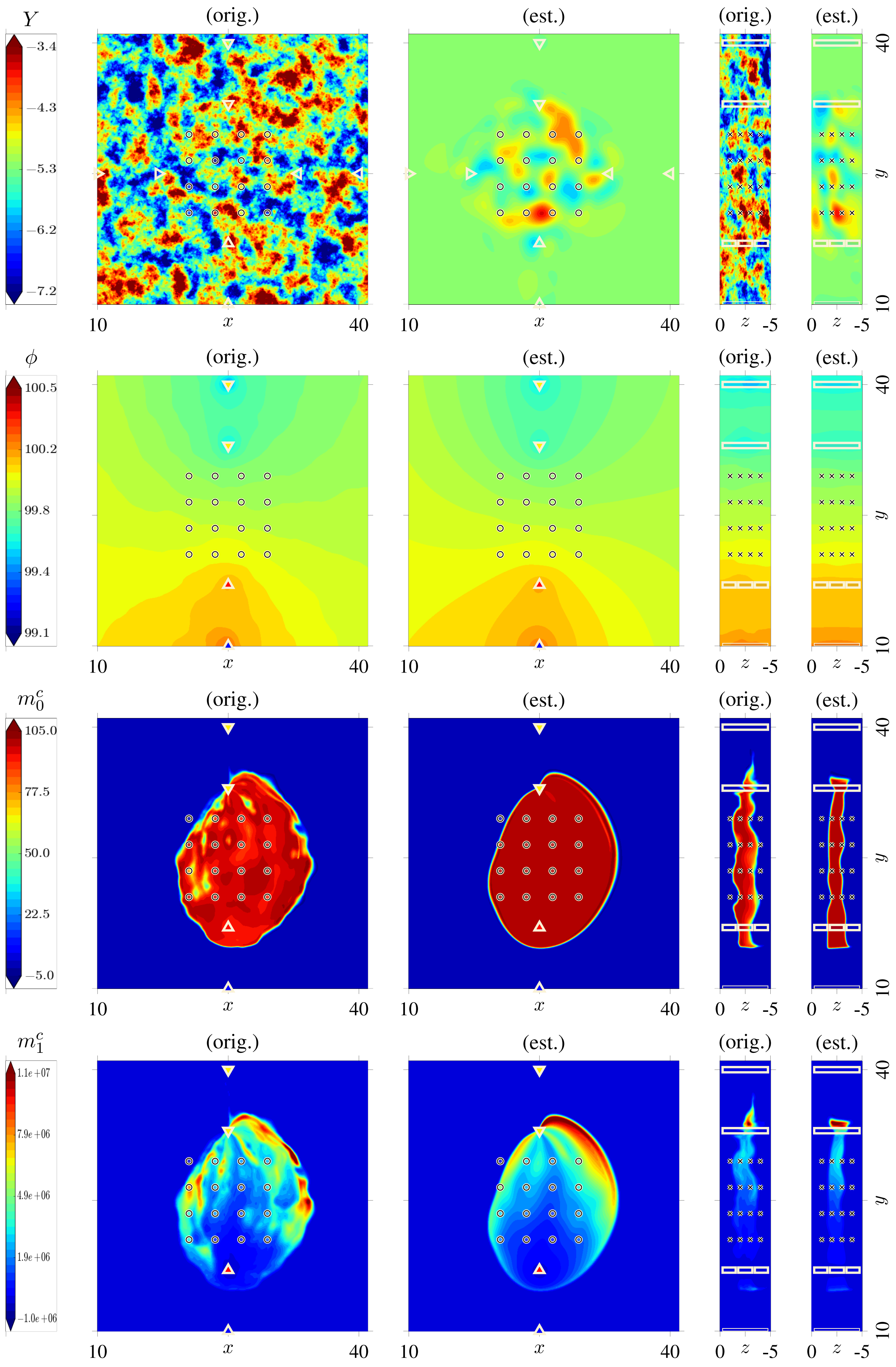


Figure 7.13: 3-D inversion: Comparing original with estimated fields (flow fields for setup #2)

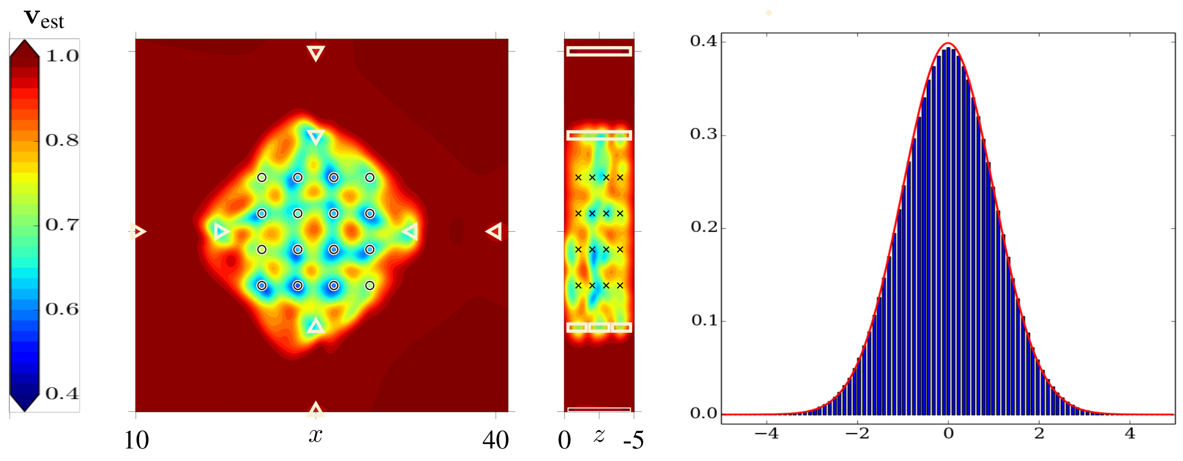


Figure 7.14: 3-D Inversion: The estimated variance $\mathbf{v}_{\text{est}}$ (3.66) has its local minima at measurement locations. Its minimum is 0.41. The histogram of the weighted error (3.67) matches very well with the standard normal distribution (red line) confirming that the estimated solution is unbiased.

Observations:

Figures 7.13 & 7.14 show the results of a combined inversion (B)+(D) taking less than 40 hours in total on the machine `dnk`. Running the inversion on the coarse mesh $\mathcal{T}_{h_0}$, where the correlation lengths and thus the geostatistical structures are still resolved well, the computing time can be reduced dramatically. The total runtime for a complete inversion on $\mathcal{T}_{h_0}$ is in the same range as the runtime of one single inversion step on $\mathcal{T}_{h_1}$.

The two test cases (B) and (C) in Table 7.14 yield almost the same results. Both solutions can be projected onto the fine mesh and reused as the initial solution for one or two final inversion steps on the fine mesh. However, this extra step on the fine mesh does not add a worthwhile improvement. The estimated parameter field on $\mathcal{T}_{h_0}$ is qualitatively (in terms of structure and recovered variances) not much different from the estimated Y-field on $\mathcal{T}_{h_1}$ shown in the first row of Figure 7.13.

7.6.6 Scenario 5: Varying the input data

We conduct three further test cases (E), (F) and (G) which are copied versions of the test case (C), slightly modified in the following manner:

(E): Same as test case (C), but $p_\epsilon = 0.5\%$. Thus, $\epsilon_\ell^{(\text{abs})}[m_1^c]$ decreases by a factor of 10, putting more weight on the m_1^c measurements.

(F): Same as test case (C), but we set directly a very low value $\epsilon_\ell^{(\text{abs})}[m_1^c] = 50$, overweighting the m_1^c measurements.

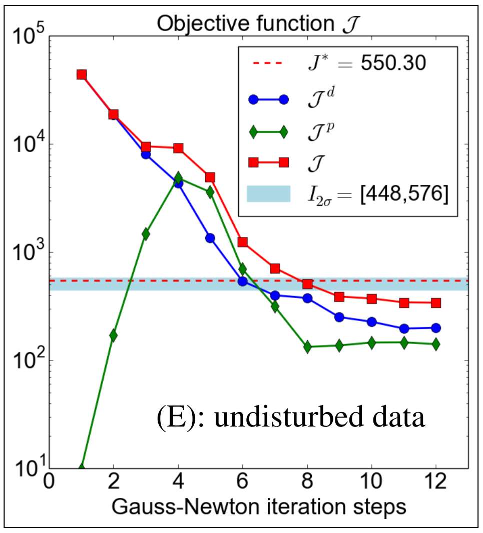
(G): Same as test case (E), but

- we work under the false assumption that the correlation lengths are double as large as listed in Table 7.6,
- the trend coefficient is $\beta = -6.2$ instead of -6.0,
- and we add a normally distributed disturbance $\delta d_\ell \sim \mathcal{N}(0, \varepsilon_\ell^2)$ to each of the measured values $d_\ell^{(\text{obs})}$ where $\varepsilon_\ell = \varepsilon_\ell^m = 5\% \cdot |d_\ell^{(\text{obs})}|$ in the case of m_1^c-measurements, and $\varepsilon_\ell = \varepsilon_\ell^\phi = 5\% \cdot |\max_k d_k^{(\text{obs})} - \min_k d_k^{(\text{obs})}|$ in the case of head measurements. Some examples are listed in the table of Figure 7.16.

The results are shown in Table 7.15 and Figures 7.15 – 7.17.

	original field	estimated field			
test case		(C)	(E)	(F)	(G)
data and prior knowledge		unchanged	unchanged	unchanged	disturbed
measurement error $\epsilon_\ell^{(\text{abs})}[m_1^c]$		$\sim O(10^5)$	$\sim O(10^4)$	$= 50$	$\sim O(10^4)$
number of St3-steps		7	12	14	8
$\max_j\{(\boldsymbol{y})_j\}$	-0.41	-3.66	-2.91	-2.78	-3.32
$\min_j\{(\boldsymbol{y})_j\}$	-10.73	-7.03	-7.72	-8.36	-8.36
$\overline{y}$	-5.43	-5.23	-5.24	-5.24	-5.23
$\sigma_{\boldsymbol{y}}$	1.0	0.114	0.146	0.180	0.160
$\mathscr{J}$ (3.28)		145.0	339.7	$1.3 \cdot 10^6$	380.0
$\min_j\{(\mathbf{v}_{\text{est}})_j\}$		0.41	0.22	0.11	0.23

Table 7.15: Scenario 4: Decreasing measurement error and disturbing prior knowledge and data. $\overline{y}$ is the arithmetic mean and $\sigma_{\boldsymbol{y}}$ is the standard deviation taken from the visible part of the Y-field shown in Figure 7.17.



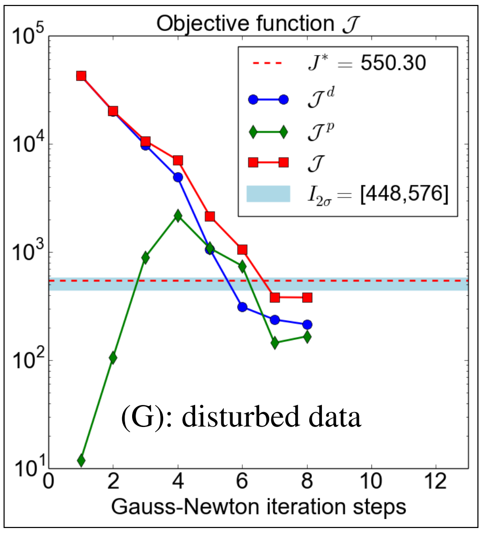


Figure 7.15: Scenario 4: Convergence behavior of the inversion scheme for the test cases (E) and (G): The 95-th percentile criterion (3.79) is fulfilled as soon as the objective function $\mathscr{J}$ falls below the threshold $\mathscr{J}^* := \frac{1}{2} \cdot \chi^2_{0.05}(1025)$. The interval $I_{2\sigma} := [\mu - 2\sigma, \mu + 2\sigma]$ prescribed by the three-sigma rule (3.76) offers only a narrow range for the optimal value of $\mathscr{J}$ to hit.

An excerpt from the measurements:

	(Setup #2)	data	disturbed
ℓ	$\vec{x}_\ell$	$\phi(\vec{x}_\ell)$	$\phi(\vec{x}_\ell) + \varepsilon^\phi_\ell$
1	(20.5, 20.5, 1.0)	99.8576	99.8587
22	(23.5, 23.5, 2.0)	99.8821	99.8824
43	(26.5, 26.5, 3.0)	99.9271	99.9277
64	(29.5, 29.5, 4.0)	99.9653	99.9653
ℓ	$\vec{x}_\ell$	$m^c_1(\vec{x}_\ell)$	$m^c_1(\vec{x}_\ell) + \varepsilon^m_\ell$
1	(20.5, 20.5, 1.0)	191.5	187.3
22	(23.5, 23.5, 2.0)	$1.75\cdot10^6$	$1.69\cdot10^6$
43	(26.5, 26.5, 3.0)	$1.31\cdot10^6$	$1.36\cdot10^6$
64	(29.5, 29.5, 4.0)	564.2	604.6

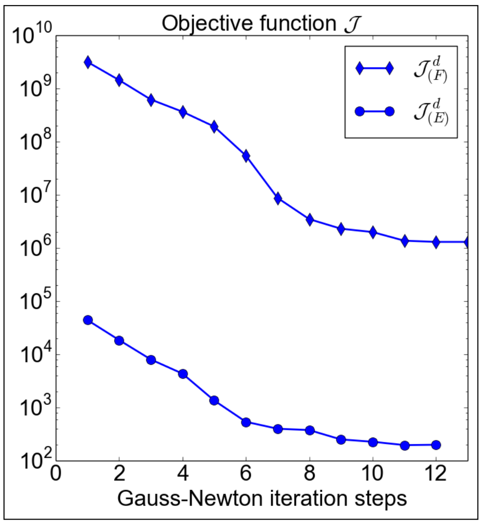


Figure 7.16: Scenario 4: Convergence behavior of the inversion scheme for the test cases (E) and (F): The $\mathscr{J}^d$-curve for the test case (F) has a similar shape as the $\mathscr{J}^d$-curve for the test case (E), only shifted upwards.

