Enhancement of the Codes d^3f and r^3t

GRS - 292

Gesellschaft für Anlagen-
und Reaktorsicherheit
(GRS) mbH

Enhancement of the Codes d^3f and r^3t

Anke Schneider (GRS), ed.

March 2012

Remark:

This report was prepared under the contract No. 02 E 10336 with the Federal Ministry of Economics and Technology (BMWi).

The work was conducted by the Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH.

The author is responsible for the content of this report.

GRS - 292
ISBN 978-3-939355-68-7

This report is dedicated to Eckhard Fein who initiated the development of d^3f and r^3t, looked after their continuous advancement through a great many years and projects, and was finally not able to complete this one.

Keywords:
Density-driven Flow, Final Repository, Fracture Flow, Free Surface, Modelling, Solute and Heat Transport

Acknowledgement

We would like to thank all our colleagues who contributed to this final report and to the successful conclusion of the project “Weiterentwicklung der Rechenprogramme d³f und r³t” (E-DuR).

In the first place the members of the working groups of Prof. Sabine Attinger at the University of Jena, Prof. Dietmar Kröner at the University of Freiburg, Prof. Martin Rumpf at the University of Bonn, Prof. Gabriel Wittum at the University of Frankfurt and Dr. Michael Heisig and his colleagues at the Steinbeis Research Centre Technical Simulation have to be mentioned. We would like to say thank you for their excellent work, the good cooperation and their personal commitment which was far beyond the completion of their contracts.

As well we like to express our thanks to the members of the Repository Safety Research Division of GRS for performing test case simulations and their consultation.

Various colleagues have contributed to this report. These were in particular:

Chapters 1, 2:	Eckhard Fein, Anke Schneider
Chapter 3:	Jude Musuuza
Chapter 4:	Alfio Grillo, Michael Lampe
Chapter 5:	Dmitrij Logashenko
Chapters 6, 7:	Peter Frolkovič
Chapter 8:	Andreas Vogel, Arne Nägel, Sebastian Reiter
Chapter 9:	Michael Hoffer, Sebastian Reiter
Chapter 10:	Mirko Kränkel
Chapter 11:	Judith Flügge, Klaus-Peter Kröhn, Anne Püschel, Anke Schneider, Sabine Spießl
Chapter 12:	Anke Schneider

Abstract

With regard to long term safety analyses, the hazardous material which may be released from repositories in deep geological formations is thought to pass three zones before it can affect men, the near field, the far field and the biosphere. The codes d³f and r³t are focused on flow and transport modelling in the far field, i. e. the geosphere. Their development started in a period when rock salt was selected as a potential host rock for radioactive and chemo-toxic waste. The codes were focussed on modelling density-driven flow and nuclide transport in the overburden of a salt dome. For this reason they were restricted to applications in porous media where heat transport could be neglected.

Taking into account crystalline rock or mudstone as alternative possible host media, fractures and fracture systems in an otherwise porous medium must now be regarded, too. Furthermore, flow and transport have also to be modelled within the host formation. Here, the maximal warming in the surrounding of the casks is not allowed to exceed approximately 100 centigrade so that the effect of heat on water density and thus on flow and transport cannot be neglected any longer.

One difficulty in density-driven flow is to predict the stability of a model. Within this project, a stability number is derived to decide if a flow regime is in a stable or unstable state.

By the extensions of d³f and r³t presented in this report the codes are now also empowered to model heat transport. The thermohaline flow problem is described mathematically, and the three field equations to be solved in d³f are realised for two variants, the Boussinesq approximation and the complete equation system.

Modelling of porous media is complemented with the explicit modelling of fractures. Here, fractures are represented by lower dimensional structures. The finite volume discretisation is adapted accordingly.

The new option to model free surface flow provides the ability to take into account pumping wells and groundwater recharge, too.

The evolving equation system was of increased complexity. The solvers within d³f had therefore to be improved and optimized. A higher order finite volume method is introduced to improve accuracy. New filtering algebraic multigrid (FAMG) methods are developed and implemented. Additionally, the UG parallelisation concept is advanced to a flexible tool, the parallel communication layer (PCL), that enables d³f and r³t to use modern parallel computers effectively.

Pre and postprocessors are adapted where necessary.

Table of contents

1 Introduction 1

2 The project 3
2.1 State of science and technology 3
2.2 Overall objective 5
2.3 The organisation 10

3 Scaling of haline and thermohaline flow in heterogeneous media 13
3.1 Introduction 13
3.1.1 Previous stability studies 14
3.1.2 The current work 15
3.1.3 The mathematical formulations 18
3.1.4 Homogenization theory 19
3.2 The Stability Criteria 21
3.2.1 Criterion for a homogeneous medium 21
3.2.2 Stability criterion for a heterogeneous medium 23
3.3 Large-scale mixing 24
3.4 Numerical results 26
3.4.1 Results for a homogeneous medium 26
3.4.2 Extension to include dispersion 28
3.4.3 Stability studies in a heterogeneous medium 33
3.4.4 The macrodispersion coefficients 38
3.5 Discussion and summary 42

4 Thermohaline-driven flow and thermodiffusion in porous media 47
4.1 Description of the mathematical model 50
4.1.1 Balance laws 50
4.1.2 Dissipation inequality 53
4.1.3 Constitutive framework 54
4.2 Thermodiffusion 58
4.2.1 Constitutive model 59
4.2.2 Phenomenological coefficients 61
4.3 Field equations 63
4.3.1 The Boussinesq-Oberbeck approximation 64

4.3.2 The Dufour effect 65
4.4 Results and discussion 65
4.4.1 Elder problem 66
4.4.2 Evolution of a solute parcel 68
4.4.3 Mixing problem 72
4.4.4 Soret cell 73
4.5 Three-dimensional simulation of the thermohaline-driven buoyancy of a brine parcel 77
4.5.1 Mathematical model 78
4.5.2 Description of the problem and specific assumptions 80
4.5.3 Numerical simulations 83
4.5.4 Discussion and outlook 87

5 Density-driven flow in fractured media 93
5.1 Introduction 93
5.2 Model of the density-driven flow in fractured porous media 96
5.2.1 Governing equations 97
5.2.2 A change of variables 100
5.2.3 Geometric model of a single fracture 102
5.2.4 Average along the thickness 103
5.2.5 Averaged equations – Further simplifying assumptions 105
5.2.6 Balance laws at the fracture-medium interface 109
5.3 Model of contaminant transport in fractured porous medium 115
5.4 Finite-volume discretization and numerical solvers 117
5.5 Discretization grids and degrees of freedom 118
5.5.1 The finite-volume discretization 120
5.5.2 Solution of the discretized system 123
5.6 Numerical experiments 125
5.6.1 Tests in 2d 125
5.6.2 Test in 3d 129

6 Modelling of free groundwater table 131
6.1 Level set function 131
6.1.1 Mathematical description 131
6.1.2 Numerical implementation 134
6.2 Computation of groundwater flow for a fixed groundwater table 136
6.2.1 Numerical implementation 137

6.2.2 Immersed interface formulation 138
6.2.3 The pressure 140
6.2.4 The groundwater velocity field 140
6.2.5 The concentration 141
6.3 Computation of the dynamic groundwater table 141
6.3.1 Example of computations 143
6.4 Level set method for the dynamic groundwater table 146
6.4.1 Computation of the signed distance function 147
6.4.2 Computation of extrapolated speed 151
6.4.3 Computation of the advection equation for the level set function 153
6.5 Computation of transport in r³t with free groundwater table 153

7 Computation of potential flow 155

8 Numerical advances 157
8.1 Higher order finite volume schemes 157
8.1.1 Definition of a Finite Volume Scheme of Higher Order 158
8.1.2 Application to the equation for density driven flow 160
8.1.3 Numerical test example 160
8.2 Algebraic multigrid solvers for density driven flow 162
8.2.1 Introduction 162
8.2.2 Density driven flow 162
8.2.3 Algebraic Multigrid 164
8.2.4 Convergence 170
8.2.5 Effective preconditioners 179
8.2.6 Numerical Experiments 185
8.2.7 Test 3: Norderney – Development of fresh water lenses 188
8.3 Parallelisation 189
8.3.1 Introduction 189
8.3.2 Concepts 190
8.3.3 Implementation Details 191
8.3.4 Scaling Results 196

9 Preprocessing 199
9.1 Introduction 199
9.2 ProMesh 199
9.3 Grid generation 201

9.3.1 Boundary and fracture descriptions 201
9.3.2 Resolving intersections 202
9.3.3 Triangle-grid optimization 203
9.3.4 Volume geometry generation 204
9.3.5 Fracture expansion 206
9.4 Graphical user interface 208
9.4.1 Introduction 208
9.4.2 Declarative GUI programming 209
9.4.3 Visual programming 216
9.4.4 VRL applications 221

10 The postprocessor 233
10.1 Fractures 234
10.2 Visualization on free and moving boundaries 238
10.3 Visualization of sources and sinks 240
10.4 Function evaluation probing and integration 241
10.5 Visualization of stochastic data 244
10.6 Visualization of sub domains 245

11 Code verification and applications 247
11.1 Test cases with thermohaline-driven flow 247
11.1.1 Test case „1d-heat transport“ 247
11.1.2 Test case heat transfer in anisotropic porous media 259
11.2 Density-driven flow in fractured media 264
11.2.1 Test case „matrix diffusion“ 264
11.2.2 Testcase Majak 273
11.2.3 Results 287
11.2.4 3d test case with a single fracture in an inhomogeneous matrix 291
11.2.5 A realistic fracture flow model - SKB’s Task8b 296
11.3 Modelling with free groundwater table 312
11.3.1 Flow within a dam 312

12 Conclusion and outlook 317
12.1 Scaling in heterogeneous media 317
12.2 Thermohaline-driven flow 317
12.3 Flow and transport in fractured media 318
12.4 Free surface modelling and potential flow 318

12.5 Numerical advances 319
12.6 Adaption of pre- and postprocessors 319
12.7 Code verification 320
12.8 Outlook 320
12.9 Summary 321

References 323

Table of figures 347

List of tables 355

A Appendix A: Notation 357

B Appendix B: Publications 359

C Appendix C: Meetings 355

1 Introduction

To assess long-term safety of a repository for radioactive waste or of a subsurface disposal for chemo-toxic waste comprehensive system understanding and well tested and powerful numerical tools are required. These tools are used to describe the most relevant processes which play an important role for solute transport through the host rock and through the geological formations above, respectively.

In the period from October 1st, 1994 to August 31st, 1998 under the identification number 02 C 0254 6 (GSF) and later under the identification number 02 C 0465 0 (GRS) and from October 1st, 1998 to December 31st, 2003 under the identification number 02 E 9148 2 both of the computer codes d³f (distributed density-driven flow) and r³t (radionuclides, reaction, retardation, and transport) were developed /FEI 99/, /FEI 04/. These developments were funded by the Federal Ministry of Education and Research (BMBF) and by the Federal Ministry of Economics and Technology (BMWi), respectively. By means of these two computer codes it became feasible to simulate density driven flow and pollutant transport in porous media, including all interactions that are currently relevant for long-term safety assessments. They enable to handle large models with complex hydrogeological structures within convenient processing times.

Both computer codes were successfully applied in various qualification projects and applications /BIR 00/, /SCH 04/, /KES 05/, /FEI 08/, /FLU 09/. Currently they are used in different projects such as 02 E 10518 (ESTRAL), 02 E 10750 (URSEL), 02 E 10669 (KOLLORADO) and 02 E 10719 (ISIBEL).

Right from the planning stage on realistic representation of flow and transport by the developed computer code was preferred over a multitude of considered processes /FEI 91/. For the sake of efficiency only the minimum requirements – compiled as a catalogue – were therefore realised.

In doing so the explicit modelling of fractures and the modelling of heat transport consciously were set aside. This procedure only demonstrates a kind of approximation of the natural relations. In /FEI 91/ it was then found that this catalogue of minimum requirements had to be revised and upgraded as soon as numerics and hardware would allow. In the meantime numerics and hardware have been enhanced to the point where

these restrictions applied to the development of d³f and r³t need not to be adhered to any longer.