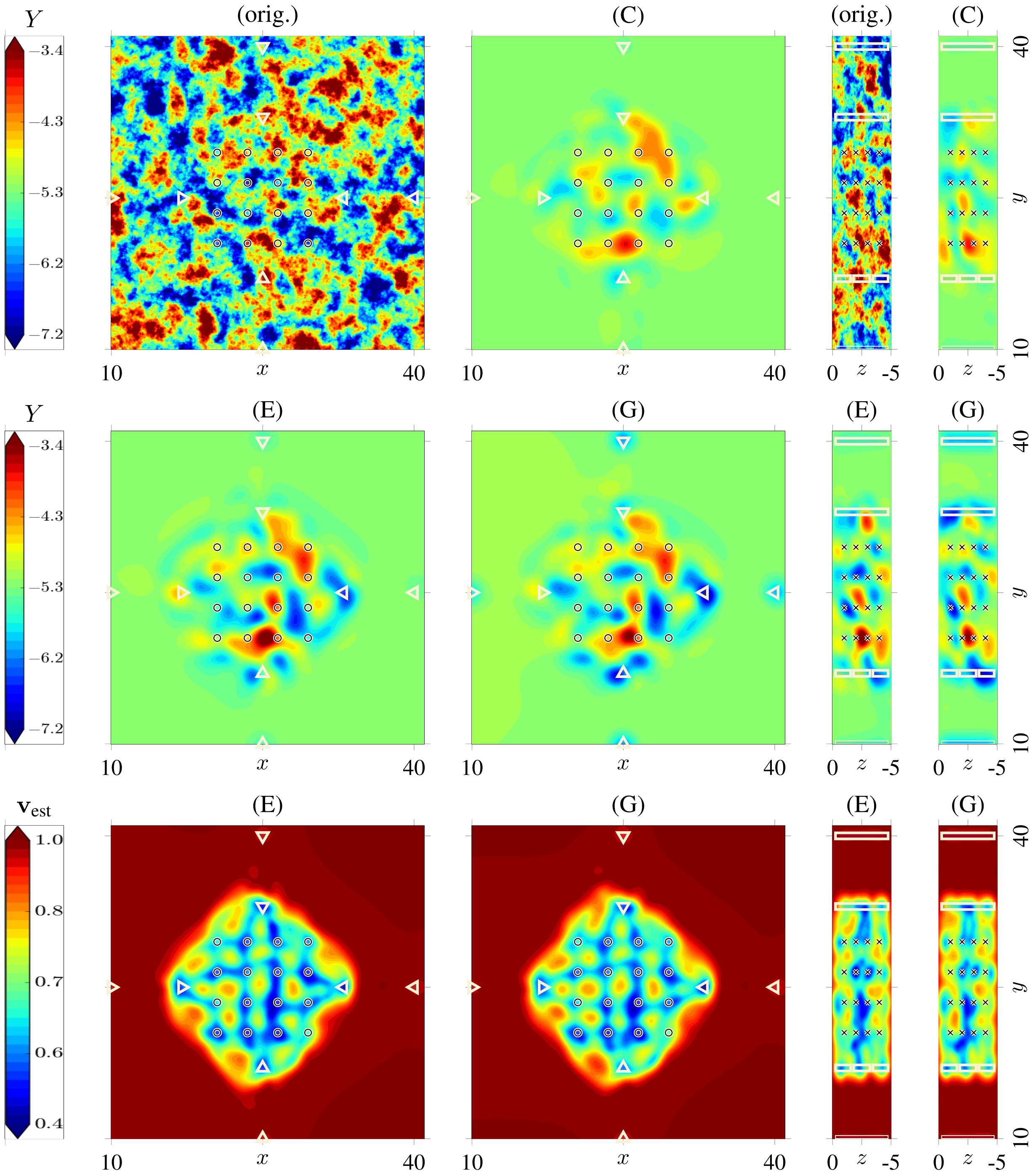


Figure 7.17: Scenario 4: Comparing the results of the test cases (C), (E) and (G)

Observations:

(C)/(E)/(F) reveal the effect that the choice of the measurement error has on the convergence behavior and the quality of the solution. The smaller the measurement error, the larger the value of the objective function. A higher number of possible iterations helps achieving a higher recovery of the spatial variability. The best recovery of $\sigma_{\boldsymbol{y}}$ is achieved in (F) although the objective function is far above the threshold (Figure 7.16). The thresholds Δ_J and $\Delta_{\boldsymbol{y}}$ in the convergence criterion of the inversion algorithm 3.1 may be adjusted to the magnitude of the measurement error $\epsilon_\ell^{(\text{abs})}[m_1^c]$ in order to achieve comparable convergence behavior.

(G) demonstrates experimentally the stability of the inversion scheme for small disturbances of the data. The results agree well with those from (E) as Figures 7.15 and 7.17. This is an indicator for the regularized inversion problem (3.28) to be well-conditioned and to fulfill the continuity property for well-posedness.

7.6.7 Overview of time consumption

The heterogeneous landscape of tasks required for the completion of a whole inversion in parallel calls for a timer measuring the time spent on the different types of tasks. Table 7.16 confirms that most of the time is typically spent on the solution of the occurring BVPs, where the main burden is borne by `DUNE`. Moreover, the efficiency of the FFTW and HDF5 implementations is verified. The extra time ($\approx 16\%$) spent on the evaluation and redistribution of data summarizes the time consumption of other tasks such as the simulated measurements, or the computation of the estimation variance at the end of the inversion procedure.

Task type:	Wall time	Fraction
I/O:	3,537.47 sec.	4.53 %
BVPs:	60,225.28 sec.	77.12 %
FFTW:	1,961.19 sec.	2.51 %
Eval.:	12,315.65 sec.	15.77 %
Redist.:	53.51 sec.	0.07 %
Sum:	78,093.10 sec.	100.00 %

Table 7.16: Overview of time consumption for the test case (E) that run on `dnk` (see Table F.1), using 64 cores on 4 nodes. The program took $81,604.20$ seconds in total. The time measured in detail covers 95.7%.

Chapter 8

Conclusion and Outlook

Conclusion

We have shown that the two-point flux cell-centered finite volume (CCFV) method combined with Discontinuous Galerkin (DG) methods for solving the steady-state groundwater flow and solute transport problems can be applied to the framework of the quasi-linear geostatistical approach for the 3-D estimation of the hydraulic conductivity in confined aquifers.

Two main issues occurring in the solution of the convection-dominated transport equation were tackled:

1. the efficient reduction of numerical under- and overshoots,
2. the efficient solution of the arising linear systems.

We have compared DG methods with the Streamline Diffusion (SDFEM) method. Putting special emphasis on a practical application, we have analyzed efficiency and accuracy. The observations made in the example with a jump in the boundary in §7.3 and in the solution of the forward problems in §7.4 and §7.5 clearly speak in favor of the post-processed DG methods if we consider computing time to be the ultimate measure of available hardware resources:

- On the same mesh level, the DG solutions resolve the steep fronts more sharply than the SDFEM solution. In order to obtain the same level of accuracy as DG(1), SDFEM would have to work on a globally refined mesh. As a consequence, the SDFEM approach would take longer than DG(1).
- In heterogeneous fields, the layers of spurious oscillations generated by SDFEM may spread into the surrounding domain whereas they stay localized for the DG method.

- The diffusive L^2-projection is able to reduce the over- and undershoots without increasing smearing effects beyond the mesh size.

- There exist cases of real world solute transport problems which can be solved by the DG(1) approach, but not by the SDFEM approach (§7.6.3).

- For less convection-dominant problems (heat transport), the SDFEM approach is more efficient than the DG method.

- On clusters with up to 100 cores, the presented DG based PDE solvers scale well as long as the full capacity of memory bandwidth can be exploited (§7.6.4).

Hence, for the solution of a convection-dominated steady-state transport problem, DG(1) post-processed by a diffusive L^2-projection offers an efficient and more accurate alternative to the well-known SDFEM method.

Adaptive mesh refinement yields the best possible solution with respect to the L^2 error, but is connected with extremely high computational costs. Using the h-adaptive DG(1) method, the numerical under- and overshoots can be reduced below a tolerated threshold of 5% only if the mesh cells at internal layers become so small that their local mesh Péclet numbers tend to a diffusion-dominant level. In 2-D, the whole adaptive loop can be finished within a time frame of a few minutes using a sequential code. In 3-D however, the problem size becomes so large that a very lengthy computation with a huge memory consumption is unavoidable. In the presented Gauss-Newton scheme in which a high number of such problems need to be solved repeatedly, adaptive mesh refinement would form a performance bottleneck. Finite volume schemes, on the other extreme, are efficient and monotone, but they cannot resolve steep fronts or point sources well enough for our purposes. The proposed post-processed DG method serves as a reliable and efficient compromise between the two extremes. The two test scenarios 7.6.2 and 7.6.6 with disturbed measurements have shown that the inversion scheme does not necessarily require a "perfect" solution of the transport problem.

Therefore, regarding the integration of the forward solvers into the inversion framework, we recommend the combination CCFV / DG(1) post-processed by a diffusive L^2-projection for the solution of steady-state transport problems with high mesh Péclet numbers. This combination works on the same structured mesh on which the hydraulic conductivity is resolved. This avoids an additional level of complexity in the implementation of the inversion scheme. Regarding the corresponding linear solver / preconditioner combinations, we recommend

- CG / AMG for the CCFV discretization and for the diffusive L^2-projection and

- GMRES / ILU(0) for the DG(1) discretization on a mesh with a downstream numbering of cells.

For transport problems with low mesh Péclet numbers, the classical FEM / SDFEM approach (with CG / AMG for FEM and GMRES / ILU(0) for SDFEM) would be the better choice.

A fully coupled DG-based 3-D parallel inversion for the setting presented in §7.6 can be accomplished within less than two days on a cluster affordable for an academic workgroup. Thereby, the mesh size for the inversion process should be chosen in such a way that

- the DG stiffness matrix easily fits the available memory,
- the matrix assembly and linear solution times stay short
- and the correlation lengths of the parameter field are still well resolved.

Of course, increasing the mesh resolution or the polynomial order of the DG basis generally improves the quality of the solution, but is paid by a longer computing time and a higher memory consumption.

The inversion code yields an estimation of the parameter field that essentially preserves its form even after disturbing the measurement data and the prior information to a large degree (§7.6.6). This verifies experimentally the well-posedness of the regularized inverse problem and the robustness of the inversion scheme.

A note on gradient based methods

The Gauss-Newton method described in this work requires the computation of the full sensitivity matrix which is the most time consuming step in the inversion scheme. The main advantage of the Gauss-Newton method in comparison to gradient-based methods is its fast (quadratic) convergence for initial estimates near the solution. Gradient-based methods usually require more iteration steps, but in each iteration step, the number of adjoint equations to be solved for each quantity is independent on the number of measuring locations. To see this, let us consider the steepest-descent method as an example of a gradient-based method for the minimization of the cost function $\mathscr{J}(\boldsymbol{p}_k)$ (3.28). Starting with an initial guess $\boldsymbol{p}_0$, a sequence $(\boldsymbol{p}_k)_{k\in\mathbb{N}}$ is constructed after the rule

$$\boldsymbol{p}_{k+1} = \boldsymbol{p}_k - \alpha_k \nabla \mathscr{J}(\boldsymbol{p}_k) \; . \tag{8.1}$$

The gradient $\nabla \mathscr{J}(\boldsymbol{p}_k)$ (3.39) requires the evaluation of the terms

$$-\mathcal{H}^T \mathcal{C}_{dd}^{-1} (\boldsymbol{d}_{\text{obs}} - \boldsymbol{f}(\boldsymbol{y})) \tag{8.2}$$

and

$$\mathcal{R}_{yy}^{-1} (\boldsymbol{y} - \mathcal{X}\boldsymbol{\beta}) \tag{8.3}$$

Using (4.70) and (3.21), the term (8.2) can be written as

$$-\sum_{\ell=1}^{M} \epsilon_\ell^{-2}(d_\ell - f_\ell) \cdot \boldsymbol{h}_\ell \tag{8.4}$$

which is a linear combination of the discrete sensitivity fields $\boldsymbol{h}_\ell$. Due to the linearity of sensitivity integral (4.64) and the differential operators with respect to the adjoint states (in all involved BVPs (4.54), (4.57) and (4.62)), we can simply replace the right-hand side of the adjoint equation (4.54) by the sum

$$\sum_{\ell=1}^{M} \epsilon_\ell^{-2}(d_\ell - f_\ell) \cdot \eta_\varepsilon^\ell \,. \tag{8.5}$$

This way, it is enough to solve only one coupled set of adjoint equations per simulated quantity, no matter how many measuring points are involved.

Another advantage of this approach is related to adaptive mesh refinement. If we applied adaptive mesh refinement to the solution of (4.54) in the Gauss-Newton approach, we would end up with as many differently refined meshes as measurement locations. In each evaluation of the sensitivity integral (4.64), the mesh for the adjoint solution would have to be recombined with the adaptively refined mesh for the forward solution. After each measuring location, the mesh would have to be reset or recreated. In the gradient-based approach, the recombination would be required only once per measurement type.

The evaluation of the term (8.3) requires the solution of a large linear system containing the covariance matrix $\mathcal{R}_{yy}$. This difficulty is handled by Klein et al. [2014] by using an appropriate preconditioner for an iterative solver with FFT-based computation of the matrix-vector products.

Due to the effort that would be required to compute the gradient $\nabla \mathscr{J}(\boldsymbol{p}_k)$, we skipped the usage of the well known Wolfe condition, Armijo rule or curvature condition to check for the acceptance of a step length in the line search loop of the inversion algorithm 3.1.

Outlook

For future developments, a natural extension of the presented methods is a combination of h- or hp-adaptive DG with the diffusive L^2-projection on unstructured meshes (with hanging nodes refinement), preferably in combination with gradient-based inversion.

Further improvements regarding efficiency and parallel scalability of the linear solver for the DG discretizations of the transport equation may be achieved by a multilevel preconditioner in which the block Gauss-Seidel method with downwind numbering plays the role of a smoother [Kanschat, 2008a].

We have seen in Tables 7.3 and 7.5 that the number of unknowns and therefore the matrix assembly and linear solver times for DG(k) grow rapidly with the order k of the polynomial basis. On quadrilateral/hexahedral meshes, where quadrature points and shape functions can be constructed from a tensor product of 1-D objects, an excellent boost in performance can be achieved for the matrix assembly part with a technique called sum-factorization [Melenk et al., 2001].

Last but not least, the presented numerical schemes for the solution of the forward problems and their implementation can be employed in alternative parameter estimation schemes such as the Ensemble Kalman Filter (EnKF) or the Pilot Points Method [Alcolea et al., 2006; Schöniger et al., 2012].

Appendix

A Basic linear algebra

1 Symmetric positive (semi-)definite matrices

Definition A.1. *Let $N \geq 1$. A square matrix $\mathbf{\Sigma} \in \mathbb{R}^{N\times N}$ is called* ***symmetric*** *if it is equal to its transpose: $\mathbf{\Sigma} = \mathbf{\Sigma}^T$. A complex square matrix $\mathbf{\Sigma} \in \mathbb{C}^{N\times N}$ is called* ***Hermitian*** *if it is equal to its conjugate transpose: $\mathbf{\Sigma} = \mathbf{\Sigma}^H = \overline{\mathbf{\Sigma}}^T$.*

Definition A.2. *Let $N \geq 1$. Let $\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$. A square matrix $\mathbf{\Sigma} \in \mathbb{K}^{N\times N}$ is called* ***positive semi-definite*** *if $\boldsymbol{x}^H\mathbf{\Sigma}\boldsymbol{x} \geq 0$ for all $\boldsymbol{x} \in \mathbb{K}^N$. $\mathbf{\Sigma}$ is* ***positive definite*** *if $\boldsymbol{x}^H\mathbf{\Sigma}\boldsymbol{x} > 0$ for all nonzero $\boldsymbol{x} \in \mathbb{K}^N$.*

Theorem A.1. ***(Spectral Theorem)****. Let $\mathbf{\Sigma} \in \mathbb{C}^{N\times N}$ be Hermitian. Then, its eigenvalues $\lambda_1, ..., \lambda_N$ are real and and there exists an orthonormal basis of eigenvectors $\{\boldsymbol{u}_1, ..., \boldsymbol{u}_N\}$ forming a unitary matrix $\mathcal{U} \in \mathbb{C}^{N\times N}$ such that*

$$\mathbf{\Sigma} = \mathcal{U}\mathbf{\Lambda}\mathcal{U}^H \tag{A.1}$$

where $\mathbf{\Lambda} = diag(\lambda_1, ..., \lambda_N)$. If, additionally, $\mathbf{\Sigma}$ is positive definite (positive semi-definite) the eigenvalues of $\mathbf{\Sigma}$ are positive (non-negative).