Furthermore it appeared to be desirable not to restrict the computational methods for long-term safety assessments of repositories to conventional porous media but to extend them to salt, clay, and crystalline rock. Hereunto advancing d³f and r³t was necessary. To model the density-driven flow and the associated radionuclide or pollutant transport through crystalline rock or mudstone fractures and fracture systems in an otherwise porous medium must be taken into account.

Maximal warming at the top of a salt dome hosting a repository for heat producing waste was assessed to amount to about 4 centigrade. The impact of such a temperature increase on the flow above the salt dome can be neglected as a first approximation. For repositories in crystalline rock or mudstones, however, flow and transport have to be modelled within the host formation. The maximal warming in the surrounding of the casks equals approximately 100 centigrade. In that case the influence of heat on the flow is too strong to be neglected /WAL 05/.

Initially, the code d³f was developed to model density-driven flow caused by salinization. But in crystalline rocks or mudstones and in the absence of heat producing wastes density varyations in the groundwater are not expected. In this case groundwater flow can much simpler be described as a potential flow. Including an option to simulate potential flow with d³f leads to a considerable acceleration of such simulations.

To model near-surface disposal realistically or to consider salt water intrusion problems, the fresh water lenses of islands, pumping wells or recharge, it is furthermore necessary to allow for the modelling of free groundwater surfaces.

With the above listed enhancements of d³f and r³t the range of application of both computer codes will be considerably enlarged.

2 The project

2.1 State of science and technology

The development of the computer codes d³f and r³t enabled the simulation of density driven flow and pollutant transport in porous media. These codes allow the modelling of large complex domains, accounting for all interactions relevant for the present long-term safety analyses, within acceptable computing times.

The code development in cooperation with universities involving PhD students assured the employment of the most recent numerical methods from the beginning up to now. Special attention was paid to an exact modelling of the physical processes and a consistent discretisation of the time dependent, non-linear partial differential equation system, /OSW 98/, /JOH 97/, /FRO 97/, /GEI 03/, /KNA 96/, /KNA 98/, /JOH 04/, /JOH 06/, /JOH 06a/.

Both codes use unstructured grids and finite volume discretizations. They are based on the software system UG /BAS 94/, a toolbox developed at universities to solve coupled systems of partial differential equations. The development of UG started in 1990. Since then it is widely used for a large range of application and is permanently extended and advanced. UG sets standards in the field of effective solvers up to now.

As effective solvers of a linear equation system d³f and r³t combine geometric and algebraic multigrid algorithms (amg) with BiCGStab solvers /HAC 85/, /WIT 89/, /WIT 92/, /BAS 94/, /BAS 00/, /FEU 03/, /JOH 04/ /JOH 05/, /JOH 05a/, /JOH 07/, /NAE 08/. An additional reduction of computing effort was achieved by adaptive grid refinement, controlled by a-posteriori error estimators, see /THI 98/, and a time step control. d³f and r³t are completely parallelised and use a sophisticated load balancing to use the processing power as effective as possible /BIR 98/, /LAN 05/. They can be run on LINUX PCs, workstations, PC clusters and on massively parallel computers.

Beside d³f and r³t the codes FEFLOW /WAS 10/, MODFLOW /HAR 00/, NAMMU /SER 03/ und RockFlow/Open GeoSys /KOH 09/, /WAN 09/ are used for the modelling of density driven flow and transport.

The commercial finite element code FEFLOW is well established in groundwater modelling, especially because it comes with a comfortable graphical user interface and

some useful tools such as water balancing. FEFLOW can handle free surface flow, fractures, density driven flow and contains an inverse modelling. Recently it was enhanced by amg methods and parallelised, too. On the other hand in the case of geometrical complex 3d-models it may be a disadvantage that FEFLOW works with so called slices, manually created grid layers, instead of real 3d-meshes. To model salt transport, FEFLOW makes use of the Boussinesq approximation, i. e. one part of the nonlinearity remains neglected. This strategy simplifies the model, and it is normally sufficient for a salt content in the range of seawater concentration. In the range up to saturated brine it reduces considerably the reliability of the results, as shown in /JOH 03/.

Many groundwater modellers use the finite difference code MODFLOW. It has a relatively inflexible grid structure. CG-based methods are used as solvers. Thus the code can easily be modified and extended. Meanwhile various modules exist for density-driven flow, transport and other purposes. Nevertheless MODFLOW is not the tool to handle complex models with large numbers of nodes.

The FORTRAN-based finite element code NAMMU is used in the assessment of long-term safety. It is able to compute density-driven flow including heat transport and nuclide transport in porous media. It is not applicable to complex models.

Open GeoSys/RockFlow is an open source finite element code. Beside density-driven flow and transport it is able to model multiphase flow in porous and fractured media, and geo-mechanical influences may be regarded. RockFlow works on adaptive grids too, and it can be coupled with other codes. CG-based algorithms are used as solvers.

The computer programs d3f and d³f are based on the simulation system UG /BAS 00/, which to this day possesses a unique position among the simulation programs as it is a general parallel and adaptive solver for coupled systems of partial differential equations. The development of UG already started in 1990 and initiated a series of further developments, /DEA 03/, /BAS 05/, and various others. None of these other developments, however, comes up to the level, functionality, implementation range, and scope of application of UG. The treatment of realistic models possessing a sufficiently fine resolution requires a grid with 10^8 or more unknowns. These can only be employed reasonably when all possibilities to reduce complexity and to increase efficiency are used. Examples of this are adaptive multi-grid methods, /BAS 94/, and their parallelisation. UG sets standards for this.

Up to now, the standard methods to discretise differential equations in UG are finite volume methods. During the recent years, new formulations for discretisation were developed. Amongst these are most notably the discontinuous Galerkin methods, /ODE 98/. They provide the possibility to handle the order of basis functions flexibly and with that to include them into an adaptive concept as well. Such methods have not yet been formulated for the density-driven flow and therefore provide for this and for the transport problems linked to it an interesting alternative. Preliminary work for developing fast solvers was already done, /JOH 05/, /JOH 05b/.

2.2 Overall objective

Aim of this project is the advancing of the computer programs d^3f and r^3t (see Tab. 2.1) by introducing

- the explicit consideration of fractures,
- the simultaneous consideration of the heat transport,
- the consideration of free water tables (phreatic flow),
- the option of modelling potential flows, and
- the adaption of pre- and post-processors to the new tasks.

In addition to this the scaling behaviour and the stability of the thermohaline flow in heterogeneous media are examined to derive criteria for stability.

Furthermore, new numerical algorithms are developed and applied, e. g. filtering algebraic multi-grid methods (FAMG) /WAG 00/, /NAE 08/ and the discontinuous Galerkin method.

By these extensions the codes d^3f and r^3t are empowered to model three-dimensional density-driven flow in case that the density is not only influenced by the dissolution of salt but also by heat transport. Additionally, the modelling of porous media is complemented with the explicit modelling of fractures. This modelling of porous/fractured media is necessary for the long-term safety analysis if the host rocks are mudstone and crystalline rocks.

The possibility to model free surface flow provides the ability to consider pumping wells and groundwater recharge.

Altogether, within long-term safety analyses of repositories the application field is enlarged to other host rocks than rock salt. At the same time, it is also extended to geothermic questions.

Tab. 2.1 Modelling with d^3f and r^3t, respectively
(hitherto possible: green; hitherto possible, but ineffective: orange; applied extension: white, orange)

Host formation / **Processes/media**	**Salt**	**Plastic clay**	**Crystalline rock**	**Mudstone**
density-driven flow (d) potential flow (p)	d	p	p	p
fractured (f)	f	not necessary	f	f
porous (p)	p	p	p	p
free surface	+	+	+	+
heat transport	not necessary	+	+	+

Further development of the codes d^3f and r^3t was a significant improvement for applications already covered by previous versions, and also opens up a wide range of new applications.

2.2.1.1 The flow model d^3f and the transport model r^3t

The computer codes d^3f for flow and r^3t for contaminant transport simulations were especially developed to model three-dimensional large regions with complex geometries over long periods of time. The hydrogeology of the modelled area can thereby comprise strong heterogeneities and anisotropies. The basic requirements for use were:

- The porous media are fluid saturated.
- The aquifer systems are confined.
- Both the porous media and the fluid are incompressible.

The flow model d³f solves the equation system of density-driven flow, and it provides the flow field for r³t. The transport code r³t includes all interactions that are currently relevant for long-term safety assessments, such as decay, sorption, precipitation, complexation, and colloidal transport /FEI 04/. Both codes - d³f and r³t - were also applied in combination to various test cases /FEI 08/.

The computer codes d³f and r³t offer interactive graphical preprocessors and a post-processor. The latter is identical for both program packages and is based on the GRAPE software /RUM 92/, /GRA 99/. It is designed to analyse and to plot the results of the simulations.

The preprocessors help preparing the input data, which itself consists of two different parts: data describing the hydrogeolocical model, and data controlling the numerical algorithms. The model data are subsumed in five and seven different files, respectively. The controlling of the simulator is managed by script files which can be altered by means of an interactive graphical user interface.

The simulators of d³f and r³t are based on the numerical library UG (“unstructured grids”) /BAS 97/. The UG library comprises robust solvers for numerical simulations on hierarchical grid structures. These multigrid solvers are used successfully in d³f as well as in r³t. All numerical algorithms applied for solving density-driven flow or radionuclide transport problems are based on finite volume methods.

For a detailed description of the state of d³t and r³t at the beginning of this project see /FEI 99/ and /FEI 04/.

2.2.1.2 Scaling of haline and thermohaline flow in heterogeneous media

The hydraulic properties of different hydrogeological units are typically varying over multiple length scales. These heterogeneities cause difficulties in the upscaling of hydraulic parameters. On the other hand, in long term safety analysis the consideration of large length and time scales is indispensable.

Up to now stability analyses of haline and thermohaline flow only existed for homogeneous media. In this project, stability analysis is combined with a multiscale approach to be able to predict the beginning of instabilities and to establish robust stability criteria for heterogeneous media.

Stable and unstable haline and thermohaline flow has to be scaled by different methods. Stable flow can be treated using asymptotic scaling methods. Herefore, existing results for weakly heterogeneous media are adapted to strongly heterogeneous and anisotropic media.

On unstable flows for the first time modern flexible filter methods are applied. These methods allow the scaling of pre-asymptotic processes. Effective scale-dependent parameters were to be found.

2.2.1.3 Explicit consideration of fractures

At the beginning of this project the application of d³f and r³t was restricted to porous media. Now, special approaches for the integration of fractures were developed and implemented.

The aperture of a fracture is usually negligible compared with the model scale. On the other hand, these structures cause a huge numerical effort if the fracture is geometrically treated as a volume. Therefore, structures of reduced dimension had to be established within d³f and r³t. The discretisation had to be adapted. Flow and transport within the fractures work independently of the processes within the sourrounding matrix. Interface conditions for the fracture matrix interaction and average methods over the fracture width had to be developed.

A special finite volume method was developed using so called degenerated elements (see chapter 5). Grid generators and refinement algorithms had to be adapted. The solvers were improved bei introducing filtering algebraic multigrid (FAMG) methods and a new parallelisation concept.

2.2.1.4 The heat transport

In the previous version of d³f a system of two equations was solved, the mass balance of the fluid-phase and the mass balance of the solute. To integrate heat transport into d³f the energy balance of the mixture as a whole had to be introduced as a third equation. Different thermodynamical concepts were developed, investigated and compared with thermohaline flow and thermodiffusion models known from literature. Theoretical

and experimental analyses were performed to determine the scope of validity of the equations.

Two variations of the energy equation had to be regarded, the complete equation and the Boussinesq approximation.

The accuracy of the solution was significantly improved by the development of a finite volume method of higher order. Formulations for the dependency of the hydraulic parameters on pressure, temperature and salinity were compiled in a separate report /KRO 10/.

2.2.1.5 Consideration of free groundwater surfaces and potential flow

In the original concept for d³f and r³t the influence of groundwater recharge or pumping wells on the water table was not considered. Now the immensely increased available computational power allowed to extent the code capabilities to a free groundwater surface. This could be done in two ways, regarding a moving numerical grid or a moving front within a fixed grid. Here, the second possibility was chosen. The moving groundwater surface is approximated by a level set function.

To improve the performance for models not involving transport processes a feature was created that allows the computing of only potential flow within one step.

2.2.1.6 Adaptation of pre- and postprocessors

To be able to visualise and analyse data on fractures and on free surfaces, the postprocessor had to be extendet, too.

For this purpose new visualisation concepts working on lower dimensional structures had to be developed and implemented. Additionally, the robust visualization and analysis of different types of data on implicitely described surfaces had to be realized.

The existing tool set for the interactive local data extraction, based on parameterized cutting tools and integration methods in space and time, had to be extendet considerably.

A new graphical user interface was developed. Additionally, the interactive graphical tool ProMesh was adopted for geometrical model building and grid generation, especially for models containing fractures.

2.3 The organisation

Goethe Center for Scientific Computing (G-CSC),
Universität Frankfurt
Kettenhofweg 139, 60325 Frankfurt
Prof. Gabriel Wittum
Dr. Alfio Grillo
Dr. Michael Lampe
Dr. Arne Nägel
Sebastian Reiter
Sabine Stichel
Andreas Vogel
Christian Wehner

Institut für Geowissenschaften der Universität Jena
Burgweg 11, 07749 Jena
Prof. Sabine Attinger
Dr. Jude Musuuza
Dr. Florin Radu
Katharina Ross

Mathematisches Institut der Universität Freiburg
Hermann-Herder-Str.10, 79104 Freiburg i. Br.
Prof. Dietmar Kröner
Mirko Kränkel

Institut für Numerische Simulation der Universität Bonn
Endenicher Allee 60, 53115 Bonn
Prof. Martin Rumpf
Dr. Martin Lenz
Dr. Nadine Olischläger
Sascha Tölkes

Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH
Fachbereich Endlagersicherheitsforschung
Theodor-Heuss-Str. 4, 38122 Braunschweig
Dr. Eckhard Fein
Anke Schneider
Dr. Judith Flügge
Dr. Klaus-Peter Kröhn
Anne Püschel
Dr. Sabine Spießl

and as RD-Contractor of GRS

Steinbeis-Forschungszentrum „Technische Simulation”
Bussardweg 6, 75446 Wiernsheim-Iptingen
Dr. Michael Heisig
Dr. Dmitrij Logashenko
Dr. Peter Frolkovič
Dr. Alexander Fuchs
Michael Hoffer

3 Scaling of haline and thermohaline flow in heterogeneous media

3.1 Introduction

The migration of leachate at waste repositories, the flow of saline water in coastal aquifers and geothermal systems with significant temperature gradients show patterns that differ significantly from passive tracers. The flow and solute transport in these systems are induced by the spatial variations in fluid density, arising from salinity or temperature differences. Such systems are generally referred to as density-driven.

The differences between density-driven systems and passive tracers arise from the coupling between the fluid flow and solute transport processes in the former. Typically, the systems can show erratic interactions between regions of the fluid with different salinities or temperatures. The interactions cause the breakdown of the sharp interface when a denser fluid overlays and displaces a less dense, or when a cooler fluid displaces a warmer one. The fingering phenomenon then occurs and the system is said to be unstable.

Instabilities can be explained physically by taking into account the various forces that act on density-driven fluids at rest or in motion. They may individually have stabilising or destabilising effects to the system. A stable system is in general attained when the external forces like inertia, viscous stresses and buoyancy balance and a state of minimum energy is reached in which no state of lower energy is accessible. A perturbed system would move back to this stable low-energy state. In contrast, the system shows unstable behaviour if states of lower energy are accessible and an infinitesimal perturbation causes it to evolve to a different state with lower energy.

Viscosity dissipates the energy of a disturbance and stabilises the system. For this reason, any bounded flow is stable if viscosity is large enough. It can also diffuse momentum, which can make some flows like parallel shear flow unstable though the same are stable in an inviscid fluid. Thermal conductivity and molecular diffusion smooth out temperature and concentration gradients respectively and so have stabilising effects. Buoyancy forces on the other hand have a destabilising effect if a denser liquid lies on top of a less dense one. Boundaries constrain the development of instabilities and the closer they are the more stable a system becomes. Boundaries can

however result in stronger shear in boundary layers, which can lead to instabilities when diffused out by viscosity.

In general, an unstable configuration results when a denser fluid overlays a less dense one as was already mentioned above. However, such a system can still remain stable if the mobility (viscosity) term does not favour finger formation or when the velocity, normal to the direction of finger development is high such that the instabilities that form have no time to grow into fingers; or when mixing smoothes them out.

In addition to physical instabilities, numerical instabilities may be introduced by inappropriate numerical schemes. The corresponding mathematical models often provide numerically non-unique solutions, which according to /OLD 95/ and /FRO 01/ arise from insufficient grid refinement and extrapolation of the initial conditions if the grid is not aligned with the sides of the domain. In mathematical analysis the first step is to determine the original state of the system, which is referred to as the basic state. In density-driven flows, the basic state depends on the velocity, pressure and solute concentration. The numerical solution must satisfy the describing equations as well as the applicable boundary conditions.

Physically, one wishes to know whether the basic state can be observed or not. If it is disturbed even so slightly, the perturbation decays away or grows infinitely in magnitude. Decaying growth is actually an evolution to another steady-state, which thermodynamically means another lower-energy state.

3.1.1 Previous stability studies

Early linear stability studies are documented in /CHA 88/ and how some e. g. /WOO 62/ studied the stability of vertical miscible displacements in homogeneous media and concluded that the interface could be stable or unstable depending on the wave numbers. Others like Perrine and Gray /PER 66/ wrongly concluded that instabilities could not form in homogeneous media, while subsequent ones like /SET 77/ showed that instabilities could form in homogeneous media provided mixing effects were sufficiently small.

More recent investigations through laboratory experiments can be found in e. g. /SCH 90/ and numerical simulations and mathematical analysis in e. g. /COS 90/ and

/SCH 97 among others. The stability of density-driven flow systems can be found in /WOO 62/, /SCH 90/, /SCH 97/, /COS 90/, /KRE 03/, and /HEL 05/. The stability of gravity-driven systems has also been studied and is documented in literature.

3.1.2 The current work

The homogenization theory ideas introduced in /HEL 05/ were extended to derive the stability criterion and develop the expressions for the macrodispersion coefficients.

3.1.2.1 The homogeneous medium

For homogeneous media, the small-scale dispersion is the only stabilising mechanism. A homogeneous medium was initially considered with the effects of dispersion neglected. It was established that systems are stable at the large scales only when stable at small scales.

With dispersion included, it was ascertained that for a given density contrast perturbation wavelengths up to a critical value are stable. A mixing zone evolves as a result of variable pore size and orientation and the size of the zone controls the perturbation wavelengths that grow into fingers.

The width of the mixing zone increases with the prevailing diffusion/dispersion, in accordance with the scale-dependency of dispersion documented in the work of Dagan /DAG 88/. In a homogeneous medium, flow is stable at the small scale only when the spreading (and therefore the mixing zone) is big enough to prevent the instabilities from growing into fingers. Even then instabilities with big enough wavelengths can still develop into fingers. The stability of a system is thereby controlled by the perturbation wavelength and the width of the mixing zone, also called the transition zone in some literature.

The investigation of dispersive effects was limited to downward flow that is solely driven by density effects. The stability of vertical density-driven flow was initially studied by Lord Rayleigh and later by Elder, who used the Rayleigh and other dimensionless numbers to investigate the onset of convection in a system heated from below. Other researchers have used the haline equivalent of the thermal Elder problem with a more saline fluid overlaying a less saline one to study convection patterns caused by

salinity stratification. A **solutal** Rayleigh number is defined for the equivalent haline problems and the density contrast, domain size, density contrast and diffusion are chosen in such a way to make it equal to the original number derived by Elder for the thermal problem. Many of the researchers reported differences in fingering patterns depending on the level of grid refinement and the density contrast. For sufficiently refined grids, /OLD 95/, /DIE 02/ and /FRO 01/ observed comparable finger evolutions.

To ease comparison with previous research, the stability number initially derived for a homogeneous medium without dispersion was reformulated in form of a solutal Rayleigh number. Henceforth, Rayleigh number refers to the solutal Rayleigh number and the Elder problem refers to the haline equivalent of the classic thermal Elder problem.

The Rayleigh number for the Elder problem is approximately 400. Only molecular diffusion is taken into account, which is erroneous considering the 20 % density contrast, which causes evident convection patterns. The entire domain height (150 m for the Elder problem) is also used in the computation. Alternative formulations for the Rayleigh number that take dispersion into account can be found in e. g. /SCH 97/ and /DIE 02/. /SCH 97/ further proposed the use of a characteristic length instead of the entire domain size. Earlier, Bues and Aachib /BUE 91/ had suggested the use of the mixing zone width as the characteristic length.

The Rayleigh number was computed in this work with the effects of dispersion included and the characteristic length taken as the mixing zone width instead of the domain size. Significantly smaller density contrasts are also used, which results in Rayleigh numbers much smaller than 400 for the Elder problem.

1. For the Elder problem, /DIE 02/ identified 3 regimes and correspondingly 2 critical Rayleigh numbers
2. The nearly diffusive regime for Rayleigh numbers up to the first critical, $Ra_{c_1} \approx 4\pi^2$
3. A convective regime with stable numerical solutions for $4\pi^2 \leq Ra \leq Ra_{c_2}$, with the second critical number Ra_{c_2} in the range 240 – 300
4. The convective regime with unstable solutions for $Ra > Ra_{c_2}$.

Johannsen in /JOH 02a/ independently showed that the number of fingers evolved from one for very small Rayleigh numbers to three (in some solution branches) at Ra > 300, with the latter coinciding with the second critical number in /DIE 02/ for the onset of the unstable convective regime.

The study attempted to quantify the critical wavelength and the mixing zone and used the transition in the number of fingers and the three regimes to infer system stability.

3.1.2.2 The heterogeneous medium

The medium heterogeneity increases the dispersivity against the local value. This is in fact a compounded effect of the small-scale dispersion at pore scale and large-scale mixing caused by the medium heterogeneity. The effect of the heterogeneity is thus similar to dispersive mixing.

Therefore, the effective dispersivity can be used in the expression for the homogeneous medium with dispersion. A new stability number can then be derived in terms of the old and the heterogeneous medium properties, namely the heterogeneity variance and the correlation length.

It is hypothesised that high variances and small correlation lengths cause intense mixing that smoothes out all instabilities and prevents them from escalating into fingers. Low variance and large correlation lengths on the other hand allow instabilities to grow into fingers. From the foregoing, big variances and correlation lengths are expected to result into a reduction and increase in the number of fingers, respectively.

3.1.2.3 Large-scale transport behaviour

Large-scale transport in heterogeneous media was also studied using the macrodispersion tensor derived using homogenization theory. The tensor was symmetric with zero off-diagonal elements while the leading diagonal elements showed scale effects as documented in the works of Dagan, e. g. /DAG 88/. Most literature pertaining to macroscopic mixing is limited to passive tracers. An attempt was made to reproduce the transport behaviour documented for passive tracers and extended the techniques to density-driven systems.

Passive tracers were “emulated” by setting the density and viscosity effects to zero. Favourable density contrasts were then investigated by arbitrarily choosing positive stability numbers and studying the trend in the coefficients. Much as these scenarios were not of much interest, the evolution of the coefficients gave useful information: the longitudinal and transverse coefficients respectively decreased and increased with increasing stability, with the longitudinal coefficient approaching an asymptotic value. This means that any stabilising variable like heterogeneity variance was expected to produce a reduction and an increase in the longitudinal and transverse coefficients, respectively.

The longitudinal coefficient increased with increased unfavourable density contrast. It will be shown later that for moderate unfavourable density contrasts (with corresponding negative stability numbers) asymptotic longitudinal coefficients could still be obtained. These physically indicated the range of unfavourable density contrasts that are stabilised by medium heterogeneity. With further increase in the density contrast, coefficients that grow indefinitely with time were obtained. This was consistent with the conclusions from previous researchers e. g. /WEL 91/ where unfavourable densities gave rise to very large longitudinal mixing coefficients. Additionally, increasing the correlation length resulted in increased longitudinal coefficients, hence a reduction in system stability, while the heterogeneity variance had the reverse effect.