Proof: *See [Horn and Johnson, 2013] Theorem 2.5.6 and Theorems 4.1.8 and 7.2.1.*

For an invertible matrix $\mathcal{A} \in \mathbb{R}^{N\times N}$ the matrix $\mathbf{\Sigma} = \mathcal{A}\mathcal{A}^T$ is symmetric positive definite. We see that the factorization of a symmetric positive (semi-)definite matrix $\mathbf{\Sigma}$ is not unique.

Definition A.3. *Let $N \geq 1$. Let $\mathbf{\Sigma} \in \mathbb{C}^{N\times N}$ be a Hermitian positive (semi-)definite matrix. A matrix $\mathcal{B} \in \mathbb{C}^{N\times N}$ is called a* ***square root*** *of $\mathbf{\Sigma}$ if $\mathcal{B} \cdot \mathcal{B}^H = \mathbf{\Sigma}$.*

Remark A.1. *Let $\mathbf{\Sigma} \in \mathbb{C}^{N\times N}$ be a Hermitian positive (semi-)definite matrix. Let*

$$\mathbf{\Lambda}^{1/2} = \operatorname{diag}\left(\sqrt{\lambda_1}, ..., \sqrt{\lambda_N}\right) \tag{A.2}$$

where the λ_k's are the eigenvalues of $\mathbf{\Sigma}$. For any orthogonal matrix $\mathcal{V} \in \mathbb{C}^{N\times N}$, the matrix

$$\mathcal{B} = \mathcal{U}\mathbf{\Lambda}^{1/2}\mathcal{V}^T \tag{A.3}$$

*is a square root of $\mathbf{\Sigma}$. The **symmetric square root** $\mathbf{\Sigma}^{1/2}$ is given by*

$$\mathbf{\Sigma}^{1/2} = \mathcal{U}\mathbf{\Lambda}^{1/2}\mathcal{U}^T. \tag{A.4}$$

Note that $\mathbf{\Sigma}^{1/2}$ and $\mathbf{\Sigma}$ share the property of being positive definite (or positive semi-definite).

The **Cholesky factorization** provides a numerically robust method to compute a further square root. We consider only the case $\mathbb{K} = \mathbb{R}$.

Theorem A.2. *Let $\mathbf{\Sigma} \in \mathbb{R}^{N\times N}$ be symmetric positive (semi-)definite. Then, there exists a unique (at least one) lower triangular matrix $\mathcal{L} \in \mathbb{R}^{N\times N}$ (**Cholesky factor**) with positive (non-negative) diagonal entries such that $\mathbf{\Sigma} = \mathcal{L}\mathcal{L}^T$.*

Proof: *See [Higham, 2002], §10.1. and §10.3.*

2 Matrix identities

Let $N = N_1 + N_2$. Let $\mathcal{M} \in \mathbb{R}^{N\times N}$ be symmetric with partitioning

$$\mathcal{M} = \begin{bmatrix} \mathcal{A} & \mathcal{B} \\ \mathcal{B}^T & \mathcal{C} \end{bmatrix} \tag{A.5}$$

such that $\mathcal{A} \in \mathbb{R}^{N_1\times N_1}$, $\mathcal{B} \in \mathbb{R}^{N_1\times N_2}$ and $\mathcal{C} \in \mathbb{R}^{N_2\times N_2}$.

Lemma A.1. *$\mathcal{M}$ is symmetric positive definite if and only if the submatrices $\mathcal{A} \in \mathbb{R}^{N_1\times N_1}$ and $\mathcal{C} \in \mathbb{R}^{N_2\times N_2}$ and their **Schur Complements***

$$\mathcal{S}_{\mathcal{A}} = \mathcal{C} - \mathcal{B}^T\mathcal{A}^{-1}\mathcal{B}$$

and

$$\mathcal{S}_{\mathcal{C}} = \mathcal{A} - \mathcal{B}\mathcal{C}^{-1}\mathcal{B}^T$$

are symmetric positive definite.

Proof: *See [Boyd and Vandenberghe, 2009] A.5.5.*

Lemma A.2. *Let $\mathcal{M}$ as in (A.5) be symmetric positive definite. The inverse of $\mathcal{M}$ is given by*

$$\mathcal{M}^{-1} = \begin{bmatrix} \mathcal{N}_{11} & \mathcal{N}_{12} \\ \mathcal{N}_{21} & \mathcal{N}_{22} \end{bmatrix} \tag{A.6}$$

$$\begin{aligned}
\textit{where, choosing the ansatz} \quad \mathcal{N}_{22} &= \mathcal{S}_{\mathcal{A}}^{-1}, && \text{(A.7)}\\
\textit{we get} \quad \mathcal{N}_{12} &= -\mathcal{A}^{-1}\mathcal{B}\mathcal{S}_{\mathcal{A}}^{-1} && \text{(A.8)}\\
\mathcal{N}_{21} &= -\mathcal{S}_{\mathcal{A}}^{-1}\mathcal{B}^T\mathcal{A}^{-1} && \text{(A.9)}\\
\mathcal{N}_{11} &= \mathcal{A}^{-1} + \mathcal{A}^{-1}\mathcal{B}\mathcal{S}_{\mathcal{A}}^{-1}\mathcal{B}^T\mathcal{A}^{-1}, && \text{(A.10)}
\end{aligned}$$

$$\begin{aligned}
\textit{or, alternatively, choosing the ansatz} \quad \mathcal{N}_{11} &= \mathcal{S}_{\mathcal{C}}^{-1}, && \text{(A.11)}\\
\textit{we get} \quad \mathcal{N}_{21} &= -\mathcal{C}^{-1}\mathcal{B}^T\mathcal{S}_{\mathcal{C}}^{-1} && \text{(A.12)}\\
\mathcal{N}_{12} &= -\mathcal{S}_{\mathcal{C}}^{-1}\mathcal{B}\mathcal{C}^{-1} && \text{(A.13)}\\
\mathcal{N}_{22} &= \mathcal{C}^{-1} + \mathcal{C}^{-1}\mathcal{B}^T\mathcal{S}_{\mathcal{C}}^{-1}\mathcal{B}\mathcal{C}^{-1}. && \text{(A.14)}
\end{aligned}$$

Proof: *The validity of $\mathcal{M}^{-1}\mathcal{M} = \mathcal{M}\mathcal{M}^{-1} = \mathbf{Id}$ can be verified easily.*

Equating (A.8) with (A.13) yields the matrix identity

$$\mathcal{A}^{-1}\mathcal{B}(\mathcal{C} - \mathcal{B}^T\mathcal{A}^{-1}\mathcal{B})^{-1} = (\mathcal{A} - \mathcal{B}\mathcal{C}^{-1}\mathcal{B}^T)^{-1}\mathcal{B}\mathcal{C}^{-1}. \tag{A.15}$$

If $N_1 \gg N_2$, the computationally demanding evaluation of $\mathcal{A}^{-1}$ can be replaced by a much more convenient evaluation of $\mathcal{C}^{-1}$. This is one crucial step in the inversion scheme (Chapter 3). Another useful matrix identity is shown to be

$$\mathcal{A}^{-1} + \mathcal{A}^{-1}\mathcal{B}(\mathcal{C} - \mathcal{B}^T\mathcal{A}^{-1}\mathcal{B})^{-1}\mathcal{B}^T\mathcal{A}^{-1} = (\mathcal{A} - \mathcal{B}\mathcal{C}^{-1}\mathcal{B}^T)^{-1} \tag{A.16}$$

by comparing (A.10) with (A.11). Cf. [Nowak, 2005] Appendix A and the references therein.

3 Quadratic functions

Consider a smooth scalar function $f : \mathbb{R}^N \longrightarrow \mathbb{R}$ and a vector $\boldsymbol{x} \in \mathbb{R}^N$. Then, the column vector

$$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} = \nabla f(\boldsymbol{x}) \tag{A.17}$$

is the gradient of f. The first order derivatives can as well be arranged as a row vector:

$$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}^T} = \left[\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right]^T. \tag{A.18}$$

Using this notation, the second order derivative or the Hessian of f can be written as

$$\nabla^2 f(\boldsymbol{x}) = \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x} \partial \boldsymbol{x}^T} \, . \tag{A.19}$$

A point $\hat{\boldsymbol{x}}$ satisfying the condition

$$\nabla f(\hat{\boldsymbol{x}}) = \mathbf{0} \tag{A.20}$$

is called a **stationary point** of f.

Let $\boldsymbol{t} \in \mathbb{R}^N$ and let $\mathcal{A} \in \mathbb{R}^{N \times N}$ be a symmetric matrix. We consider quadratic functions of the form

$$Q(\boldsymbol{x}) = \boldsymbol{t}^T \boldsymbol{x} + \frac{1}{2} \boldsymbol{x}^T \mathcal{A} \boldsymbol{x}. \tag{A.21}$$

defined in $\mathbb{R}^N$ or in a convex subset of $\mathbb{R}^N$. Clearly, the gradient of Q is

$$\nabla Q = \frac{\partial(\boldsymbol{x}^T \boldsymbol{t})}{\partial \boldsymbol{x}} + \frac{1}{2} \frac{\partial(\boldsymbol{x}^T \mathcal{A} \boldsymbol{x})}{\partial \boldsymbol{x}} = \boldsymbol{t} + \mathcal{A} \boldsymbol{x} \tag{A.22}$$

and the Hessian of Q is

$$\nabla^2 Q = \frac{1}{2} \frac{\partial^2(\boldsymbol{x}^T \mathcal{A} \boldsymbol{x})}{\partial \boldsymbol{x} \partial \boldsymbol{x}^T} = \mathcal{A}. \tag{A.23}$$

Depending on the properties of the matrix $\mathcal{A}$, a stationary point can be a minimizer, a maximizer or an inflection point. Throughout this work, we are essentially interested in global minimizers of a non-negative quadratic cost function.

Lemma A.3. ***(Second Order Sufficient Conditions)***

(a) $\hat{\boldsymbol{x}} = -\mathcal{A}^{-1} \boldsymbol{t}$ *is the unique global minimizer of (A.21) if and only if* $\mathcal{A}$ *is positive definite.*

(b) If $\mathcal{A}$ *is positive semi-definite, then every solution of the under-determined linear system* $\mathcal{A} \boldsymbol{x} = -\boldsymbol{t}$ *is a global minimizer of (A.21).*

Proof: *See Lemma 4.7 in [Nocedal and Wright, 2006].*

4 Toeplitz and circulant matrices

Two special classes of matrices play an important role in the spatial description of hydraulic conductivity fields: Toeplitz and circulant matrices. [Gray, 2006] gives an overview of their properties. The relation between circulant matrices and the discrete Fourier transformation is a major ingredient in the construction of efficient algorithms for the generation of random fields [Dietrich and Newsam, 1997] and for the computation of large cross-covariance matrices [Nowak et al., 2003].

Definition A.4. *Let $n \geq 1$. A matrix $\mathcal{T}_n \in \mathbb{R}^{n \times n}$ of the form*

$$\begin{bmatrix} t_0 & t_1 & t_2 & \dots & t_{n-1} \\ t_{-1} & t_0 & t_1 & & t_{n-2} \\ t_{-2} & t_{-1} & t_0 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & t_1 \\ t_{-(n-1)} & \dots & \dots & t_{-1} & t_0 \end{bmatrix} \tag{A.24}$$

*with entries $t_{i,j} = t_{j-i}$ is called a **Toeplitz matrix**.*

Definition A.5. *Let $n \geq 1$. A matrix $\mathcal{C}_n \in \mathbb{R}^{n \times n}$ of the form*

$$\begin{bmatrix} c_0 & c_1 & c_2 & \dots & c_{n-1} \\ c_{n-1} & c_0 & c_1 & & c_{n-2} \\ c_{n-2} & c_{n-1} & c_0 & \ddots & \vdots \\ \vdots & & \ddots & \ddots & c_1 \\ c_1 & \dots & \dots & c_{n-1} & c_0 \end{bmatrix} \tag{A.25}$$

*in which every row is a right cyclic shift of the row above it is called a **circulant matrix**.*

Definition A.6. ***(Discrete Fourier Transformation)***
Let $n \geq 1$. Given a complex vector $\mathbf{c} = (c_0, ..., c_{n-1})^T \in \mathbb{C}^n$ and the complex n-th root of unity

$$\omega_n := \exp\left(-\frac{2\pi i}{n}\right), \qquad \textit{where } i^2 = -1, \tag{A.26}$$

the vector $\hat{\mathbf{c}} = (\hat{c}_0, ..., \hat{c}_{n-1})^T \in \mathbb{C}^n$ defined by

$$\hat{c}_k = \frac{1}{\sqrt{n}} \sum_{\ell=0}^{n-1} c_\ell \cdot (\omega_n)^{k\ell} \qquad k = 0, ..., n-1 \tag{A.27}$$

*is the **discrete Fourier transform (DFT)** of **c**. Conversely, given a complex vector $\hat{\mathbf{c}}$, the **inverse DFT** is defined by*

$$c_k = \frac{1}{\sqrt{n}} \sum_{\ell=0}^{n-1} \hat{c}_\ell \cdot (\omega_n)^{-k\ell} \qquad k = 0, ..., n-1. \tag{A.28}$$

Cf. [Butz, 2006].

Introducing the **Fourier matrix** $\mathcal{F}_n$ with entries $(\mathcal{F}_n)_{k\ell} = \frac{1}{\sqrt{n}}(\omega_n)^{k\ell}$ it can be shown that its inverse fulfills $\mathcal{F}_n^{-1} = \mathcal{F}_n^H$, i.e. $\mathcal{F}_n$ is unitary. Then, the forward DFT (A.27) reads $\hat{\boldsymbol{c}} = \mathcal{F}_n \boldsymbol{c}$ and the backward DFT (A.28) reads $\boldsymbol{c} = \mathcal{F}_n^H \hat{\boldsymbol{c}}$.