3.1.3 The mathematical formulations

Presented here are the formulations used in the study. The system of equations used to describe variable-density systems are the flow, transport and Darcy equations.

$$\begin{aligned}
&\frac{\partial(\phi\,\rho(\omega))}{\partial\hat{t}} + \nabla_X \cdot (\rho(\omega)\mathbf{u}) = 0 \\
&\frac{\partial(\phi\,\rho(\omega)\omega)}{\partial\hat{t}} + \nabla_X \cdot (\rho(\omega)\omega\mathbf{u} - \phi\,\rho(\omega)\mathbf{D}\nabla_X\omega) = 0 \\
&\mathbf{u} = -\frac{\mathbf{k}}{\mu(\omega)}(\nabla_X p - \rho(\omega)\mathbf{g})
\end{aligned} \tag{3.1}$$

In the above equations, $\omega\,[-]$ is the solute mass fraction, $\rho(\omega)\,[kg\cdot m^{-3}]$ the fluid density, $\phi\,[-]$ the porosity, $\mathbf{D}\,[m^2\cdot s^{-1}]$ the diffusion/dispersion tensor, $\mathbf{X}, \hat{t}$ the respective space and time variables, $\mathbf{u}\,[m\cdot s^{-1}]$ the Darcy velocity, $\mathbf{k}\,[m^2]$ the intrinsic

permeability tensor, $\mu(\omega)[Pa \cdot s]$ the dynamic viscosity, $p[Pa]$ the pressure and $\mathbf{g}[m \cdot s^{-2}]$ the gravitational acceleration. Sources and sinks, temperature and sorption were neglected. The system is defined on $J \times \Omega$, where the domain $\Omega \subset R^2$ and the time $J = (0, T)$, with T the end time.

The hydrodynamic dispersion tensor is implemented according to Scheidegger's law

$$\mathbf{D} = D_m \mathbf{I} + (\alpha_{\|} - \alpha_{\perp}) \frac{\mathbf{v} \otimes \mathbf{v}}{\| \mathbf{v} \|} + \alpha_{\perp} \| \mathbf{v} \| \mathbf{I}, \tag{3.2}$$

where **I** is the identity matrix, D_m the molecular diffusion coefficient, $\alpha_{\|}$ and $\alpha_{\perp}$ the respective longitudinal and transverse dispersion lengths and v the ground water velocity.

To solve the above density-driven system, the dependencies of density on salinity and viscosity have to be specified beforehand. Linear state dependencies were used

$$\rho(\omega) = \rho_0 (1 + \alpha\omega) \tag{3.3}$$

$$\mu(\omega) = \mu_0 (1 + \beta\omega), \tag{3.4}$$

where ρ_0, μ_0 are the density and viscosity of pure water, and α, β are coefficients defining the maximum relative density and viscosity, respectively.

3.1.4 Homogenization theory

Homogenization theory is an up-scaling method with advantages over spatial averaging techniques. The biggest advantage over spatial averaging is that its results can undergo the rigorous scrutiny of mathematical proofs. Essentially, two scales are assumed: the mesoscale l at which processes occur and the macroscale L at which processes are observed. The scales are related through the quantity $\varepsilon = l / L$. Related to the two scales are two dimensionless spatial and temporal variables. The scales are assumed to be well-separated making the quantity ε approach zero.

Quantities that show spatial and temporal variations like pressure and mass fraction can then be written as series expansions in the parameter ε. For example the mass fraction in the transport equation can be written as:

$$\omega^{\varepsilon}(\mathbf{x},\mathbf{y},t,\tau)=\omega_0(\mathbf{x},t)+\varepsilon\omega_1(\mathbf{x},\mathbf{y},t,\tau)+O(\varepsilon^2)\,, \tag{3.5}$$

where $\omega_0(\mathbf{x},t)$ is the large-scale mass fraction. The function $\omega_1(\mathbf{x},\mathbf{y},t,\tau)$ is assumed to be periodic in **y** (of period 1). The spatial and temporal derivative can also be written in terms of those variables:

$$\nabla_X=\frac{1}{L}\left(\nabla_{\mathbf{x}}+\frac{1}{\varepsilon}\nabla_{\mathbf{y}}\right), \tag{3.6}$$

$$\frac{\partial}{\partial\hat{t}}=\frac{D_{\|}}{L^2}\left(\frac{\partial}{\partial t}+\frac{1}{\varepsilon^2}\frac{\partial}{\partial\tau}\right). \tag{3.7}$$

When the spatial and temporal derivatives are applied to the 2-scale transport equation and the terms with the same powers of ε collected, one obtains the following three equations:

1. The compatibility condition

$$\rho(\omega)\frac{\partial\omega_0}{\partial\tau}+\frac{L}{D_{\|}}\mathbf{v}\cdot\nabla_{\mathbf{y}}\omega_0-\nabla_{\mathbf{y}}\cdot\mathbf{D}^*\nabla_{\mathbf{y}}\omega_0=0\,, \tag{3.8}$$

where L is the macroscopic length taken as the domain size in mean flow direction, **v** the total velocity and $\mathbf{D}^*=\rho(\omega)\mathbf{D}/D_{\|}$ the modified hydrodynamic dispersion tensor. The equation is a statement of the independence of macroscopic quantities from small scales.

2. The small-scale equation describes the variation of the solute on the meso scale:

$$\rho\frac{\partial\omega_1}{\partial\tau}+\frac{L}{D_{\|}}\mathbf{v}\cdot\nabla_{\mathbf{y}}\omega_1+\frac{L}{D_{\|}}\tilde{\mathbf{v}}\cdot\nabla_{\mathbf{x}}\omega_0-\nabla_{\mathbf{y}}\cdot\mathbf{D}^*\nabla_{\mathbf{y}}\omega_1=0\,, \tag{3.9}$$

Where $\mathbf{v},\tilde{\mathbf{v}}$ are the total and fluctuating velocities, respectively. The small-scale equation is also called the cell problem in literature.

3. **The large-scale equation** that contains the homogenized tensor, in this case the macrodispersion tensor.

$$\rho\frac{\partial\omega_0}{\partial t}+\frac{L}{D_{\|}}\mathbf{v}_0\cdot\nabla_{\mathbf{x}}\omega_0-\nabla_{\mathbf{x}}\cdot\mathbf{D}^{\text{eff}}\ \nabla_{\mathbf{x}}\omega_0=0 \tag{3.10}$$

The macrodispersion tensor is defined by $\mathbf{D}^{\text{eff}}=\mathbf{D}^{*}-L/D_{\|}\cdot\overline{\tilde{\mathbf{v}}\otimes\boldsymbol{\chi}^{\omega}}$, where χ^{ω} is the solution to the cell problem. The small- and large-scale equations can be used to derive the stability criterion and the expressions for the macrodispersion coefficients, respectively.

3.2 The Stability Criteria

3.2.1 Criterion for a homogeneous medium

This was derived from the small-scale equation on the explicit assumption that the velocity fluctuations depended on the fluctuations in the mesoscale mass fraction only.

$$\tilde{\mathbf{v}}(\mathbf{q},\tau)=\mathbf{M}(\mathbf{q})\omega_1(\mathbf{q},\tau), \tag{3.11}$$

where $\mathbf{q}$ is the Fourier space variable and $\mathbf{M}(\mathbf{q})$ the contribution of solute to the velocity fluctuations.

With dispersion neglected, the small-scale equation reduced to the form $\partial\omega_1/\partial\tau+b\omega_1=0$ with the solution $\omega_1(t)=\omega_1(0)e^{-bt}$. Depending on the sign of b, the solution would decay to zero or grow indefinitely. It was referred to as the stability number and it is a function of density, viscosity, flow velocity and concentration gradients.

By assuming a divergence-free velocity, modifying the velocity and considering only up to linear terms, $\mathbf{M}(\mathbf{q})$ could be evaluated and the stability criterion derived.

The criterion took the following forms for flow aligned parallel and orthogonal to gravity:

$$\Lambda_p=\frac{L}{D_{\|}}\left[(\alpha-\beta)v_0+\alpha v_0^{g}\right] \tag{3.12}$$

$$\Lambda_o = \frac{L}{D_{\|}}\left(v_0^g G_2 + v_0^p G_1\right)\left[\left(\alpha-\beta\right)+\left(\frac{a^2}{a^2+1}\right)\alpha\right] \tag{3.13}$$

α, β are respectively the maximum relative density and viscosity coefficients; v_0 the total downward velocity; v_0^p the pressure-driven velocity component; v_0^g the velocity driven velocity component; $a = -v_o^g / v_0^p$; G_1, G_2 the respective concentration gradient components in the directions orthogonal and parallel to gravity. The flow was stable when the stability numbers were positive.

Extension to dispersive effects

The inclusion of dispersion focused on how the mixing zone controls the wavelengths that can pass into instabilities. To that end, an expression was derived that expressed the stability number in terms of the old number and a dispersive contribution

$$\Lambda_p^* = \Lambda_p + \frac{D_{\perp}\zeta^2}{D_{\|}\lambda^2} \quad . \tag{3.14}$$

$D_{\|}, D_{\perp}$ are the longitudinal and transverse dispersion coefficients, respectively, λ the perturbation wavelength and ζ the characteristic length. The characteristic length was initially required to preserve the dimensionless form of the stability number.

Following the ideas in /KEM 94/, the characteristic length could be expressed in terms of the dispersivities. Mathematical fitting eventually led to

$$\Lambda_p^* = \Lambda_p + \frac{\left(\alpha_{\perp}^3 \alpha_{\|}\right)^{\frac{1}{2}}}{\lambda^2}. \tag{3.15}$$

In order to determine the range of stable wavelengths, the inflow region was perturbed with the following sinus function

$$\omega = A_0\left[1+\sin\left(\frac{2\pi\left(x-x_0\right)}{\lambda}\right)\right] \tag{3.16}$$

A_0 was set to 0.5 so that the maximum mass fraction was constrained to 1.0 when the sinus function attained its maximum value, and 1 was added to avoid unphysical negative values of the salt when the sinus function was at the minimum value. $x_0 = 0.4$ is the left abscissa of the inflow region.

3.2.2 Stability criterion for a heterogeneous medium

The extension of the previously derived criterion for a homogeneous medium to include heterogeneity effects is presented here. A log-normally distributed permeability field with a Gaussian auto-covariance function was used:

$$w_f(x) = \sigma_f^2 \exp\left(-\sum_{j=1}^{2} \frac{|x_j|^2}{2\lambda_j^2} \right). \tag{3.17}$$

σ_f^2 is the variance and x_j, λ_j are the components of the space and correlation length in direction *j,* respectively.

In the extension of the stability criterion to heterogeneous fields, effective dispersion lengths have to be used instead of the local values. This is to account for the increase induced by medium heterogeneity.

The effective values, in the form of the sums $\alpha^{\text{eff}} = \alpha + \delta(\alpha)$, with α and $\delta(\alpha)$ the local dispersivities and their changes due to the heterogeneous medium are substituted in the previous stability number. Multiplying out the terms and neglecting the products of the changes and expressing the heterogeneous-medium-induced longitudinal dispersivity as the product of the variance and correlation length (see /GEL 83/, /GEL 93/), the expression below is obtained:

$$\Lambda_p^{**} = \Lambda_p^* - \frac{\left(n^2 - 1\right)\left(\alpha_\parallel^3 \alpha_\perp\right)^{\frac{1}{2}}}{\lambda_v^{\,2}} + \frac{3\sigma^2 \left(\alpha_\parallel \alpha_\perp\right)^{\frac{1}{2}}}{2\lambda_v} \tag{3.18}$$

where *n* is the number of critical homogeneous medium perturbation wavelengths λ_{crit} in a heterogeneous medium critical correlation length $\lambda_{v,\text{crit}}$.

The above expression comprises of the previous stability number Λ_p^* and the properties of the heterogeneous medium σ^2 and λ_v. The expression is consistent with physical expectations because the system stability increases and decreases with the heterogeneity variance and correlation length respectively. Unfavourable density contrasts in a homogeneous medium ($\Lambda_p^* < 0$) can be stabilised by inclusion of medium heterogeneities.

3.3 Large-scale mixing

Large-scale mixing was studied using the macrodispersion tensor that was derived from the large-scale transport equation (itself derived via homogenization theory in /HEL 05/. The individual tensor elements initially evaluated to functions containing diffusion and averaged products of the solution to the small-scale equation and the mesoscale velocity fluctuations, with the former expressed as a definite time integral. The mesoscale velocity was rewritten with the contribution from the fluctuations in the medium heterogeneity explicitly included (see derivations in /MUS 10/):

$$\tilde{\mathbf{v}}(\mathbf{q},\tau) = \mathbf{M}(\mathbf{q})\omega_1(\mathbf{q},\tau) + \mathbf{L}(\mathbf{q})\tilde{k}(\mathbf{q}). \tag{3.19}$$

$\mathbf{L}(\mathbf{q})$ is the contribution of medium heterogeneity to velocity fluctuations and $\tilde{k}(\mathbf{q})$ the permeability fluctuations. This was then substituted into the effective dispersion tensor in the homogenized equation:

$$\mathbf{D}^{\text{eff}} = \mathbf{D}^* - \frac{L}{D_{\|}}\overline{\tilde{\mathbf{v}} \otimes \boldsymbol{\chi}^{\omega}} \tag{3.20}$$

The solution to the cell problem was obtained by first assuming the salt mass fluctuations to be of the form $\omega_1(\mathbf{q},\tau) = \boldsymbol{\chi}^{\omega}(\mathbf{q},\tau)\cdot\mathbf{G}$ (see /CIO 99/ for details). This was then substituted into the small-scale equation that had been transformed into Fourier space and heterogeneities included in the velocity fluctuations to give

$$\chi_n^{\omega}(\mathbf{q},\tau) = \kappa\int_0^{\tau} d\tau' L_n(\mathbf{q})\tilde{f}(\mathbf{q})\exp\left(-\left(i\frac{L}{\rho D_{\|}}\mathbf{v}\cdot\mathbf{q} + \frac{\mathbf{D}^*\mathbf{q}\cdot\mathbf{q}}{\rho} + \frac{\Lambda}{\rho}\right)\tau'\right), \tag{3.21}$$

with $\kappa = (LG_n)/(D_{\|}\rho)$. Multiplying out the product and carrying out the averaging operation on the tensor elements was essentially the evaluation of integrals in space

and time: over the entire space and finite times. By expressing the respective terms as Gaussian functions and neglecting diffusion, the integrals could be conveniently evaluated with the software MAPLE®: analytically up to the time integrals and then completely by numerical techniques. The evaluation gave a symmetric tensor with zero off-diagonal elements, while the leading diagonal elements depicted the scale dependency mentioned in the preceding paragraphs. The leading-diagonal elements evaluated to the following forms:

$$\begin{aligned} D_{11}^{eff} &= \theta \int_s^\infty d\eta \int_0^\infty ds \frac{3\pi}{4\kappa^3 \left(\xi^2+\eta\right)^{5/2}} \\ &\times \Big[\left(2\Lambda_p^2 (1+\eta)\right)\left(2\, erf(C) - erf(B) - erf(D)\right) \\ &+ \kappa^2 \left(2\,\Lambda_p \tau \left(erf(C) - erf(D)\right) + erf(C) - erf(B)\right)\Big) \\ &- 2\Lambda_p \kappa \sqrt{(1+\eta)}(1 - 2F + G)\Big]. \end{aligned} \tag{3.22}$$

$$\begin{aligned} D_{22}^{eff} &= \theta \int_s^\infty d\eta \int_0^\infty ds \frac{\pi}{4\kappa^5 \left(\xi^2+\eta\right)^{3/2} (1+\eta)^{3/2}} \\ &\times \Big\{ A^*\, 2\,\Lambda_p^2 (1+\eta)\Big[\left(2\,\Lambda_p^2 (1+\eta)\right)\left(erf(D) - 2\, erf(C) + erf(B)\right) \\ &+ \kappa^2 \left(\left(\tau\,\Lambda_p + 1\right) erf(D) - \left(2\,\tau\,\Lambda_p + 5\right) erf(C) + 3\, erf(B)\right)\Big] \\ &+ 2\,\Lambda_p \kappa (1+\eta)\Big[2\,\Lambda_p^2 (1+\eta)(G - 2F + 1) + \kappa^2 (G - 3F + 2)\Big] + \kappa^5 \tau F \Big\}. \end{aligned} \tag{3.23}$$

$$A = \sqrt{\pi}\, \exp\left(\frac{\Lambda_p^2 (1+\eta)}{\kappa^2}\right), \qquad B = \frac{\Lambda_p (1+\eta)}{\kappa},$$

$$C = \left(\frac{\left(\tau\,\kappa^2 + 2\,\Lambda_p + 2\,\Lambda_p \eta\right)}{2\,\kappa\sqrt{1+\eta}}\right), \qquad D = \frac{\Lambda_p \eta + \tau\kappa^2 + \tau\Lambda_p}{\kappa\sqrt{1+\eta}},$$

$$F = \exp\left(-\frac{\tau\left(\tau\kappa^2 + 4\Lambda_p + 4\Lambda_p \eta\right)}{4(1+\eta)}\right), \qquad G = \exp\left(-\frac{\tau\left(\tau\kappa^2 + 2\Lambda_p + 2\Lambda_p \eta\right)}{1+\eta}\right),$$

$$A^* = A\sqrt{1+\eta}.$$

As previously mentioned, the stability of the systems could be inferred from the temporal evolution of the longitudinal coefficient. Additionally, the stabilising or destabilising behaviour of different variables could be studied using the responses in the coefficients induced by changes in those variables.

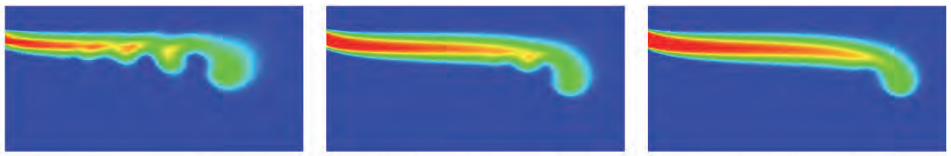
3.4 Numerical results

3.4.1 Results for a homogeneous medium

The stability number for flow orthogonal to gravity was used. In the following, the criterion is tested for the effects of density, viscosity and flow velocity within a homogeneous medium without dispersion. The parameters were adopted from /SCH 97/.

3.4.1.1 The Schincariol problem

The problem in /SCH 97/ was used to test the criterion. Initially an adequate grid and time stepping had to be established that ensured numerical solutions free from artefacts without having to apply upwind techniques. The procedure outlined in /SCH 97/ involved having to gradually reduce the mesh and time steps until a stable solution was obtained. Similar Peclet and Courant numbers were used and the results, which were comparable to those documented in /SCH 97/ are shown below.



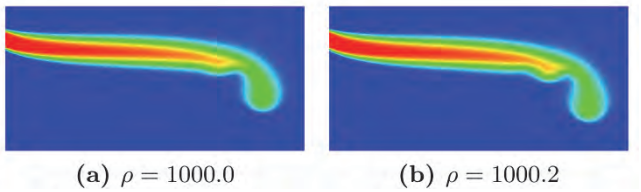
(a) Pe=16.5, Cr=0.22 (b) Pe=8.40, Cr=0.40 (c) Pe=2.32, Cr=1.10

Fig. 3.1 The Schincariol results

Having obtained the stable numerical solution and an optimum grid refinement, physical instabilities were induced by varying the various physical parameters and their effects studied by making the relevant changes in the stability criterion. The results are presented in the following sections. The grid and time stepping in Fig. 3.1(c) was taken as standard on which all simulations were based.

Density-driven systems are not rotation-free. The rotation of the local velocity vectors cause a recirculation cell to emerge at the tip of the plume. Solute is trapped and transported inside the cell where it remains at all times. The ‘nose’ therefore does not count as a finger and unstable behaviour is when other undulations develop additionally.

Even then, the decision regarding which configurations were stable remained somehow subjective. Fig. 3.2 gives two simulations that were predicted as stable and unstable, respectively.



(a) $\rho = 1000.0$ (b) $\rho = 1000.2$

Fig. 3.2 Stable or unstable?

The differences in the finger patterns are not very pronounced but when allowed to evolve for a longer time however, the finger in Fig. 3.3(b) had more replications and travelled through a bigger vertical distance.

It is therefore hypothesised that there exists a horizontal plane that demarcates stable from unstable systems. The salt fronts in unstable systems travels beyond that plane.

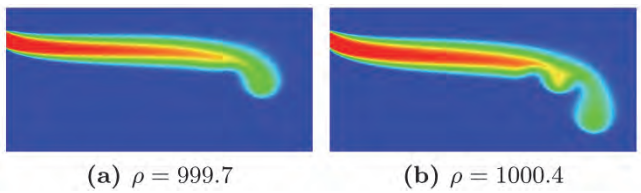
3.4.1.2 Effects of density

The table below gives the stability numbers computed at a viscosity of $1.006\times10^{-3}\,\mathrm{Pa\cdot s}$, $v_0^p = 2.75\times10^{-6}\,\mathrm{m\cdot s^{-1}}$ and various density contrasts.

Tab. 3.1 The Effect of Density

Max. Density [kg m^{-3}]	998.70	1000.0	1000.2	1000.4
$\alpha\ \left(\times10^{-3}\right)$	1.5027	1.8032	2.0036	2.2040
$\Lambda_o\ \left(\times10^{-3}\right)$	5.8377	2.2813	-0.0898	-2.4606

The Figures below show the simulations at the two extreme densities in the table.



(a) $\rho = 999.7$ (b) $\rho = 1000.4$

Fig. 3.3 Homogeneous Medium: Density Effects

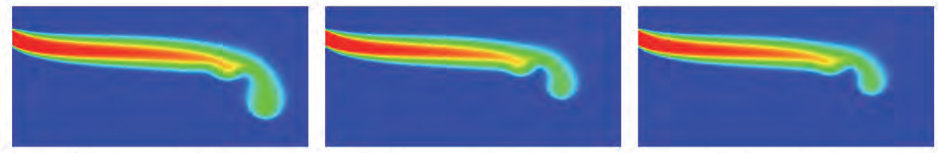
3.4.1.3 Effects of viscosity

The table below show the effect of increasing viscosity at a density contrast that had been predicted as unstable.

Tab. 3.2 Homogeneous Medium: Viscosity Effects

Max. Viscosity [10^{-3} Pa s]	1.006	1.200	1.250
$\beta\left(\times 10^{-3}\right)$	3.992	197.605	247.505
$\Lambda_o\left(\times 10^{-3}\right)$	-0.09	114.50	144.07

The figure below shows the simulations at the tabulated viscosities.



(a) $\mu = 0.001006$ (b) $\mu = 0.001200$ (c) $\mu = 0.001250$

Fig. 3.4 Homogeneous Medium: Viscosity Effects

The second finger became less pronounced and the vertical distance was significantly reduced. The finger in Fig. 3.4(a) could be the only one that went beyond the demarcation plane mentioned previously.

The criterion was also tested for the effects of velocity. It made reasonable predictions at low velocity but failed with increasing flow. The failure was attributed to dispersion that was up to that point not included.

3.4.2 Extension to include dispersion

The same simulation parameters from /SCH 97/ were used again. A configuration with vertically downward flow was used and the maximum density was reduced to 998.5 kg m^{-3} to ensure moderate fingering. The two relevant factors in the study of dispersion effects are the perturbation wavelength and the mixing zone width. The

critical perturbation wavelength was obtained by perturbing the inflow region with sinus functions of various wavelengths.