Remark A.2. *A direct implementation of the DFT of length n would require $\mathcal{O}(n^2)$ floating point operations. The **fast Fourier transform (FFT)** can achieve the same result with only $\mathcal{O}(n\log(n))$ operations. Its basic idea was published by [Cooley and Tukey, 1965]. We use the implementation by [Frigo and Johnson, 2005].*

Theorem A.3. ***(Eigenvalues and eigenvectors of a circulant matrix)***
Let $\mathcal{C}_n$ be a circulant matrix of the form (A.25), fully determinable by its first column $\boldsymbol{c} = (c_0, ..., c_{n-1})^T$. Then, $\mathcal{C}_n$ has eigenvectors

$$\boldsymbol{v}_\ell = \frac{1}{\sqrt{n}} \left(1,\ \omega_n^\ell, ...,\ \omega_n^{(n-1)\ell} \right)^T \tag{A.29}$$

with corresponding eigenvalues

$$\lambda_\ell = \left(\sqrt{n} \cdot \mathcal{F}_n \cdot \boldsymbol{c}\right)_\ell \qquad \ell = 0, ..., n-1. \tag{A.30}$$

Proof: *See [Gray, 2006], §3.1.*

Theorem A.3 states that any circulant matrix $\mathcal{C}_n$ has a spectral decomposition of the form

$$\mathcal{C}_n = \mathcal{F}_n \cdot \operatorname{diag}(\lambda_0, ..., \lambda_{n-1}) \cdot \mathcal{F}_n^H. \tag{A.31}$$

All circulant matrices share the same eigenvectors which are the columns of the Fourier matrix and the vector of eigenvalues is proportional to the forward DFT of the first column:

$$\boldsymbol{\lambda} = \sqrt{n}\, \hat{\boldsymbol{c}} = \sqrt{n}\, \mathcal{F}_n \boldsymbol{c}\,.$$

One square root factorization of $\mathcal{C}_n$ is given by

$$\mathcal{B}_n = \mathcal{F}_n \cdot \operatorname{diag}\left(\lambda_0^{1/2}, ..., \lambda_{n-1}^{1/2}\right) \tag{A.32}$$

since $\mathcal{B}_n \cdot \mathcal{B}_n^H = \mathcal{C}_n$.

B Basic multivariate statistics

1 Random vectors and probability distributions

We take it for granted that the reader is familiar with the basic concepts of a continuous random variable, its probability density and cumulative distribution functions, and its expected value and variance. We start directly with some basic definitions from multivariate statistics. For a mathematically rigorous introduction, the reader might look into the textbooks [Bilodeau and Brenner, 1999] or [Georgii, 2008].

Definition B.1. *Let $N \geq 1$. A multivariate random variable or a* ***random vector*** $\boldsymbol{X} = (X_1, ..., X_N)^T$ *is a finite sequence of random variables on the same probability space $(\boldsymbol{\Omega}, \mathcal{F}, P)$, where $\boldsymbol{\Omega}$ is the sample space, $\mathcal{F}$ is the collection of all events (σ-algebra), and P is the probability measure.*

Definition B.2. *For continuous $X_1, ..., X_N$, the* ***cumulative distribution function (cdf)*** *is the function $F : \mathbb{R}^N \longrightarrow [0, 1]$ defined by*

$$F(\boldsymbol{x}) = P(\boldsymbol{X} \leq \boldsymbol{x}) = P(X_1 \leq x_1, ..., X_N \leq x_N). \tag{B.1}$$

Definition B.3. *If the cumulative distribution function has the form*

$$F(\boldsymbol{x}) = F(x_1, ..., x_N) = \int\limits_{-\infty}^{x_1} \dots \int\limits_{-\infty}^{x_N} \varrho(x_1, ..., x_N)\, dx_N \dots dx_1 \tag{B.2}$$

where $\varrho : \mathbb{R}^N \longrightarrow [0, \infty)$ is a non-negative function with

$$\int\limits_{\mathbb{R}^N} \varrho(\boldsymbol{x})\, \boldsymbol{dx} = 1 \tag{B.3}$$

we say that ϱ is the ***probability density function (pdf)*** *of* $\boldsymbol{X}$.

Definition B.4. *The* ***mean*** *of a random vector $\boldsymbol{X}$ is defined as the vector of expected values of the random variables X_j:*

$$\boldsymbol{\mu}_X = E[\boldsymbol{X}] = \begin{bmatrix} E[X_1] \\ \vdots \\ E[X_N] \end{bmatrix} = \begin{bmatrix} \int\limits_{\mathbb{R}^N} x_1 \varrho(\boldsymbol{x}) \boldsymbol{dx} \\ \vdots \\ \int\limits_{\mathbb{R}^N} x_N \varrho(\boldsymbol{x}) \boldsymbol{dx} \end{bmatrix}, \tag{B.4}$$

provided that the integrals exist.

The $N \times N$ matrix

$$\boldsymbol{\Sigma}_{XX} = \mathit{Var}[\boldsymbol{X}] = \mathit{Cov}[\boldsymbol{X}, \boldsymbol{X}] = E[(\boldsymbol{X} - \boldsymbol{\mu}_X)(\boldsymbol{X} - \boldsymbol{\mu}_X)^T] \tag{B.5}$$

is the ***covariance matrix*** *of $\boldsymbol{X}$. In summary, we may write concisely*

$$\boldsymbol{X} \sim (\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_{XX}). \tag{B.6}$$

The i-th diagonal entry of $\boldsymbol{\Sigma}_{XX}$ is the ***variance*** *$\sigma_i^2 = E[(X_i - E[X_i])^2]$.*

2 Joint distributions and marginalization

Let $N_x, N_y \geq 1$. Let $N = N_x + N_y$. Let

$$\boldsymbol{X} \in \mathbb{R}^{N_x}, \boldsymbol{X} \sim (\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_{XX}) \text{ with density } \varrho_X(\boldsymbol{x}) > 0$$

and

$$\boldsymbol{Y} \in \mathbb{R}^{N_y}, \boldsymbol{Y} \sim (\boldsymbol{\mu}_Y, \boldsymbol{\Sigma}_{YY}) \text{ with density } \varrho_Y(\boldsymbol{y}) > 0.$$

Definition B.5. *The $N_x \times N_y$ matrix*

$$\boldsymbol{\Sigma}_{XY} = \mathit{Cov}[\boldsymbol{X}, \boldsymbol{Y}] = E[(\boldsymbol{X} - \boldsymbol{\mu}_X)(\boldsymbol{Y} - \boldsymbol{\mu}_Y)^T] \tag{B.7}$$

is the ***cross-covariance matrix*** *between $\boldsymbol{X}$ and $\boldsymbol{Y}$.*

Remark B.1. *$\boldsymbol{\Sigma}_{XY}$ is used to describe the correlation between $\boldsymbol{X}$ and $\boldsymbol{Y}$. It follows directly from the definition that*

$$\boldsymbol{\Sigma}_{YX} = \boldsymbol{\Sigma}_{XY}^T. \tag{B.8}$$

Remark B.2. ***(Linear transformation)***
Let $\boldsymbol{X} \in \mathbb{R}^{N_x}, \boldsymbol{X} \sim (\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_{XX})$. Let $\mathcal{A} \in \mathbb{R}^{N_y \times N_x}$ be a constant matrix and let $\boldsymbol{V} \in \mathbb{R}^{N_y}$ be a constant vector. Then,

$$\boldsymbol{Y} = \mathcal{A}\boldsymbol{X} + \boldsymbol{V} \quad \in \quad \mathbb{R}^{N_y}$$

is a random vector with mean $\boldsymbol{\mu}_Y = \mathcal{A}\boldsymbol{\mu}_X + \boldsymbol{V}$. The covariance and cross-covariance matrices are

$$\boldsymbol{\Sigma}_{YY} = E[\mathcal{A}(\boldsymbol{X} - \boldsymbol{\mu}_X) \cdot (\boldsymbol{X} - \boldsymbol{\mu}_X)^T \mathcal{A}^T] = \mathcal{A}\boldsymbol{\Sigma}_{XX}\mathcal{A}^T \tag{B.9}$$

and

$$\boldsymbol{\Sigma}_{YX} = E[\mathcal{A}(\boldsymbol{X} - \boldsymbol{\mu}_X) \cdot (\boldsymbol{X} - \boldsymbol{\mu}_X)^T] = \mathcal{A}\boldsymbol{\Sigma}_{XX} \tag{B.10}$$

and

$$\boldsymbol{\Sigma}_{XY} = \boldsymbol{\Sigma}_{XX}\mathcal{A}^T. \tag{B.11}$$

Definition B.6. *The combined random variable* $\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{X} \\ \boldsymbol{Y} \end{bmatrix} \in \mathbb{R}^N$ *is called the* ***joint*** *of* $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *whereas* $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *are called the* ***marginals*** *of* $\boldsymbol{Z}$*. Let* $\varrho_{X,Y}(\boldsymbol{x}, \boldsymbol{y})$ *denote the* ***joint density*** *for* $\boldsymbol{X}$ *and* $\boldsymbol{Y}$*. The* ***marginal density*** *of* $\boldsymbol{X}$ *is found by "integrating out" the variable* $\boldsymbol{y}$*:*

$$\varrho_X(\boldsymbol{x}) = \int\limits_{\mathbb{R}^{N_y}} \varrho_{X,Y}(\boldsymbol{x}, \boldsymbol{y}) \boldsymbol{dy}. \tag{B.12}$$

This procedure is called ***marginalization****. Similarly, the pdf*

$$\varrho_Y(\boldsymbol{y}) = \int\limits_{\mathbb{R}^{N_x}} \varrho_{X,Y}(\boldsymbol{x}, \boldsymbol{y}) \boldsymbol{dx} \tag{B.13}$$

is the marginal density of $\boldsymbol{Y}$*.*

3 Conditional distributions and stochastic independence

Definition B.7. *The* ***conditional density*** *of* $\boldsymbol{Y}$ *given* $\boldsymbol{X} = \boldsymbol{x}$*, written* $\boldsymbol{Y}|\boldsymbol{X} = \boldsymbol{x}$*, is defined by*

$$\varrho_{Y|X}(\boldsymbol{y}|\boldsymbol{x}) = \frac{\varrho_{X,Y}(\boldsymbol{x}, \boldsymbol{y})}{\varrho_X(\boldsymbol{x})}. \tag{B.14}$$

Definition B.8. $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *are called* ***independent*** *if*

$$\varrho_{X,Y}(\boldsymbol{x}, \boldsymbol{y}) = \varrho_X(\boldsymbol{x}) \cdot \varrho_Y(\boldsymbol{y}). \tag{B.15}$$

In this case, $\varrho_{Y|X}(\boldsymbol{y}|\boldsymbol{x}) = \varrho_{Y|X}(\boldsymbol{y})$, i.e. the outcome of Y does not depend on the outcome of X.

Theorem B.1. *If* $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *are independent, then*

$$E[\boldsymbol{X}, \boldsymbol{Y}] = E[\boldsymbol{X}]E[\boldsymbol{Y}] \quad \text{and} \quad \Sigma_{XY} = \mathbf{0}\,. \tag{B.16}$$

Proof: *Cf. Theorems 4.7 and 4.11(d) [Georgii, 2008]*

The converse is not true in general.

Theorem B.2. *The conditional density of* $\boldsymbol{X}$ *given* $\boldsymbol{Y} = \boldsymbol{y}$ *is*

$$\varrho_{X|Y}(\boldsymbol{x}|\boldsymbol{y}) = \frac{\varrho_{Y|X}(\boldsymbol{y}|\boldsymbol{x}) \cdot \varrho_X(\boldsymbol{x})}{\varrho_Y(\boldsymbol{y})}. \tag{B.17}$$

Proof: *Cf.* ***Bayes' formula*** *in [Marden, 2013]*

For the sake of readability, we do not use subscripts for the density function when its meaning is clear from the context.

C The multivariate Gaussian distribution

1 Definition and basic properties

Definition C.1. *A random variable $X \in \mathbb{R}$ has a* ***normal distribution*** *with mean μ and variance σ^2, short*

$$X \sim \mathcal{N}(\mu, \sigma^2) \tag{C.1}$$

if it has the probability density function

$$\varrho(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right). \tag{C.2}$$

Definition C.2. *Let $\boldsymbol{\Sigma} \in \mathbb{R}^{N\times N}$ be symmetric positive definite matrix and let $\boldsymbol{\mu} \in \mathbb{R}^N$. A random vector $\boldsymbol{Y} \in \mathbb{R}^N$ has a* ***multivariate normal (or Gaussian) distribution*** *with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ if it has the probability density function*

$$\varrho(\boldsymbol{y}) = \frac{1}{(2\pi)^{N/2}} \cdot \frac{1}{|\det \boldsymbol{\Sigma}|^{1/2}} \cdot \exp\left(-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{y}-\boldsymbol{\mu})\right). \tag{C.3}$$

We write

$$\boldsymbol{Y} \sim \mathcal{N}_N(\boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{C.4}$$

Important special cases:

- $\mathcal{N}(0,1)$ and $\mathcal{N}_N(\mathbf{0}, \mathbf{Id})$ are called **standard normal** distributions.
- The absence of variance, $\boldsymbol{Y} \sim \mathcal{N}_N(\boldsymbol{\mu}, \mathbf{0})$, is equivalent to $\boldsymbol{Y} = \boldsymbol{\mu}$ with 100% probability (being a constant random vector).
- **Singular or degenerate case:** If $\boldsymbol{\Sigma} \in \mathbb{R}^{N\times N}$ is positive semi-definite with $\operatorname{rank}(\boldsymbol{\Sigma}) = k < N$, by Theorem A.1, we have

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathcal{U}_1 & \mathcal{U}_2 \end{bmatrix} \begin{bmatrix} \operatorname{diag}(\sigma_1^2, ..., \sigma_k^2) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathcal{U}_1^T \\ \mathcal{U}_2^T \end{bmatrix} \tag{C.5}$$

  where $\sigma_1^2, ..., \sigma_k^2$ are the non-zero eigenvalues of $\boldsymbol{\Sigma}$ and the columns of $\mathcal{U}_1$ form a basis of $\operatorname{range}(\boldsymbol{\Sigma})$. The probability density function is defined only in the affine subspace $\boldsymbol{\mu} + \operatorname{range}(\boldsymbol{\Sigma}) \subset \mathbb{R}^N$ (see [Bilodeau and Brenner, 1999]):

$$\varrho(\boldsymbol{y}) = \frac{1}{(2\pi)^{k/2}} \cdot \frac{1}{\sigma_1 \cdot \ldots \cdot \sigma_k} \cdot \exp\left(-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-} (\boldsymbol{y}-\boldsymbol{\mu})\right) \tag{C.6}$$

where

$$\mathbf{\Sigma}^{-} = \begin{bmatrix} \mathcal{U}_1 & \mathcal{U}_2 \end{bmatrix} \begin{bmatrix} \text{diag}(\sigma_1^{-2}, ..., \sigma_k^{-2}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathcal{U}_1^T \\ \mathcal{U}_2^T \end{bmatrix} \tag{C.7}$$

is the Moore-Penrose pseudo-inverse of $\mathbf{\Sigma}$ (see [Higham, 2002], problem 20.3).