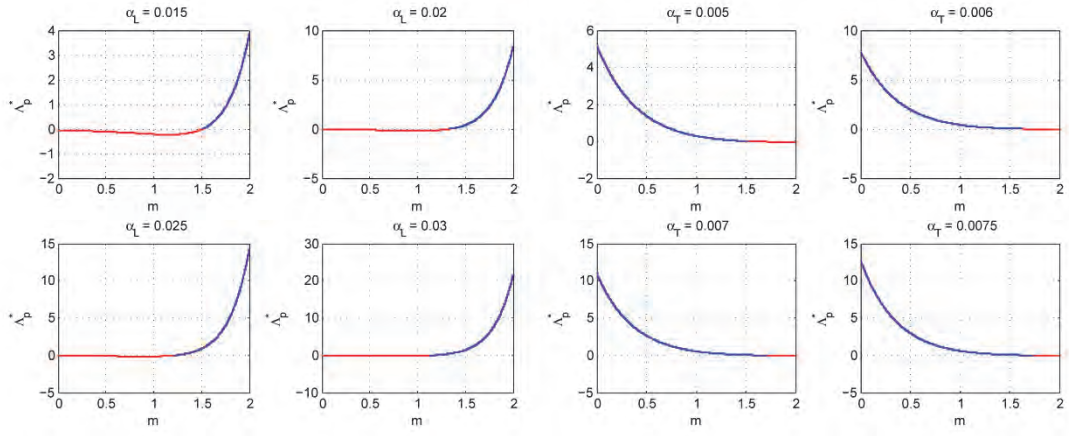
The critical wavelength was when the third finger started developing $\lambda_{\text{crit}} \approx 2.0 \times 10^{-3}\,\text{m}$ was obtained.

By using physical considerations to constrain the range of values that the exponents of dispersivities could take, the expression for the characteristic length was tested and fitted. This was done by utilising the change in sign of the stability number at the onset of convection. To that end, the stability number could be written as

$$\Lambda_p^* = \Lambda_p + \frac{\alpha_{\|}^{m} \alpha_{\perp}^{2-m}}{\lambda} \tag{3.24}$$

The change in stability was accomplished by using the critical wavelength and varying *m* for various $\alpha_{\|}$ and fixed $\alpha_{\perp}$ and then varying the transverse while fixing the longitudinal dispersivity.

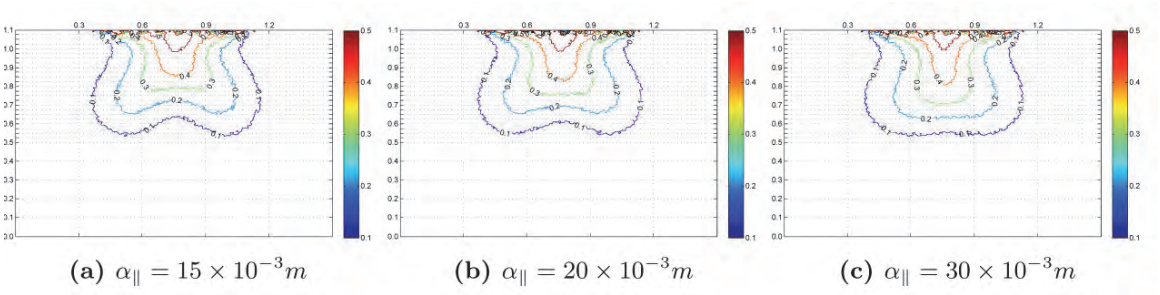
The figures below show the ranges of *m* corresponding to stable and unstable configurations obtained by varying the longitudinal and transverse dispersivities respectively. The blue and red portions of the curves correspond to the stable and unstable configurations respectively.



(a) $\Lambda_p^\star$ for various m and $\alpha_{\|}$ (b) $\Lambda_p^\star$ for various m and $\alpha_{\perp}$

Fig. 3.5 Fitting the mixing zone width

By overlapping the two regions, the value of *m* could be approximated as 1.5. The figure below shows simulations at the longitudinal dispersivities at which stability transition occurred. Simulations with the transverse dispersivity also showed the transition from two to one finger.



(a) $\alpha_{\|} = 15 \times 10^{-3} m$ (b) $\alpha_{\|} = 20 \times 10^{-3} m$ (c) $\alpha_{\|} = 30 \times 10^{-3} m$

Fig. 3.6 The fitted longitudinal dispersivities

Taking a wavelength slightly bigger that that at which the shift was observed as λ_{crit}, the derived stability criterion was used to study the effects of density and dispersivity as given in the following.

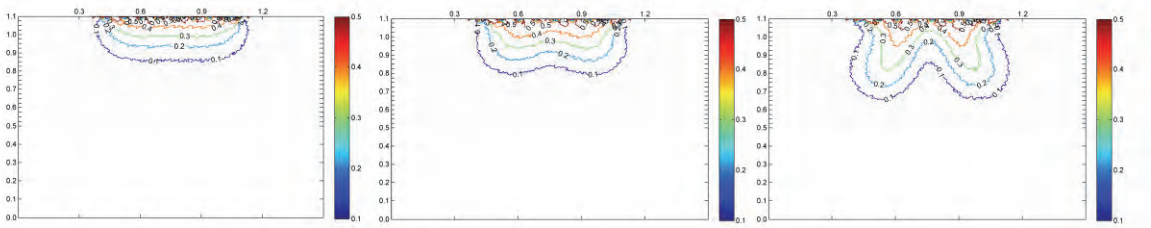
3.4.2.1 Density effects

The following section is the test of the derived criterion for the effects of density. In the table are the stability numbers computed at various densities with and without dispersion effects.

Tab. 3.3 Dispersion extension: density

Max density [kg m^{-3}]	998.25	998.30	998.35	998.40	998.50
Λ_p	3.881	0.769	-0.571	-1.317	-2.121
Λ_p^*	4.026	0.091	-0.426	-1.172	-1.976

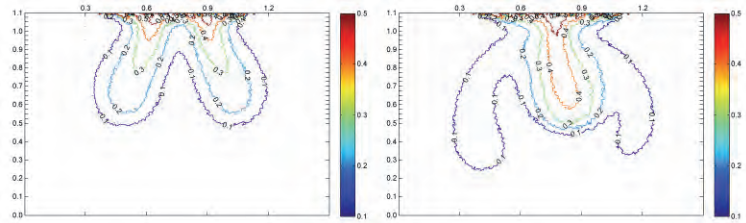
The following figures are simulations at different density contrasts to show the onset of stable convection.



(a) $\rho_{max} = 998.25kg \cdot m^{-3}$ (b) $\rho_{max} = 998.30kg \cdot m^{-3}$ (c) $\rho_{max} = 998.35kg \cdot m^{-3}$

Fig. 3.7 Density effects: onset of convection

The growth of the second finger (the onset of convection) in the simulations was reasonably predicted by the change in the sign of the stability number from the criterion. The following figures are simulations at higher density contrasts to show the transition into the unstable convective regime i. e. the growth of the third finger.



(a) $\rho_{max} = 998.40kg \cdot m^{-3}$ (b) $\rho_{max} = 998.50kg \cdot m^{-3}$

Fig. 3.8 Density effects: onset of unstable convection

The change from the stable to the unstable convective regime was indicated by the growth of the third finger in simulations and further reduction in the stability numbers. Although the range over which the transition occurred could not be pinpointed, it is envisaged that it was over a specific range of stability numbers.

3.4.2.2 Dispersivity effects

In the following, the results of investigations regarding the longitudinal and transverse dispersivities, both of which were found to be stabilising, are presented. By choosing a density very close to the onset of convection, a transition from two to one finger was achieved by increasing either dispersivity. Now higher density contrasts are investigated.

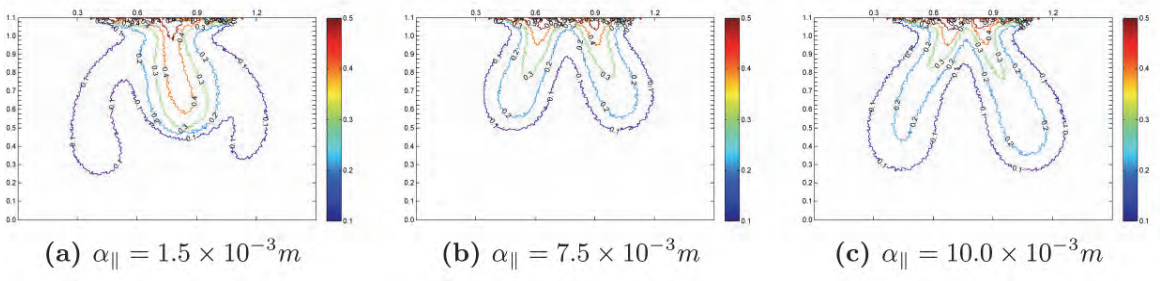
3.4.2.2.1 The longitudinal dispersivity

The table below gives the stability numbers computed at the reference parameters and various longitudinal dispersion lengths.

Tab. 3.4 Dispersion extension: longitudinal dispersivity

$\alpha_{\parallel} \times 10^{-3}\mathrm{m}$	1.5	7.5	10.0
Λ_p^*	-1.976	-1.761	-1.153

The stabilising effect is manifested through the increase of the stability number down the table. The simulations at the tabulated dispersivities capture the stabilisation through a reduction in the number of fingers as shown in the figures below.



(a) $\alpha_{\parallel} = 1.5 \times 10^{-3} m$ (b) $\alpha_{\parallel} = 7.5 \times 10^{-3} m$ (c) $\alpha_{\parallel} = 10.0 \times 10^{-3} m$

Fig. 3.9 Effects of longitudinal dispersivity

The transition from three to two fingers occurred over the range $-1.976 < \Lambda_p^* < -1.761$.

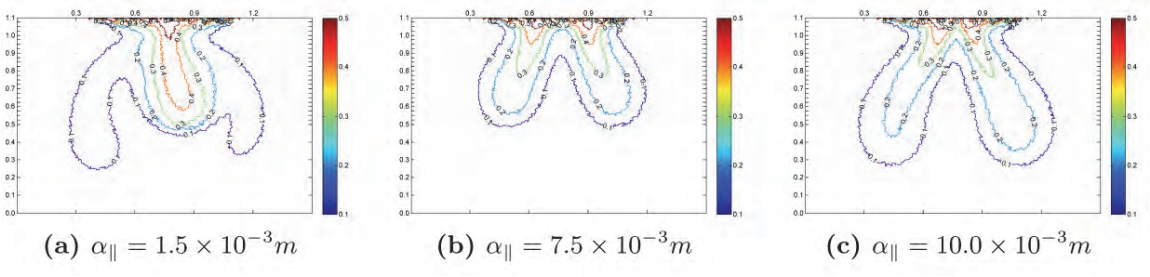
3.4.2.2.2 The transverse dispersivity

The stability numbers computed at various transverse dispersivities are given in the table below.

Tab. 3.5 Effects of transverse dispersivity

$\alpha_{\parallel}$ [10^{-4} m]	1.0	10.0	30.0
Λ_p^*	-1.976	-0.734	-0.111

The increase of the stability numbers again signify stabilisation, which is matched by a reduction in the number of fingers in the simulations below.



(a) $\alpha_{\|} = 1.5 \times 10^{-3} m$ (b) $\alpha_{\|} = 7.5 \times 10^{-3} m$ (c) $\alpha_{\|} = 10.0 \times 10^{-3} m$

Fig. 3.10 The transverse dispersivity effects

Both dispersivities caused an increase in the stability numbers and correspondingly a reduction in the number of fingers from three to two. The effect of the longitudinal dispersivity however seemed to be more than that of the transverse dispersivity, as the exponents in the proposed expression predicted.

It is plausible that if the dispersivities were sufficiently large, the regime could become nearly diffusive with a single finger. However, due to the limitations of the selected domain size, the dispersivities could not be increased enough to obtain that scenario.

3.4.3 Stability studies in a heterogeneous medium

This part of the stability studies will be presented into two: the extension of the previously derives stability criterion to medium heterogeneity effects and the analysis of the evolution of the macrodispersion coefficients to study large-scale transport behaviour.