Lemma C.1. *The collection of random variables $X_1, ..., X_N$ is independent and identically distributed (**i.i.d.**) with $\mathcal{N}(0, \sigma^2)$ if and only if the random vector $\boldsymbol{X} = (X_1, ..., X_N)^T$ has the distribution $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{Id})$. $\boldsymbol{X}$ is called a **(Gaussian) white noise vector**.*

Theorem C.1. *Let $\boldsymbol{\mu} \in \mathbb{R}^N$ and let $\mathbf{\Sigma} \in \mathbb{R}^{N\times N}$ be a symmetric positive definite matrix with square root $\mathcal{B} \in \mathbb{R}^{N\times N}$.*

(a) If $\boldsymbol{X} \sim \mathcal{N}_N(\mathbf{0}, \mathbf{Id})$ then

$$\boldsymbol{Y} = \mathcal{B}\boldsymbol{X} + \boldsymbol{\mu} \sim \mathcal{N}_N(\boldsymbol{\mu}, \mathbf{\Sigma}). \tag{C.8}$$

(b) If $\boldsymbol{Y} \sim \mathcal{N}_N(\boldsymbol{\mu}, \mathbf{\Sigma})$ then

$$\boldsymbol{X} = \mathcal{B}^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}) \sim \mathcal{N}_N(\mathbf{0}, \mathbf{Id}). \tag{C.9}$$

Proof: *Apply the transformation formula of multidimensional integration for the change of variable $\boldsymbol{Y} = \Phi(\boldsymbol{X}) = \mathcal{B}\boldsymbol{X} + \boldsymbol{\mu}$ and make use of $|\det \mathcal{B}| = |\det \mathbf{\Sigma}|^{1/2}$, see [Georgii, 2008], §9.1. for the proof of (a). The proof of (b) follows the same argumentation.*

Theorem C.2. *Let $\boldsymbol{\mu} \in \mathbb{R}^N$ and let $\mathbf{\Sigma} \in \mathbb{R}^{N\times N}$ be a symmetric positive semi-definite matrix with square root $\mathcal{B} \in \mathbb{R}^{N\times N}$. If $\boldsymbol{X} \sim \mathcal{N}_N(\mathbf{0}, \mathbf{Id})$ then*

$$\boldsymbol{Y} = \mathcal{B}\boldsymbol{X} + \boldsymbol{\mu} \sim \mathcal{N}_N(\boldsymbol{\mu}, \mathbf{\Sigma}). \tag{C.10}$$

Proof:

$$E[\boldsymbol{Y}] = E[\mathcal{B}\boldsymbol{X} + \boldsymbol{\mu}] = \mathcal{B}\,\underbrace{E[\boldsymbol{x}]}_{\mathbf{0}} + \boldsymbol{\mu} = \boldsymbol{\mu}$$

and

$$Cov[\boldsymbol{Y}, \boldsymbol{Y}] = E[\boldsymbol{Y}\boldsymbol{Y}^T] = E[\mathcal{B}\boldsymbol{X}\boldsymbol{X}^T\mathcal{B}^T] = \mathcal{B}\; E[\boldsymbol{X}\boldsymbol{X}^T]\; \mathcal{B}^T = \mathcal{B}\;\mathbf{Id}\;\mathcal{B}^T = \mathbf{\Sigma}.$$

For many applications in which the likelihood function has the form (C.3), it is more convenient to consider the natural logarithm of the likelihood function,

$$\ln(\varrho(\boldsymbol{y})) = -\frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{\mu}) - const. \tag{C.11}$$

called the **log-likelihood** or the **information content** in [Sun, 1994]. We work with the non-negative quadratic function

$$L(\boldsymbol{y}) := -\ln(\varrho(\boldsymbol{y})) \tag{C.12}$$

called the **normal negative log-likelihood**.

Remark C.1. *Because the logarithm is a monotonically increasing function, $\varrho(\boldsymbol{y})$ assumes its maximum at the same point where $L(\boldsymbol{y})$ assumes its minimum.*

Remark C.2. *$L(\boldsymbol{y})$ is a quadratic form with Hessian*

$$\frac{\partial^2 L(\boldsymbol{y})}{\partial \boldsymbol{y} \partial \boldsymbol{y}^T} = \boldsymbol{\Sigma}^{-1} \tag{C.13}$$

and $\widehat{\boldsymbol{y}} = \boldsymbol{\mu}$ is a global minimum of $L(\boldsymbol{y})$ (see Lemma A.3).

In other words, for a Gaussian distributed random vector $\boldsymbol{Y} \sim \mathcal{N}_N(\boldsymbol{\mu}_Y, \boldsymbol{\Sigma})$, the maximizer of the probability density $\rho(\boldsymbol{y})$ coincides with the expected value

$$\boldsymbol{\mu}_Y = E[\boldsymbol{Y}] = \arg\max_{\boldsymbol{y}\in\mathbb{R}^N}\{\varrho(\boldsymbol{y})\}. \tag{C.14}$$

Furthermore, the covariance matrix $\boldsymbol{\Sigma}$ of a Gaussian distribution is the inverse of the Hessian of $L(\boldsymbol{y})$:

$$\boldsymbol{\Sigma} = \left(\frac{\partial^2 L(\boldsymbol{y})}{\partial \boldsymbol{y} \partial \boldsymbol{y}^T}\right)^{-1}. \tag{C.15}$$

2 Conditioning

Let $N_x, N_y \geq 1$ and $N = N_x + N_y$. Let $\mathbf{\Sigma}_{XX} \in \mathbb{R}^{N_x \times N_x}$ and $\mathbf{\Sigma}_{YY} \in \mathbb{R}^{N_Y \times N_Y}$ be symmetric positive definite matrices. Let $\boldsymbol{\mu}_X \in \mathbb{R}^{N_x}$ and $\boldsymbol{\mu}_Y \in \mathbb{R}^{N_y}$. Let

$$\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}_X, \mathbf{\Sigma}_{XX})$$

and

$$\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{\mu}_Y, \mathbf{\Sigma}_{YY})$$

be two normally distributed random vectors correlated through $\mathbf{\Sigma}_{XY}$. If $\mathbf{\Sigma}_{XY} = \mathbf{0}$, it can be shown that the two random vectors $\boldsymbol{X}$ and $\boldsymbol{Y}$ are independent, i.e. for normal distributions the converse of Theorem B.1 is also true. If $\mathbf{\Sigma}_{XY}$ is chosen in such a way that the matrix

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{XX} & \mathbf{\Sigma}_{XY} \\ \mathbf{\Sigma}_{YX} & \mathbf{\Sigma}_{YY} \end{bmatrix} \in \mathbb{R}^{N \times N} \tag{C.16}$$

is also symmetric positive definite, then the composite vector $\begin{bmatrix} \boldsymbol{X} \\ \boldsymbol{Y} \end{bmatrix} \in \mathbb{R}^N$ is normally distributed with mean $\begin{bmatrix} \boldsymbol{\mu}_X \\ \boldsymbol{\mu}_Y \end{bmatrix} \in \mathbb{R}^N$ and covariance matrix $\mathbf{\Sigma}$.

Theorem C.3. *Let $\mathbf{\Sigma} \in \mathbb{R}^{N \times N}$ be a symmetric positive definite matrix of the form (C.16). Let $\boldsymbol{X} \in \mathbb{R}^{N_x}$ and $\boldsymbol{Y} \in \mathbb{R}^{N_y}$ be two random vectors fulfilling*

$$\begin{bmatrix} \boldsymbol{X} \\ \boldsymbol{Y} \end{bmatrix} \sim \mathcal{N}_N \left(\begin{bmatrix} \boldsymbol{\mu}_X \\ \boldsymbol{\mu}_Y \end{bmatrix}, \mathbf{\Sigma} \right). \tag{C.17}$$

Then, the conditional distribution of $\boldsymbol{Y}$ given $\boldsymbol{X} = \boldsymbol{x}$ is

$$\mathcal{N}(\widehat{\boldsymbol{Y}}, \mathbf{\Sigma}_{YY|X}) \tag{C.18}$$

*where the **conditional mean** $\widehat{\boldsymbol{Y}}$ is equal to*

$$\boldsymbol{\mu}_Y + \mathbf{\Sigma}_{YX} \mathbf{\Sigma}_{XX}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_X) \tag{C.19}$$

*and the **conditional covariance matrix** $\mathbf{\Sigma}_{YY|X}$ is the Schur complement of $\mathbf{\Sigma}_{XX}$:*

$$\mathbf{\Sigma}_{YY|X} = \mathbf{\Sigma}_{YY} - \mathbf{\Sigma}_{YX} \mathbf{\Sigma}_{XX}^{-1} \mathbf{\Sigma}_{XY}. \tag{C.20}$$

Proof: *See [Kaipio and Somersalo, 2005], §3.4*

3 χ^2 distribution

Definition C.3. *For each $N \geq 1$, a random variable X has a χ^2**-distribution** with N degrees of freedom, or short:*

$$X \sim \chi^2_N, \tag{C.21}$$

if it has the density function

$$\varrho(x) = \begin{cases} \dfrac{x^{N/2-1}}{2^{N/2} \cdot \Gamma(N/2)}\, e^{-x/2} & \textit{for } x \geq 0, \\ 0 & \textit{for } x < 0. \end{cases} \tag{C.22}$$

Remark C.3. *Using integration by parts, it can be shown that the expected value and the variance for $X \sim \chi^2_N$ are $E[X] = N$ and $Var[X] = 2N$.*

Theorem C.4. *Let $N \geq 1$. The squared sum of i.i.d. random variables $X_1, ..., X_N$ with standard normal distribution has χ^2-distribution with N degrees of freedom:*

$$X_1^2 + ... + X_N^2 \sim \chi^2_N \tag{C.23}$$

Proof: *See [Georgii, 2008], §9.2.*

Remark C.4. *Using the central limit theorem, it can be shown that the χ^2_N distribution can be approximated by the normal distribution with mean value N and standard deviation $\sqrt{2N}$:*

$$X \sim \chi^2_N \qquad \Longrightarrow \qquad X \sim \mathcal{N}(N, 2N) \;\text{ as }\; N \to \infty \tag{C.24}$$

D Random fields and variogram models

Definition D.1. *Let $d \geq 1$. Let $\overline{\Omega} \subset \mathbb{R}^d$ be a compact subset of the Euclidean space. A second order **(spatial) random field** is a function $Y : \Omega \longrightarrow \mathbb{R}$ whose values are random variables which can be described by the first two statistical moments:*

1. *the **mean function***

$$\mu_Y(\vec{x}) = E[Y(\vec{x})] \tag{D.1}$$

which gives the expected value at any point $\vec{x} \in \Omega$,

2. *the **covariance function***

$$R(\vec{x}, \vec{y}) = E\big[(Y(\vec{x}) - \mu_Y(\vec{x}))(Y(\vec{y}) - \mu_Y(\vec{y}))\big] \tag{D.2}$$

describing the spatial covariance of Y between any two locations $\vec{x}, \vec{y} \in \Omega$.

*Y is called **stationary** if the mean is constant,*

$$\mu_Y(\vec{x}) = const. \quad \forall \vec{x} \tag{D.3}$$

and the covariance function

$$R(\vec{x}, \vec{y}) = R(\vec{h}), \tag{D.4}$$

depends only on the separation vector $\vec{h} = \vec{x} - \vec{y}$.

The separation distance in the euclidean norm is $h = \|\vec{h}\|_2$. We will work with the distance

$$\tilde{h}(\ \vec{x},\ \vec{y}\ ) := \sqrt{\sum_{j=1}^{d} \left(\frac{x_j - y_j}{\ell_j} \right)^2} \tag{D.5}$$

that is scaled by different **correlation lengths** ℓ_j. This can be used to simulate anisotropic behavior in the different coordinate directions. In typical sedimentary layers, $\ell_1 \approx \ell_2$ and $\ell_3 \ll \ell_1$.

Most often, the constant in (D.3) is unknown and needs to be estimated from the data. For this reason, it may be more convenient to work with the following model:

Definition D.2. *A random field Y is called **intrinsic** if*

$$\begin{aligned} E[Y(\vec{x}) - Y(\vec{y})] &= 0 & \text{(D.6)} \\ R(\vec{x}, \vec{y}) &= r(\tilde{h}) := \sigma^2 - \gamma(\tilde{h}) \quad \forall \vec{x}, \vec{y} \in \Omega, & \text{(D.7)} \end{aligned}$$

*where $\sigma^2 < \infty$ is the variance and $\gamma(\tilde{h}) = \frac{1}{2} E[(Y(\vec{x}) - Y(\vec{y}))^2]$ is a **semi-variogram** function.*

The most commonly used variogram models are

- the Gaussian model,

$$\gamma(\tilde{h}) = \sigma^2 \left(1 - \exp\left(-\tilde{h}^2 \right) \right), \tag{D.8}$$

- the exponential model,

$$\gamma(\tilde{h}) = \sigma^2 \left(1 - \exp\left(-\tilde{h} \right) \right), \tag{D.9}$$

- the spherical model,

$$\gamma(\tilde{h}) = \begin{cases} \sigma^2 \left(\dfrac{3}{2} \dfrac{\tilde{h}}{\alpha} - \dfrac{1}{2} \dfrac{\tilde{h}^3}{\alpha^3} \right) & \text{for} \quad 0 \leq \tilde{h} \leq \alpha, \\ \sigma^2 & \text{for} \quad \tilde{h} > \alpha, \end{cases} \tag{D.10}$$

for some prescribed range $\alpha > 0$.