3.4.3.1 The stability criterion for a heterogeneous medium

The effects of varying the density, dispersivities, variance and correlation length on stability can then be realised in the criterion as well as the evolution of fingers in numerical simulations. First a reference set of simulation parameters was obtained against which changes in all variables could be based.

The correlation length plays the role played by perturbation wavelengths in homogeneous media, namely controlling the instabilities that grow into fingers. If the correlation length is gradually increased, a cut off value is obtained where fingering is observed in the system. The figures below show the simulations at $\rho_{\max} = 998.5\,\mathrm{kg\,m^{-3}}$, $\alpha_{\|} = 1.5\times10^{-3}\,\mathrm{m}$, $\alpha_{\perp} = 1\times10^{-4}\,\mathrm{m}$, $\sigma^2 = 0.30\,\mathrm{m}^2$ and various correlation lengths.

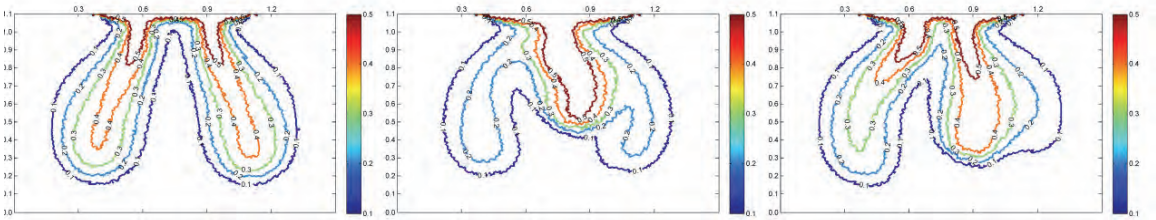


Fig. 3.11 The critical correlation length

The correlation length at which the third finger appeared was taken as the approximate critical value. The table below additionally shows the stability numbers computed at the various correlation lengths.

Tab. 3.6 The effect of the correlation length

$\lambda_v(\times10^{-3})\,\mathrm{m}$	5.0	7.5	10.0	20.0
Λ_p^{**}	-1.529	-1.859	-1.975	-2.083

In the following, the general stabilising behaviour of medium heterogeneity will be shown by achieving a reduction in the number of fingers to two at a density contrast that produced three fingers in a homogeneous medium. This was achieved through an appropriate choice of the heterogeneity variance. Afterwards, the critical correlation length ($\lambda_v = 7.5\times10^{-3}\,\mathrm{m}$) was stabilised by increasing the dispersivities and variance.

3.4.3.2 Density effects

Tab. 3.7 contains the computed stability numbers (at $\sigma^2 = 0.6\,\mathrm{m}^2$) and the corresponding simulations at the various density contrasts.

Tab. 3.7 The stabilising effect of heterogeneities

Max. density [kg m^{-3}]	998.4	998.5	998.6	998.7
Λ_p^{**}	-0.893	-1.699	-2.125	-2.397

The increase in the new stability numbers against the old ones reflects the stabilisation induced by the medium heterogeneity. The figure below shows the simulations at some selected density contrasts.

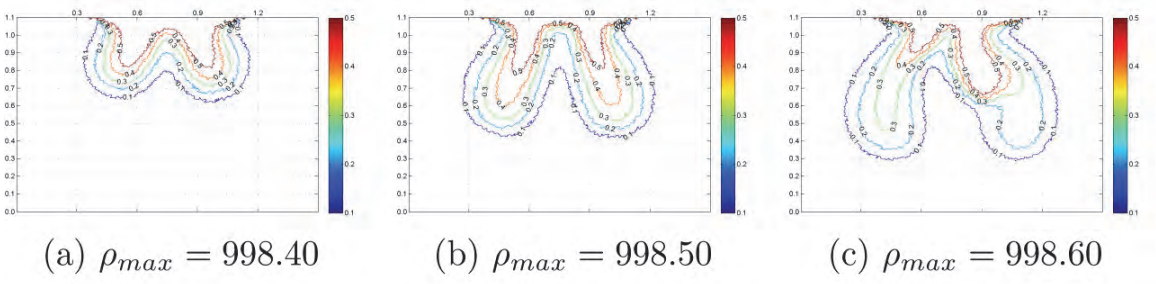


Fig. 3.12 The stabilised range of densities

The onset of the unstable convective regime occurred at a density of 998.6 instead of 998.5 that was for a homogeneous medium. That was due to mixing induced by the heterogeneities. Mixing reduces the spectrum of perturbations with long-enough wavelengths to grow into fingers i. e. it smoothes out the instabilities and a bigger density contrast is required to restore the spectrum wavelengths to values capable of finger formation.

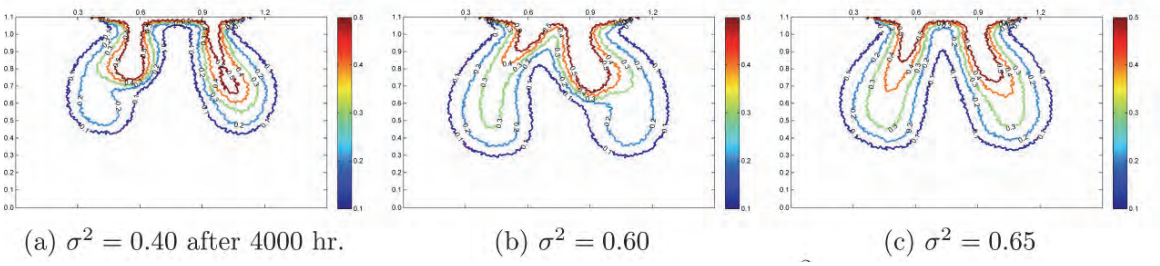
3.4.3.3 Effects of the heterogeneity variance σ^2

Increasing the heterogeneity variance means including more heterogeneity values in the distribution, which increases more heterogeneous, increases mixing and so stabilises. An increase in variance is therefore expected to increase the stability number and reduce the number of fingers in numerical simulations. The table below shows the stability numbers computed at various variances, in which the expected trend is shown.

Tab. 3.8 Effect of variance

Variance σ^2 [m²]	0.400	0.600	0.650
Λ_p^{**}	-2.200	-2.125	-2.108

Below are simulations at various variances. The stabilising behaviour of the heterogeneity variance reduces the number of fingers from three to two and the two fingers are shorted at a higher variance.



(a) $\sigma^2 = 0.40$ after 4000 hr. (b) $\sigma^2 = 0.60$ (c) $\sigma^2 = 0.65$

Fig. 3.13 Stabilising effect of variance

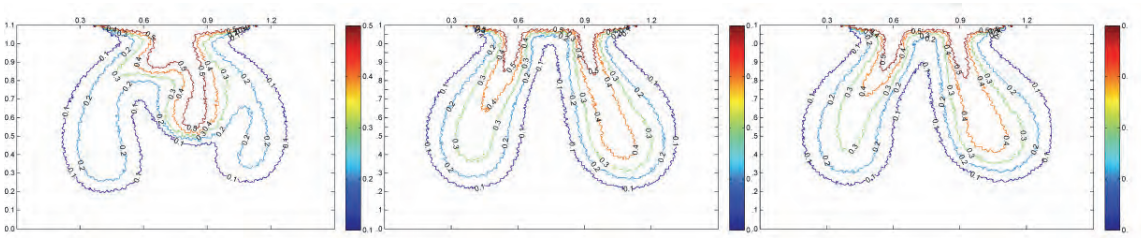
3.4.3.4 The longitudinal dispersivity

The effect of the longitudinal dispersivity was also investigated. The stabilising effect is shown by the increasing stability numbers down the table.

Tab. 3.9 Effect of longitudinal dispersivity

$\alpha_{\|} \times 10^{-3}$ m	1.5	7.5	10
Λ_p^{**}	-1.77	-1.58	-1.21

The evolution of fingers at various longitudinal dispersivities is shown in the figure below.



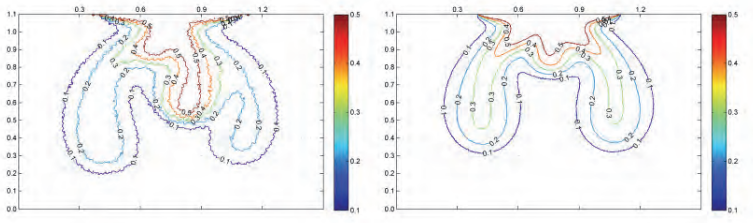
(a) $\alpha_{\|} = 1.5 \times 10^{-3}$ (b) $\alpha_{\|} = 7.5 \times 10^{-3}$ (c) $\alpha_{\|} = 10.0 \times 10^{-3}$

Fig. 3.14 Effect of longitudinal dispersivity

The stabilising effects of variance and longitudinal dispersivity and the destabilising effects of correlation length and density that are shown in the simulations could be captured by the stability criterion. The transition from two to three fingers does not occur at a specific stability number but from the considered variables, the transition occurred between -1.744 and -1.774.

3.4.3.5 Medium anisotropy

The figure below shows the effect of increasing the anisotropy of the medium. It was defined as the ratio of the horizontal to the vertical correlation length. By this definition, increasing the anisotropy could be achieved by either increasing the horizontal correlation length or reducing the vertical correlation length. In either case, transport in the horizontal direction is boosted at the expense of the vertical direction, which stabilises the system. No stability numbers could be computed for this because medium anisotropy is not included in the stability criterion.



(a) $\xi = 1,7000hr$ (b) $\xi = 100,7000hr$

Fig. 3.15 Effect of the medium anisotropy

3.4.4 The macrodispersion coefficients

The large-scale transport behaviour was studied by using the macrodispersion tensor in the large-scale transport equation derived via homogenization theory. The derivation of the tensor resulted in a symmetric tensor with zero off-diagonal elements. The longitudinal and transverse coefficients respectively read:

The temporal evolution of the leading-diagonal coefficients in the tensor was then studied and how the evolution is affected by modifications in the density contrast, heterogeneity variance, correlation length and medium anisotropy. By setting the density and viscosity effects to zero $\left(\Lambda_p^* = 0\right)$, passive tracers could be emulated.

The evolution of the coefficients for such tracers for various arbitrarily-chosen anisotropy ratios can be found in Dagan (1988). Dagan defined the anisotropy ratio as λ_h / λ_v and considered flow orthogonal to gravity. He obtained longitudinal (horizontal) coefficients that were not affected by changes in anisotropy and died off at large times and transverse (vertical) coefficients that approached asymptotes at large times and reduced with increasing medium anisotropy.

The longitudinal coefficient reduced with increasing anisotropy and approached an asymptote while the transverse hardly changed and grew to zero at large times. This behaviour is consistent with the results in /DAG 88/.

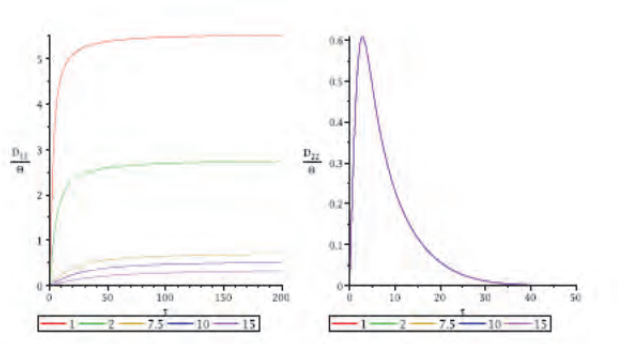
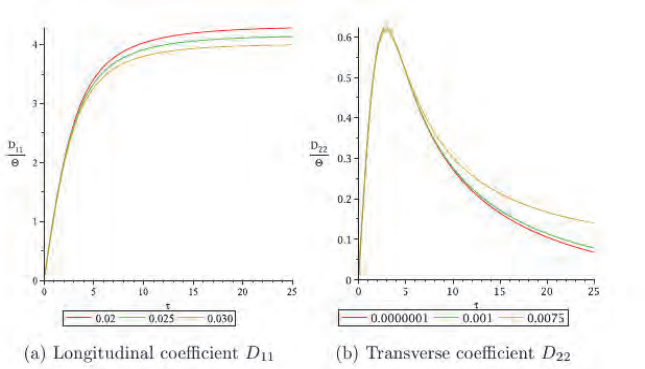


Fig. 3.16 Passive tracers

The effects of density and viscosity were then introduced. One would be interested to know how the coefficients respond to changes in the stability of the system. To achieve this, arbitrarily chosen positive stability numbers were used to compute the coefficients and their evolution over time noted.