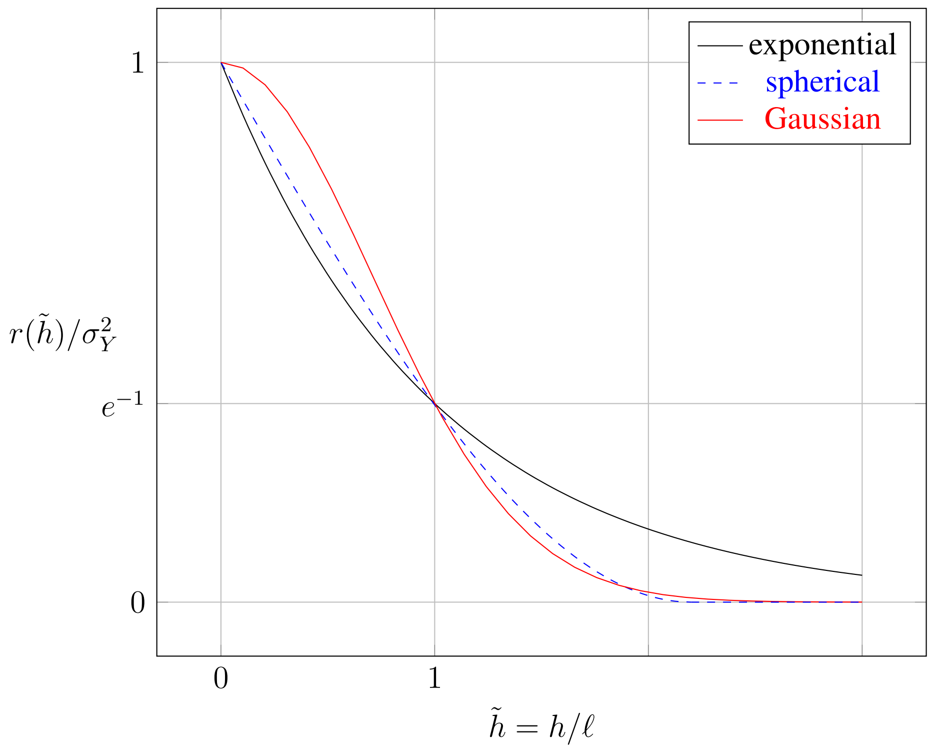


Figure D.1: Comparison of variogram models

Due to the smooth transition ($r'|_{\tilde{h}=0} = 0$) of the Gaussian covariance function for short distances, the corresponding field looks very smooth, see Figure 2.4. The Gaussian model drops more quickly to zero than the exponential model after the correlation length is exceeded ($h > \ell$). The near-distance behavior of the spherical model is similar to that of the exponential model, whereas for $h > \alpha\ell$, the values of Y are immediately uncorrelated. As in the Gaussian model, this tend to generate connected zones with a smaller radius. Choosing $\alpha = 2.2$ for the spherical model, we obtain $r(1)/\sigma_Y \approx e^{-1}$ (dashed blue curve in Figure D.1), which can be compared to the curves of the other two models.

Remark D.1. *Given any two locations $\vec{x}, \vec{y} \in \Omega$, the covariance function (D.7) fulfills the following properties:*

- $R(\vec{x}, \vec{y}) \geq 0$ *(non-negative)*
- $R(\vec{x}, \vec{y}) = R(\vec{y}, \vec{x})$ *(symmetric)*
- $R(\vec{x} + \vec{h}, \vec{y}) = R(\vec{y} + \vec{h}, \vec{x}) = R(\vec{h})$ *for any lag $\vec{h}$* *(translation-invariant)*

E Basic definitions of functional analysis

1 Function spaces

The notion of norm, completeness and inner product in a real vector space, the concept of Lebesgue integral as well as the upcoming definitions and statements can be looked up in [Hinze et al., 2009] or in introductory textbooks on linear functional analysis and finite element analysis [Brenner and Scott, 2008].

Recall that a complete, normed vector space V is called a **Banach space**. A vector space H with inner product $(.,.)_H$ is called a **Hilbert space** if it is complete under the induced norm $\|u\| := \sqrt{(u,u)_H}$.

If U is a normed vector space and V is a Banach space, then the space of all linear and continuous (or bounded) mappings, denoted by $\mathcal{L}(U,V)$, and equipped with the norm

$$\|A\|_{\mathcal{L}(U,V)} := \sup_{u \in U, u \neq 0} \frac{\|Au\|_V}{\|u\|_U} < \infty \, ,$$

is a Banach space. An element of $\mathcal{L}(U,V)$ is called a **linear operator**. If $V = \mathbb{R}$, the space $U^* := \mathcal{L}(U, \mathbb{R})$ is called the **dual space** of U. An element $f \in U^*$ is called a **bounded linear functional**. The bilinear form $\langle .,. \rangle_{U^*,U} : U^* \times U \longrightarrow \mathbb{R}$ defined by

$$\langle \, L, \; u \, \rangle_{U^*,U} := L(u) \tag{E.1}$$

is called **duality pairing** between U^* and U. We identify $\langle \, u, \; L \, \rangle_{U,U^*}$ with $\langle \, L, \; u \, \rangle_{U^*,U}$.

Theorem E.1. ***(Riesz representation theorem)***
Let H be a real Hilbert space with inner product $(.,.)_H$. Then, for any bounded linear functional $L \in H^*$ there exists a unique $v \in H$ with $\|L\|_{H^*} = \|v\|_H$ such that

$$L(u) = \langle L, u \rangle_{H^*,H} := (v,u)_H \quad \forall u \in H \, .$$

Let $\Omega \subset \mathbb{R}^d$ be an open subset with piecewise smooth boundary. Let u be a real-valued function on Ω for which the Lebesgue integral

$$\int_\Omega u(\vec{x})\, d\vec{x} := \int_\Omega u(\vec{x})\, d\mu(\vec{x}) \tag{E.2}$$

exists. Consider the inner product

$$(u, v)_{0,\Omega} := \int_\Omega u(\vec{x})v(\vec{x})\, d\vec{x}\,. \tag{E.3}$$

The **Lebesgue space** of square-integrable functions

$$L^2(\Omega) := \{u : \Omega \longrightarrow \mathbb{R} : \|u\|_{0,\Omega} < \infty\} \tag{E.4}$$

is defined as a space of equivalence classes of functions, equipped with the norm

$$\|u\|_{L^2(\Omega)} := \|u\|_{0,\Omega} := \sqrt{(u, u)_{0,\Omega}}\,. \tag{E.5}$$

In $L^2(\Omega)$, two functions are said to be identical (almost everywhere) if they differ at most on a set of zero measure. $L^2(\Omega)$ with inner product $(.,.)_{0,\Omega}$ is a Hilbert space. For vector-valued square-integrable functions in $[L^2(\Omega)]^d$, the definition of the inner product (E.3) is straight forwardly extended to

$$\left(\, \vec{u},\ \vec{v}\, \right)_{0,\Omega} = \int_\Omega \vec{u}(\vec{x}) \cdot \vec{v}(\vec{x})\, d\vec{x} = \sum_{j=1}^{d} (u_j, v_j)_{0,\Omega}\,. \tag{E.6}$$

$C^0(\Omega)$ denotes the space of continuous functions. For $s > 0$, $C^s(\Omega)$ denotes the space of s-times continuously differentiable functions, $C^\infty(\Omega)$ the space of infinitely often differentiable functions and $C_0^\infty(\Omega)$ the subspace of $C^\infty(\Omega)$-functions that have compact support in Ω (i.e. vanishing almost everywhere outside a compact subset of Ω).

For a vector field $\vec{A} \in [C^1(\Omega)]^d$ (i.e. all components are C^1-functions), the **Gauss's divergence theorem** states:

$$\int_\Omega \text{div}\vec{A}(\vec{x})\, d\vec{x} = \int_{S=\partial\Omega} \vec{n} \cdot \vec{A}(\vec{x})\, dS(\vec{x})\,. \tag{E.7}$$

Applying this formula to the vector field $\vec{A} := v \cdot K\nabla u$, where $v \in C^1(\Omega)$, $u \in C^2(\Omega)$ and $K \in L^2(\Omega)$, yields **Green's first identity**:

$$\int_\Omega \left(\text{div}(K\nabla u)\, v + K\nabla u \nabla v\right)\, d\vec{x} = \int_{S=\partial\Omega} \vec{n} \cdot K\nabla u\, dS(\vec{x})\,. \tag{E.8}$$

For a function $u \in L^2(\Omega)$, the function v denotes the **distributional partial derivative** or **weak partial derivative** of u with respect to the coordinate x_i, if $v \in L^2(\Omega)$ and

$$(\psi, v)_{0,\Omega} = -(\partial_{x_i}\psi, u)_{0,\Omega} \qquad \forall \psi \in C_0^\infty(\Omega)\,. \tag{E.9}$$

The same symbol is used as in the case of the classical partial derivative, since both definitions yield the same derivative for $C^s(\Omega)$-functions: $v = \partial_{x_i} u$ or $v = \partial_i u$. For a d-tuple $\alpha \in \mathbb{N}^d$ with $|\alpha| := \alpha_1 + ... + \alpha_d$, we set $\partial^\alpha u := \partial_1^{\alpha_1} \dots \partial_d^{\alpha_d} u$ and (E.9) is generalized to

$$(\psi, \partial^\alpha u)_{0,\Omega} = (-1)^{|\alpha|}(\partial^\alpha \psi, u)_{0,\Omega} \qquad \forall \psi \in C_0^\infty(\Omega)\,. \tag{E.10}$$

For $\alpha = 0$, the convention is $\partial^\alpha u := u$.

Sobolev spaces play an important role in the solution of partial differential equations. Their definition in the general case is based on the Lebesgue space $L^p(\Omega)$ with $p \geq 1$. We restrict ourselves to the case $p = 2$. For an integer s, we introduce the **Sobolev space**

$$H^s(\Omega) = \{v \in L^2(\Omega) : \partial^\alpha v \in L^2(\Omega) \quad \forall 0 \leq \alpha \leq s\}\,. \tag{E.11}$$

Equipped with the scalar product

$$(u, v)_{s,\Omega} = \sum_{|\alpha| \leq s} \left(\partial^\alpha u,\ \partial^\alpha v \right)_{0,\Omega} \tag{E.12}$$

and the induced norm

$$\|u\|_{s,\Omega} = \sqrt{(u,u)_{s,\Omega}} = \left(\sum_{|\alpha| \leq s} \|\ \partial^\alpha u\ \|_{0,\Omega}^2 \right)^{1/2}, \tag{E.13}$$

$H^s(\Omega)$ is a Hilbert space. In the distributional sense, Green's first identity (E.8) can be extended to $v \in H^1(\Omega)$, $u \in H^2(\Omega)$ and $K \in L^2(\Omega)$ ([Šolín, 2006], §A.4.8).

The function space in which we seek the flow field induced by the pressure field is the space

$$H(\text{div}, \Omega) = \left\{ \vec{q} \in [L^2(\Omega)]^d \ :\ \text{div}(\vec{q}) \in L^2(\Omega) \right\}\,. \tag{E.14}$$

Together with the inner-product given by

$$\left(\vec{u},\ \vec{v} \right)_{H(\text{div},\Omega)} = \left(\vec{u},\ \vec{v} \right)_{0,\Omega} + \left(\text{div}(\vec{u}),\ \text{div}(\vec{v}) \right)_{0,\Omega}, \tag{E.15}$$

$H(\text{div}, \Omega)$ is a Hilbert space [Brenner and Scott, 2008].

2 Broken Sobolev spaces

Given a partitioning of the domain $\Omega \subset \mathbb{R}^d$ as in (5.6), for any mesh element $t \in \mathcal{T}_h$, the Sobolev space $H^s(t)$ can be defined as in (E.11) if we replace Ω by t. The **broken Sobolev space** is then defined by

$$H^s(\mathcal{T}_h) = H^s(\Omega, \mathcal{T}_h) = \left\{ v \in L^2(\Omega) : v\big|_t \in H^s(t) \quad \forall\, t \in \mathcal{T}_h \right\} . \tag{E.16}$$

In the derivation of the Discontinuous Galerkin method (5.41) for the solution of the convection-diffusion problem (5.3), it is assumed that the 'real' solution u is in $H^2(\Omega) \subset H^2(\mathcal{T}_h)$. The broken polynomial space defined in (5.11) is a finite-dimensional approximation of the broken Sobolev space $H^s(\mathcal{T}_h)$. The broken gradient operator acting on the broken Sobolev space $H^1(\mathcal{T}_h)$ is defined by

$$\begin{array}{rccl} \nabla_h : & H^1(\mathcal{T}_h) & \longrightarrow & [L^2(\Omega)]^d \\ & \vec{q} & \mapsto & \nabla_h \vec{q}\,|_t := \nabla \vec{q}\,|_t \ . \end{array} \tag{E.17}$$

In the formulation of the DG method (5.41), the index h in the broken gradient operator is dropped for the sake of readability.

Similarly, the broken version of the space $H(\text{div}, \Omega)$ is defined by

$$H(\text{div}, \mathcal{T}_h) = \left\{ \vec{q} \in [L^2(\Omega)]^d \ : \ \vec{q}\,|_t \in H(\text{div}, t) \quad \forall\, t \in \mathcal{T}_h \right\} \tag{E.18}$$

where the function space $H(\text{div}, t)$ is defined as in (E.14) if we replace Ω by $t \in \mathcal{T}_h$. The broken divergence operator is

$$\begin{array}{rccl} \text{div}_h : & H(\text{div}, \mathcal{T}_h) & \longrightarrow & L^2(\Omega) \\ & \vec{q} & \mapsto & \text{div}_h(\vec{q}\,)|_t := \text{div}(\vec{q}\,|_t) \ . \end{array} \tag{E.19}$$

Lemma E.1. *A vector-valued function $\vec{q} \in H(\text{div}, \mathcal{T}_h)$ with bounded components and bounded first order derivatives belongs to $H(\text{div}, \Omega)$ if and only if*

$$[\![\, \vec{q}\,]\!]_f \cdot \vec{n}_f = 0 \qquad \forall f \in \mathcal{E}_h \ . \tag{E.20}$$

Proof: *Cf. Lemma 1.24 in [Pietro and Ern, 2012]*

This Lemma gives a characterization of the space $H(\text{div}, \Omega)$. It says that the normal component of the diffusive flux is continuous across internal faces. For the DG based discretization of the transport problem, this important property must be considered when constructing the discrete flux.

3 Differentiation in Banach spaces

Definition E.1. ***(Gâteaux differentiability)***
Let U and V be real Banach spaces and U_0 be a non-empty open subset of U. Let $F : U_0 \longrightarrow V$ be a mapping.

(a) Given $u_0 \in U_0$ and $\delta u \in U$, if the limit

$$\delta F(u_0; \delta u) := \lim_{\varepsilon \to 0+0} \frac{F(u_0 + \varepsilon \delta u) - F(u_0)}{\varepsilon} \quad \in V \tag{E.21}$$

exists, it is called the ***directional derivative*** *of F in the direction δu. If it exists for all $\delta u \in U$, the mapping $\delta u \mapsto \delta F(u_0; \delta u)$ is called the* ***first variation*** *of F in u_0. In general it is non-linear.*

(b) F is called ***Gâteaux differentiable*** *in u_0 if the first variation of F in u_0 exists and is bounded and linear, i.e. if there exists a linear and bounded operator $A : U \longrightarrow V$ such that*

$$\delta F(u_0; \delta u) = A(\delta u)\ . \tag{E.22}$$

For A we write $F'(u_0)$, or alternatively $D_u[F(u_0)]$ or $F'_u(u_0)$. If $V = \mathbb{R}$, we have $F'(u_0) \in U^*$. If, in addition to this, U is a Hilbert space, there exists a vector $\nabla F(u_o) \in U$, called the **gradient** of F, such that

$$F'(u_0)(\delta u) = \big(\nabla F(u_o), \delta u\big)_U \quad \forall \delta u \in U \tag{E.23}$$

according to the Riesz representation theorem.

Example:

Let $N \geq 1$ and let $\Omega \subset \mathbb{R}^{\mathbb{N}}$ be compact. Let $U = L^2(\Omega)$ and let $f \in U$. Consider the mapping

$$\begin{array}{rccl} F: & U & \longrightarrow & \mathbb{R} \\ & u & \mapsto & \big(f, u\big)_{0,\Omega}\ . \end{array} \tag{E.24}$$

Let $u \in U$. Then,

$$\begin{aligned} D_u[(f,u)_{0,\Omega}](\delta u) &= \lim_{\varepsilon \to 0+0} \frac{(f, u + \varepsilon \delta u)_{0,\Omega} - (f,u)_{0,\Omega}}{\varepsilon} \\ &= \left(f, \lim_{\varepsilon \to 0+0} \frac{u + \varepsilon \delta u - u}{\varepsilon}\right)_{0,\Omega} \\ &= \big(f, \delta u\big)_{0,\Omega}\ . \end{aligned} \tag{E.25}$$

Hence, $f \in L^2(\Omega)$ is the gradient of the linear functional F.

F Hardware in use

Machine name	`dnk`	`h3a`
Number of nodes	5	32
RAM per node	128 GB (DDR-3/1600)	128 GB (DDR-3/1333)
CPU-sockets per node	2	4
Total #cores	100	1024
OS	Linux CentOS 6.5	Debian GNU 7
CPU socket	*Intel® Xeon® E5-2680 v2*	*AMD Opteron™ 6212*
Clock speed	2.80 GHz	2.60 GHz
#cores	10	8
#threads	20	8
Launch date	Q3/2013	Q3/2011
L3 Cache	25 MB	16 MB
#memory channels	4	4
InfiniBand	QDR 40Gbit/s Mellanox MIS5023Q QDR-Switch	Mellanox 40G QDR single port PCIe Interconnect QSFP

Table F.1: Computing clusters at the IWR Heidelberg

Machine name	`quadxeon4`	`fna`
Number of nodes	1	1
RAM per node	1024 GB (DDR-3/1066)	128GB (DDR-3/1333 MHz)
CPU-sockets per node	4	4
Total #cores	40	48
OS	Debian GNU 7	Debian GNU 7
CPU socket	*Intel® Xeon® E7-4870*	*AMD Opteron™ 6172*
Clock speed	2.40 GHz	2.10 GHz
#cores	10	12
#threads	20	12
Launch date	Q2/2011	Q1/2010
L3 Cache	30 MB	12 MB
#memory channels	4	4

Table F.2: Multi-core machines at the IWR Heidelberg

G Hydraulic parameters

Soil type	θ
Gravel	$0.3 \; \dots \; 0.4$
Gravel and sand	$0.3 \; \dots \; 0.35$
Medium to coarse mixed sand	$0.35 \; \dots \; 0.4$
Uniform sand	$0.3 \; \dots \; 0.4$
Fine to medium mixed sand	$0.3 \; \dots \; 0.35$
Silt	$0.4 \; \dots \; 0.5$
Till	$0.1 \; \dots \; 0.2$
Clay	$0.45 \; \dots \; 0.55$

Table G.1: Typical values for the porosity [Bear and Cheng, 2010], §2.4.2

Soil type	$K[m/s]$	$Y = \ln(K)$
Gravel	$3 \cdot 10^{-4} \; \dots \; 3 \cdot 10^{-2}$	$-8.1 \; \dots \; -3.5$
Coarse sand	$9 \cdot 10^{-7} \; \dots \; 6 \cdot 10^{-3}$	$-13.9 \; \dots \; -5.1$
Medium sand	$9 \cdot 10^{-7} \; \dots \; 5 \cdot 10^{-4}$	$-13.9 \; \dots \; -7.6$
Fine sand	$2 \cdot 10^{-7} \; \dots \; 2 \cdot 10^{-4}$	$-15.4 \; \dots \; -8.5$
Silt, loess	$1 \cdot 10^{-9} \; \dots \; 2 \cdot 10^{-5}$	$-20.7 \; \dots \; -10.8$
Till	$1 \cdot 10^{-12} \; \dots \; 2 \cdot 10^{-6}$	$-27.6 \; \dots \; -13.1$
Clay	$1 \cdot 10^{-11} \; \dots \; 5 \cdot 10^{-9}$	$-25.3 \; \dots \; -19.1$
Unweathered marine clay	$8 \cdot 10^{-13} \; \dots \; 2 \cdot 10^{-9}$	$-27.8 \; \dots \; -20.0$

Table G.2: Typical values for the hydraulic conductivity [Domenico and Schwartz, 1998]

References

A. Alcolea, J. Carrera, and A. Medina. Pilot points method incorporating prior information for solving the groundwater flow inverse problem. *Advances in Water Resources*, 29:1678–1689, 2006. URL http://dx.doi.org/10.1016/j.advwatres.2005.12.009. 3, 123

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*, 39:1749–1779, 2002. URL http://dx.doi.org/10.1137/S0036142901384162. 43

M. Augustin, A. Caiazzo, A. Fiebach, J. Fuhrmann, V. John, A. Linke, and R. Umla. An assessment of discretizations for convection-dominated convection-diffusion equations. *Computer Methods in Applied Mechanics and Engineering*, 200:3395–3409, 2011. URL http://dx.doi.org/10.1016/j.cma.2011.08.012. 3, 4, 5

G. Bal. *Introduction to Inverse Problems, Lecture Notes, University of Illinois at Urbana-Champaign*. 2012. URL http://www.columbia.edu/~gb2030/PAPERS/IntroductionInverseProblems.pdf. Accessed: 2014-08-29. 21

P. Bastian. Benchmark 3D: Symmetric Weighted Interior Penalty Discontinuous Galerkin Scheme. In Jaroslav Fořt, Jiří Fürst, Jan Halama, Raphaèle Herbin, and Florence Hubert, editors, *Finite Volumes for Complex Applications VI Problems & Perspectives*, volume 4 of *Springer Proceedings in Mathematics*, pages 949–959. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-20670-2. doi: 10.1007/978-3-642-20671-9_92. URL http://dx.doi.org/10.1007/978-3-642-20671-9_92. 67

P. Bastian, K. Birken, S. Lang, K. Johannsen, N. Neuß, H. Rentz-Reichert, and C. Wieners. UG: A flexible software toolbox for solving partial differential equations. *Computing and Visualization in Science*, 1:27–40, 1997. URL http://www.iwr.uni-heidelberg.de/iwrwikiequipment/software/ug. Accessed: 2014-08-29. 75, 81

P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. Klöfkorn, R. Kornhuber, M. Ohlberger, and O. Sander. A Generic Grid Interface for Parallel and Adaptive Scientific Computing. Part II: Implementation and Tests in DUNE. *Computing*, 82 (2–3):121–138, 2008a. URL http://www.springerlink.com/content/gn177r643q2168g7/. 8

P. Bastian, M. Blatt, A. Dedner, C. Engwer, R. Klöfkorn, M. Ohlberger, and O. Sander. A Generic Grid Interface for Parallel and Adaptive Scientific Computing. Part I: Abstract Framework. *Computing*, 82(2–3):103–119, 2008b. URL http://www.springerlink.com/content/4v77662363u41534/. 8

P. Bastian, M. Blatt, A. Dedner, Ch. Engwer, J. Fahlke, C. Gräser, R. Klöfkorn, M. Nolte, M. Ohlberger, and O. Sander. DUNE Web page. http://www.dune-project.org, 2011. 8

J. Bear and A.H.-D. Cheng. *Modeling Groundwater Flow and Contaminant Transport*, volume 23 of *Theory and Applications of Transport in Porous Media*. Springer, 2010. ISBN 978-1-4020-6682-5. 4, 9, 11, 37, 38, 147

J. Bey and G. Wittum. Downwind numbering: robust multigrid for convection-diffusion problems. *Applied Numerical Mathematics*, 23:177–192, 1997. URL http://dx.doi.org/10.1007/978-3-663-14125-9_5. 6, 72

M. Bilodeau and D. Brenner. *Theory of Multivariate Statistics*. Springer Texts in Statistics. Springer, 1999. ISBN 978-0-387-98739-2. 131, 134

M. Blatt. A Parallel Algebraic Multigrid Method for Elliptic Problems with Highly Discontinuous Coefficients. *PhD Thesis, Heidelberg University*, 2010. URL http://nbn-resolving.de/urn:nbn:de:bsz:16-opus-108562. 3, 72

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2009. ISBN 978-0-521-83378-3. URL https://www.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf. 126

D. Braess. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press, 2007. ISBN 978-0521705189. 42

S. Brenner and R. Scott. *The Mathematical Theory of Finite Element Methods*, volume 15 of *Texts in Applied Mathematics*. Springer, 2008. ISBN 978-0-387-75934-0. 141, 143

F. Brezzi and M. Fortin. *Mixed and Hybrid Finite Elements*. Springer-Verlag, New York, 1991. 65

A. N. Brooks and T. J. R. Hughes. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 32(1-3):199–259, 1982. URL http://dx.doi.org/10.1016/0045-7825(82)90071-8. 4, 61, 62

M. Burger and F. Lucka. Maximum-A-Posteriori Estimates in Linear Inverse Problems with Log-concave Priors are Proper Bayes Estimators. 2014. URL http://arxiv.org/pdf/1402.5297.pdf. 22

E. Burman and P. Hansbo. Edge stabilization for Galerkin approximations of convection–diffusion–reaction problems. *Computer Methods in Applied Mechanics and Engineering*, 193(15-16):1437–1453, 2004. URL http://dx.doi.org/10.1016/j.cma.2003.12.032. 4

E. Burman, A. Quarteroni, and B. Stamm. Interior Penalty Continuous and Discontinuous Finite Element Approximations of Hyperbolic Equations. *Computer Methods in Applied Mechanics and Engineering*, 43(3):293–312, 2010. URL http://dx.doi.org/10.1007/s10915-008-9232-6. 4

T. Butz. *Fourier Transformation for Pedestrians*. Springer, 2006. ISBN 978-3-540-31108-9. 129

J. Carrera and S. P. Neuman. Estimation of Aquifer Parameters Under Transient and Steady State Conditions: 1. Maximum Likelihood Method Incorporating Prior Information. *Water Resources Research*, 22(2):199–210, 1986. URL http://dx.doi.org/10.1029/WR022i002p00199. 22

O. A. Cirpka and P. K. Kitanidis. Sensitivity of temporal moments calculated by the adjoint-state method and joint inversing of head and tracer data. *Advances in Water Resources*, 24:89–103, 2001. URL http://dx.doi.org/10.1016/S0309-1708(00)00007-5. 3, 4, 10, 45, 53, 62

J.W. Cooley and J.W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19:297–301, 1965. URL http://dx.doi.org/10.1090/S0025-5718-1965-0178586-1. 130

P.R.L. Couto and S.M.C. Malta. Interaction between sorption and biodegradation processes in the contaminant transport. *Ecological Modelling*, 214(1):65–73, 2008. URL http://dx.doi.org/10.1016/j.ecolmodel.2008.01.012. 4

E. Crestani, M. Camporese, D. Baú, and P. Salandin. Ensemble Kalman Filter Versus Ensemble Smoother for Assessing Hydraulic Conductivity via Tracer Test Data

Assimilation. *Hydrology and Earth System Sciences*, 9:13083–13115, 2012. URL http://dx.doi.org/10.5194/hessd-9-13083-2012. 3

G. De Marsily. *Quantitative Hydrogeology: Groundwater Hydrology for Engineers*. Academic Press, 1986. ISBN 0-12-208915-4. 37

A. Dedner, R. Klöfkorn, and M. Nolte. The dune-alugrid module. *CoRR*, abs/1407.6954, 2014. URL http://arxiv.org/abs/1407.6954. 75, 81

C. R. Dietrich and G. N. Newsam. A Fast and Exact Method for Multidimensional Gaussian Stochastic Simulations. *Water Resources Research*, 29(8):2861–2869, 1993. URL http://dx.doi.org/10.1029/93WR01070. 14

C. R. Dietrich and G. N. Newsam. Fast and exact simulation of stationary Gaussian processes through circulant embedding of the covariance matrix. *SIAM J. Sci. Comp.*, 18(4):1088 – 1107, 1997. URL http://dx.doi.org/10.1137/S1064827592240555. 14, 128

J.E. Doherty, M.N. Fienen, and R.J. Hunt. Approaches to highly parameterized inversion: Pilot-point theory, guidelines, and research directions. *U.S. Geological Survey Scientific Investigations Report 2010-5168, 36p*, 2010. URL http://pubs.usgs.gov/sir/2010/5168/. 3

P.A. Domenico and F.W. Schwartz. *Physical and chemical hydrogeology, 2nd edition*. Wiley, 1998. ISBN 978-0-471-59762-9. 12, 147

H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite Elements and Fast Iterative Solvers with Applications in Incompressible Fluid Dynamics*. Oxford University Press, 2005. 60, 62

H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Mathematics and Its Applications. Kluwer Academic Publishers, 2000. ISBN 0-7923-4157-0. 20, 21

A. Ern, A. F. Stephansen, and P. Zunino. A discontinuous Galerkin method with weighted averages for advection-diffusion equations with locally small and anisotropic diffusivity. *IMA Journal of Numerical Analysis*, 29:235–246, 2008. URL http://dx.doi.org/10.1093/imanum/drm050. 62, 66

A. Ern, A.F. Stephansen, and M. Vohralík. Guaranteed and robust discontinuous galerkin a posteriori error estimates for convection-diffusion-reaction problems. *Journal of Computational and Applied Mathematics*, 234(1):114–130, 2010. URL http://dx.doi.org/10.1016/j.cam.2009.12.009. 69

M. Frigo and S. G. Johnson. The Design and Implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005. URL http://dx.doi.org/10.1109/JPROC.2004.840301. 79, 130

H.-O. Georgii. *Stochastics: Introduction to Probability and Statistics.* Walter de Gruyter, 2008. ISBN 978-3-11-020676-0. 131, 133, 135, 138

E. H. Georgoulis, E. Hall, and P. Houston. Discontinuous Galerkin Methods on hp-Anisotropic Meshes II: A Posteriori Error Analysis and Adaptivity. *Applied Numerical Mathematics*, 59(9):2179 – 2194, 2009. URL http://dx.doi.org/10.1016/j.apnum.2008.12.008. 66, 69

E. Gordon, U. Shamir, and J. Bensabat. Optimal management of a regional aquifer under salinization conditions. *Water Resources Research*, 36:3193–3203, 2000. URL http://dx.doi.org/10.1029/2000WR900177. 4

R. M. Gray. Toeplitz and Circulant Matrices: A review. *Foundations and Trends® in Communications and Information Theory*, 2(3):155–239, 2006. 128, 130

W. Hackbusch and T. Probst. Downwind Gauß-Seidel smoothing for convection dominated problems. *Numerical Linear Algebra with Applications*, 4(2):85–102, 1997. URL http://dx.doi.org/10.1002/(SICI)1099-1506(199703/04)4:2<85::AID-NLA100>3.0.CO;2-2. 6, 73

C.F. Harvey and S.M. Gorelick. Temporal moment-generating equations: modeling transport and mass-transfer in heterogeneous aquifers. *Water Res. Research*, 31(8):1895–1911, 1995. URL http://dx.doi.org/10.1029/95WR01231. 40

N.J. Higham. *Accuracy and Stability of Numerical Algorithms: 2nd Ed.* Society for Industrial and Applied Mathematics, 2002. ISBN 0-89871-521-0. 126, 135

M. Hinze, R. Pinnau, and S. Ulbrich. *Optimization with PDE Constraints*, volume 23 of *Mathematical Modelling: Theory and Applications*. Springer, 2009. ISBN 978-1-4020-8838-4. 45, 141

R.A. Horn and C.R. Johnson. *Matrix Analysis: 2nd Ed.* Cambridge University Press, 2013. ISBN 978-0-521-54823-6. 125

T.J.R. Hughes, M. Mallet, and M. Akira. A new finite element formulation for computational fluid dynamics: II. Beyond SUPG. *Computer Methods in Applied Mechanics and Engineering*, 54(3):341–355, 1986. URL http://dx.doi.org/10.1016/0045-7825(86)90110-6. 4

V. John. A numerical study of a posteriori error estimators for convection-diffusion equations. *Comp. Methods in Appl. Mechanics and Engineering*, 190:757–781, 2000. URL http://dx.doi.org/10.1016/S0045-7825(99)00440-5. 69

V. John and P. Knobloch. On spurious oscillations at layers diminishing (SOLD) methods for convection-diffusion equations: Part I. *Computer Methods in Applied Mechanics and Engineering*, 196:2197–2215, 2007a. URL http://dx.doi.org/10.1016/j.cma.2006.11.013. 4

V. John and P. Knobloch. On the performance of SOLD methods for convection-diffusion problems with interior layers. *International Journal of Computing Science and Mathematics*, 1(2-4):245–258, 2007b. URL http://dx.doi.org/10.1504/IJCSM.2007.016534. 4

V. John and P. Knobloch. On spurious oscillations at layers diminishing (SOLD) methods for convection-diffusion equations: Part II - Analysis for P1 and Q1 finite elements. *Comp. Methods in Appl. Mechanics and Engineering*, 197(21-24):1997–2014, 2008. URL http://dx.doi.org/10.1016/j.cma.2007.12.019. 4

V. John and E. Schmeyer. Finite element methods for time-dependent convection-diffusion-reaction equations with small diffusion. *Computer Methods in Applied Mechanics and Engineering*, 198:475–494, 2008. URL http://dx.doi.org/10.1016/j.cma.2008.08.016. 5

V. John and Ellen Schmeyer. On Finite Element Methods for 3D Time-Dependent Convection-Diffusion-Reaction Equations with Small Diffusion. *BAIL 2008 - Boundary and Interior Layers: Lecture Notes in Computational Science and Engineering*, 69:173–181, 2009. URL http://dx.doi.org/10.1007/978-3-642-00605-0_13. 5

V. John, J.M. Maubach, and L. Tobiska. Nonconforming streamline-diffusion-finite-element-methods for convection-diffusion problems. *Numerische Mathematik*, 78(2): 165–188, 1997. URL http://dx.doi.org/10.1007/s002110050309. 82

T. Kailath. A Theorem of I. Schur and Its Impact on Modern Signal Processing. *I. Schur Methods in Operator Theory and Signal Processing, Operator Theory: Advances and Applications*, 18:9–30, 1986. URL http://dx.doi.org/10.1007/978-3-0348-5483-2_2. 14

J. Kaipio and E. Somersalo. *Statistical and Computational Inverse Problems*, volume 160 of *Applied Mathematical Sciences*. Springer, 2005. ISBN 0-387-22073-9. 21, 22, 137

G. Kanschat. Robust smoothers for high-order discontinuous Galerkin discretizations of advection-diffusion problems. *Journal of Computational and Applied Mathematics*, 218(1):53–60, 2008a. URL http://dx.doi.org/10.1016/j.cam.2007.04.032. 122

G. Kanschat. *Discontinuous Galerkin Methods for Viscous Incompressible Flow*. Advances in Numerical Mathematics. Springer, 2008b. ISBN 978-3-8350-5519-3. 5

P. K. Kitanidis. Parameter Uncertainty in Estimation of Spatial Functions: Bayesian Analysis. *Water Resources Research*, 22(4):499–507, 1986. URL http://dx.doi.org/10.1029/WR022i004p00499. 23

P. K. Kitanidis. Generalized covariance functions in estimation. *Mathematical Geology*, 25(5):525–540, 1993. URL http://dx.doi.org/10.1007/BF00890244. 12

P. K. Kitanidis. Quasi-Linear Geostatistical Theory For Inversing. *Water Resources Research*, 31(10):2411–2419, 1995. URL http://dx.doi.org/10.1029/95WR01945. 3, 22, 28

P. K. Kitanidis. On the geostatistical approach to the inverse problem. *Advances in Water Resources*, 19(6):333–342, 1996. URL http://dx.doi.org/10.1016/0309-1708(96)00005-X. 22, 23

P. K. Kitanidis. Comment on ”A reassessment of the groundwater inverse problem” by D. McLaughlin and L. R. Townley. *Water Resources Research*, 33(9):2199–2202, 1997. URL http://dx.doi.org/10.1029/97WR00998. 12

O. Klein, A. Ngo, O.A. Cirpka, P. Bastian, and O. Ippisch. Efficient Geostatistical Inversion of Transient Groundwater Flow. *in preparation*, 2014. 122

D. Kuzmin. On the design of general-purpose flux limiters for implicit FEM with a consistent mass matrix. I. Scalar convection. *Journal of Computational Physics*, 219(2):513–531, 2006. URL http://dx.doi.org/10.1016/j.jcp.2006.03.034. 4, 5

D. Kuzmin. Explicit and implicit FEM-FCT algorithms with flux linearization. *Journal of Computational Physics*, 228(7):2517–2534, 2009. URL http://dx.doi.org/10.1016/j.jcp.2008.12.011. 5

D. Kuzmin. A Guide to Numerical Methods for Transport Equations. 2010. URL http://www.mathematik.uni-dortmund.de/~kuzmin/cfdbook.html. Accessed: 2014-08-29. 3, 4

W. Li, W. Nowak, and Cirpka O.A. Geostatistical inverse modelling of transient pumping tests using temporal moments of drawdown. *Water Resources Research*, 41(8), 2005. URL http://dx.doi.org/10.1029/2004WR003874. 31, 32

Xiaoye S. Li. An Overview of SuperLU: Algorithms, Implementation, and User Interface. *ACM Transactions on Mathematical Software*, 31(3):302–325, 2005. URL http://dx.doi.org/10.1145/1089014.1089017. 75

J. L. López and E. P. Sinusía. Asymptotic expansions for two singularly perturbed convection-diffusion problems with discontinuous data: The quarter plane and the infinite strip. *Studies in Applied Mathematics*, 113:57 – 89, 2004. URL http://dx.doi.org/10.1111/j.1467-9590.2004.01508.x. 87

J. Luo, W.-M. Wu, M.N. Fienen, and O.A.Cirpka. A Nested-Cell Approach for In Situ Remediation. *Groundwater*, 44:266–274, 2006. URL http://dx.doi.org/10.1111/j.1745-6584.2005.00106.x. 10, 99

S. Mandelbrojt and L. Schwartz. Jacques Hadamard (1865-1963). *Bulletin of the American Mathematical Society*, 71:107–129, 1965. URL http://projecteuclid.org/euclid.bams/1183526393. 1

J. I. Marden. *Multivariate Statistics - Old School, Lecture Notes, University of Illinois at Urbana-Champaign*. 2013. URL http://www.istics.net/pdfs/mathstat.pdf. 133

P.R. McGillivray and D.W. Oldenburg. Methods for calculating Fréchet derivatives and sensitivities for the non-linear inverse problem: a comparative study. *Geophysical Prospecting*, 38(5):499–524, 1990. URL http://dx.doi.org/10.1111/j.1365-2478.1990.tb01859.x. 44

J.M. Melenk, K. Gerdes, and C. Schwab. Fully discrete hp-finite elements: fast quadrature. *Comput. Meth. in Appl. Mech. & Engineering*, 190(32-33):4339–4364, 2001. URL http://dx.doi.org/10.1016/S0045-7825(00)00322-4. 123

A. Ngo and R. L. Schwede. dune-gesis 1.0.0 (GEoStatistical Inversion based on Stationary problems). Nov 2014. URL http://dx.doi.org/10.5281/zenodo.12976. 8

J. Nitsche. Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 36(1):9–15, 1971. URL http://dx.doi.org/10.1007/BF02995904. 43

J. Nocedal and S. J. Wright. *Numerical Optimization, 2nd edition*. Springer Series in Operations Research. Springer, 2006. ISBN 978-0-387-40065-5. 22, 44, 128

W. Nowak. Geostatisical Methods for the Identification of Flow and Transport Parameters in the Subsurface. *PhD Thesis, Institut für Wasserbau, Universität Stuttgart*, 2005. URL `http://elib.uni-stuttgart.de/opus/volltexte/2005/2275/`. 23, 30, 31, 127

W. Nowak and O. A. Cirpka. A Modified Levenberg-Marquardt Algorithm for Quasi-Linear Geostatistical Inversing. *Adv. in Water Res.*, 27(7):737–750, 2004. URL `http://dx.doi.org/10.1016/j.advwatres.2004.03.004`. 23, 30

W. Nowak and O. A. Cirpka. Geostatistical inference of hydraulic conductivity and dispersivities from hydraulic heads and tracer data. *Water Resources Research*, 42 (8), 2006. URL `http://dx.doi.org/10.1029/2005WR004832`. 4

W. Nowak, S. Tenkleve, and O. A. Cirpka. Efficient computation of linearized cross-covariance and auto-covariance matrices of interdependent quantities. *Math. Geol.*, 35(1):53–66, 2003. URL `http://dx.doi.org/10.1023/A:1022365112368`. 3, 53, 128

D. A. Di Pietro and A. Ern. *Mathematical Aspects of Discontinuous Galerkin Methods*. Springer, 2012. ISBN 978-3-642-22980-0. 5, 66, 144

L. B. Rall and G. F. Corliss. An introduction to automatic differentiation. In M. Berz, C. H. Bischof, G. F. Corliss, and A. Griewank, editors, *Computational Differentiation: Techniques, Applications, and Tools*, pages 1–17. SIAM, Philadelphia, PA, 1996. 45

P. A. Raviart and J. M. Thomas. Mathematical aspects of finite element methods. *Lecture Notes in Mathematics*, 606:292 – 315, 1977. URL `http://dx.doi.org/10.1007/BFb0064470`. 65

W. Reed and T. Hill. Triangular mesh methods for the neutron transport equation. *Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory*, 1973. 6, 72

B. Rivière. *Discont. Galerkin Methods for Solving Elliptic and Parabolic Equations: Theory and Implementation*. SIAM, 2008. ISBN 978-0-898716-56-6. 5, 43, 66

H.J. Roos, M. Stynes, and L. Tobiska. *Robust Numerical Methods for Singularly Perturbed Differential Equations*. Springer, 2008. ISBN 978-3-540-34466-7. 3, 4

K. Roth. Soil Physics. *Lecture Notes: Institute of Environmental Physics, Heidelberg University*, 2012. URL `http://www.iup.uni-heidelberg.de/institut/forschung/groups/ts/students/sp`. 37

A. E. Scheidegger. General theory of dispersion in porous media. *Journal of Geophysical Research*, 66(10):3273 – 3278, 1961. URL `http://dx.doi.org/10.1029/JZ066i010p03273`. 39

A. Schöniger, W. Nowak, and H.-J. Hendricks Franssen. Parameter estimation by ensemble Kalman filters with transformed data: Approach and application to hydraulic tomography. *Water Resources Research*, 48(4), 2012. URL `http://dx.doi.org/10.1029/2011WR010462`. 3, 123

D. Schötzau and L. Zhu. A robust a-posteriori error estimator for discontinuous Galerkin methods for convection-diffusion equations. *Applied Numerical Mathematics*, 59(9): 2236–2255, 2009. URL `http://dx.doi.org/10.1016/j.apnum.2008.12.014`. 7, 68

R. L. Schwede, A. Ngo, P. Bastian, O. Ippisch, W. Li, and O. A. Cirpka. Efficient parallelization of geostatistical inversion using the quasi-linear approach. *Computers and Geosciences*, 44:78 – 85, 2012. URL `http://dx.doi.org/10.1016/j.cageo.2012.03.014`. 78

R.L. Schwede, W. Li, C. Leven, and O.A. Cirpka. Three-dimensional geostatistical inversion of synthetic tomographic pumping and heat-tracer tests in a nested-cell setup. *Advances in Water Resources*, 63:77–90, 2014. URL `http://dx.doi.org/10.1016/j.advwatres.2013.11.004`. 10, 102

P. Šolín. *Partial Differential Equations and the Finite Element Method*. John Wiley & Sons, 2006. ISBN 9780471720706. URL `http://dx.doi.org/10.1002/0471764108`. 143

J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis, 2nd Edition*. Texts in Applied Mathematics. Springer, 1996. ISBN 0-387-97878-X. 20

N. Z. Sun. *Inverse Problems in Groundwater Modeling*. Theory and Applications of Transport in Porous Media. Kluwer Academic Publishers, 1994. ISBN 0-7923-2987-2. 3, 21, 45, 136

H. Sutter. A Fundamental Turn Toward Concurrency in Software. *Dr. Dobb's Journal, 30(3)*, 03/2005. URL `http://www.drdobbs.com/web-development/a-fundamental-turn-toward-concurrency-in/184405990`. Accessed: 2014-08-29. 6

B.-T. Tan, C. Burstedde, O. Ghattas, J. Martin, G. Stadler, and L.C. Wilcox. Extreme-Scale UQ for Bayesian Inverse Problems Governed by PDEs. *IEEE*, 2012. URL `http://dx.doi.org/10.1109/SC.2012.56`. 31

D. Uciński. *Optimal Measurement Methods for Distributed Parameter System Identification*. CRC Press, 2004. ISBN 978-0-8493-2313-4. 44

D. Vandevoorde and N.M. Josuttis. *C++ Templates - The Complete Guide*. Addison Wesley, 2002. ISBN 0-201-73484-2. 75, 80

R. Verfürth. A posteriori error estimators for convection-diffusion equations. *Numerische Mathematik, Springer*, 80:641–663, 1998. URL `http://dx.doi.org/10.1007/s002110050381`. 68

R. Verfürth. Robust a posteriori error estimates for stationary convection-diffusion equations. *SIAM Journal on Numerical Analysis*, 43(4):1766–1782, 2005. URL `http://dx.doi.org/10.1137/040604261`. 68

L. Zhu and D. Schötzau. A robust a-posteriori error estimate for hp-adaptive DG methods for convection-diffusion equations. *IMA Journ. of Num. Analysis*, 31(3):971 – 1005, 2011. URL `http://dx.doi.org/10.1093/imanum/drp038`. 69