3.4.4.1 Favourable density contrasts

The longitudinal coefficient reduced with increasing stability numbers while the transverse coefficient increased with increasing stability.



(a) Longitudinal coefficient D_{11} (b) Transverse coefficient D_{22}

Fig. 3.17 Favourable density contrasts

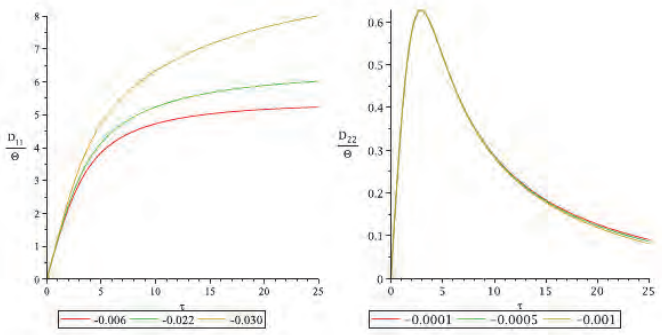
This can be explained by examining the factors that increase systems stability e. g. a reduction in the density contrast, which reduces the gravity effects. One could also look at an increase in medium heterogeneity as the cause of increased stability. Heterogeneous mixing promotes solute transport in the transverse direction thus the increased coefficient.

3.4.4.2 Unfavourable density contrasts

The behaviour depicted in the foregoing indicates that variables that stabilise the system should eventually lead to respective reductions and increases in the longitudinal and transverse coefficients. Conversely, destabilising variables should cause respectively increase and reduction in the transverse and longitudinal coefficients.

3.4.4.2.1 Density effects

The effect of density was studied by selecting density contrasts that resulted into negative stability numbers. Using these in the expressions, the evolutions of the coefficients could be studied.



(a) Longitudinal coefficients D_{11} (b) Transverse coefficients D_{22}

Fig. 3.18 Unstable density contrasts

It was noted that stability numbers close to zero produced asymptotic longitudinal coefficients. The asymptotic behaviour was however lost as the density contrasts increased, eventually becoming unbounded. The bounded coefficients signify the range of density contrasts that are unstable in a homogeneous medium but can be stabilised by medium heterogeneity. The unbounded coefficients indicate density contrasts that cause very intense fingering, which cannot be stabilised by heterogeneities. The prevailing wavelengths are very long and cannot be damped by heterogeneous mixing.

The transverse coefficient on the other hand reduces with increasing density contrasts. It is possible that it approaches an asymptote after long times but this could not be fully investigated.

3.4.4.2.2 Correlation length

Varying the correlation length has the effect of changing the medium heterogeneity in an inversely. In these studies the vertical correlation length was increased, which effectively reduced the heterogeneity in the vertical direction. The reduced heterogeneity boosted transport in that direction and resulted in increased longitudinal coefficients. This in turn indicates a reduction in the system stability.

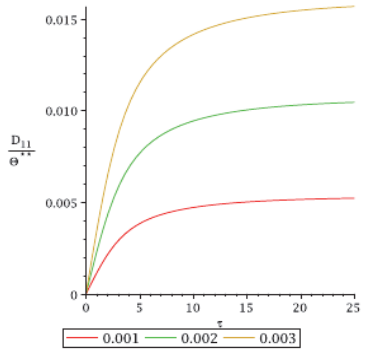


Fig. 3.19 Effect of the correlation length

3.4.4.2.3 Heterogeneity variance

The heterogeneity variance has the opposite effect to the correlation length, namely increasing the medium heterogeneity. The increased mixing results in increasing transverse coefficients, thus the stabilising behaviour.

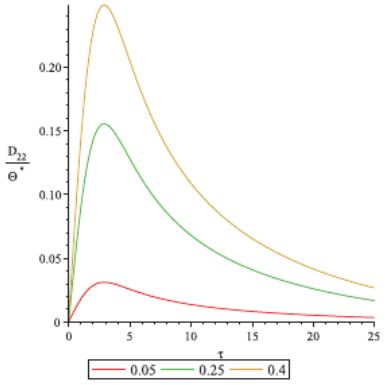
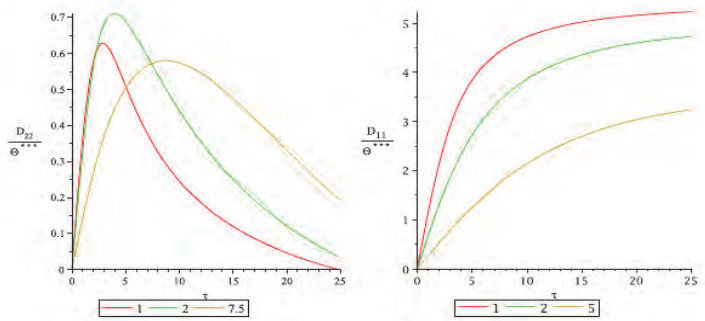


Fig. 3.20 Effect of the heterogeneity variance

3.4.4.2.4 Medium anisotropy

The anisotropy was defined as the ratio of the horizontal to vertical correlation lengths. Increasing would require either increasing the horizontal or reducing the vertical correlation lengths. Either way, transport in the horizontal direction is boosted at the expense of the vertical direction. This leads to increasing and decreasing transverse and longitudinal coefficients respectively.



(a) Transverse coefficient D_{22} (b) Longitudinal coefficient D_{11}

Fig. 3.21 Effect of medium anisotropy

3.4.4.2.5 Dispersivities

In the derivation of the formulas for the dispersion coefficients, infinite Peclet numbers were assumed. This resulted in the term containing the dispersivities going to zero. The assumption could not be relaxed to study finite Peclet numbers because of computer resource limitations.

3.5 Discussion and summary

A stability criterion was successively derived for density-driven systems via homogenization theory without the Oberbeck-Boussinesq assumption. For systems aligned orthogonal to gravity a stability number that was a function of density and viscosity contrasts, flow velocity and concentration gradients could predict the onset of fingering. The criterion however performed poorly with increasing flow velocity, a fact attributed to missing dispersion effects.

As a result of omitting the Oberbeck-Boussinesq approximation, timely stability predictions could be achieved. The studies in which the approximation had been invoked (see /MUS 09/ for comparison) predicted stability transition at much higher density contrasts.

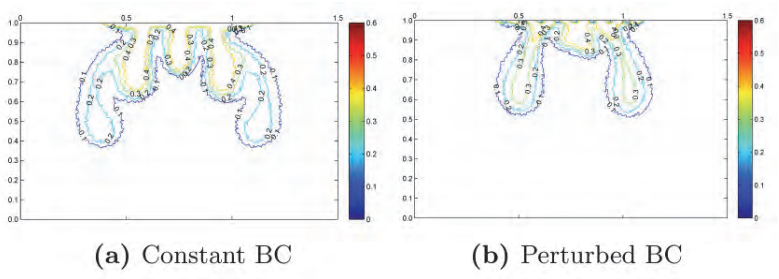
The criterion was extended to include dispersion and was subsequently tested on an Elder-type vertical system. The change in sign of the stability number (expressed in form of a Rayleigh number) then indicated the onset of stable convection. With increasing density contrast, the system transited into the unstable convective regime,

which was indicated by further reduction in the stability number and the appearance of three fingers in numerical simulations.

The stability number is expressed as a function of the perturbation wavelength and the mixing zone width. The mixing zone width was derived and fitted according to physical considerations while perturbations with different wavelengths were obtained by imposing a sinus function as a boundary condition at the salt inflow region.

The stability regimes could be inferred from the number of fingers present: stable diffusive, stable convective and unstable convective for one, two or three fingers respectively. Stabilising variables like dispersivity and viscosity caused a reduction in the number of fingers e. g. from three to two. This meant that the usual notion of unconditional instability of vertical Elder-type was probably due to neglecting the stabilising dispersion effects while computing the Rayleigh number, and considering very high density contrasts at that.

The perturbation of the inflow region caused asymmetry in the concentration front. Fig. 3.22 shows simulations with identical parameters but with a constant and perturbed boundary condition. Asymmetric behaviour in perturbed boundary conditions was also reported by /MAR 81/.



(a) Constant BC (b) Perturbed BC

Fig. 3.22 Asymmetry caused by perturbing inflow region

The criterion was extended to heterogeneous media and an expression that predicted stabilising and destabilising effects of variance and correlation length respectively, was derived. By selecting appropriate variance and correlation lengths, the growth of the third finger could be shown to occur at a higher density contrast than in the homogeneous medium. This indicated the general stabilising effects of medium heterogeneity.

Below is a summary of the dispersive homogeneous and heterogeneous stability studies.

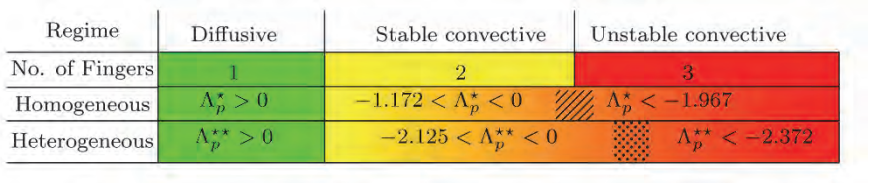


Fig. 3.23 Summary of homogeneous and heterogeneous media

The figure shows the appearance of three fingers at a bigger density contrast than in homogeneous media. This is caused by mixing, which damps out certain instabilities and reduces the quantity of instabilities with wavelengths long enough to grow into fingers.

From the figure, the demarcation of the stability regimes by the stability number can be stated into a stability criterion as follows:

1. A nearly diffusive regime $\Lambda_p^{**} > 0$,
2. A convective regime with stable solutions $-2.125 < \Lambda_p^{**} < 0$,
3. A convective regime with unstable solutions $-2.372 < \Lambda_p^{**} < -2.125$.

The homogeneous and heterogeneous studies could not be compared directly. That is because of the difference in the way the perturbations are induced: via a sinus function in the homogeneous medium and by the heterogeneities in the heterogeneous medium. The fluctuations of the sinus function also reduce the total amount of salt that comes into the domain. Even with these discrepancies, a critical correlation length was obtained that is of the order of the critical wavelength.

Medium anisotropy was not included in the expression for the stability number. However, the stabilising effect was shown in the evolution of the longitudinal coefficient. The range of unstable density contrasts that resulted into asymptotic longitudinal coefficients can be used to estimate the density contrasts stabilised by heterogeneities.

The studies addressed the central question regarding when medium heterogeneities stabilised: when the variance and correlation length are big and small respectively.

Mixing is then very intense that the instabilities that form are smoothed out. On the other hand when the variance is small and the correlation length large, mixing is not very effective and some instabilities grow into fingers.

4 Thermohaline-driven flow and thermodiffusion in porous media

The purpose of this contribution was to investigate some aspects of thermodiffusion in the context of thermohaline-driven flow in porous media of hydrogeological interest. The reasons for undertaking this study were: (a) to determine the circumstances in which thermodiffusion is visible, and those in which it is not, in benchmark problems for variable density flow; (b) to deepen the understanding of some thermodynamic aspects related to transport processes in porous media; and, above all, (c) to acquire expertise for the development of the software d³f and r³t. The results presented in this part of the report have been firstly discussed in /GRI 11/.

In order to achieve the goals, some thermodynamic concepts related to transport processes in porous media were reviewed, and this approach was compared with some models of thermohaline-driven flow and thermodiffusion found in the literature. Five test cases were considered, in which the role of thermodiffusion becomes increasingly visible.

Before entering the technical aspects of the study, an overview is provided of the type of problems that is investigated.

A leading topic in hydrogeology and environmental science is the investigation of salinity- and thermohaline-driven flow in porous media of hydrological relevance. The porous medium and the fluid are usually identified with a soil and groundwater, respectively. The name given to these two types of flow depends on the causes that have either brought the fluid out of mechanic equilibrium or altered its dynamic state. Salinity-driven flow is due to the non-uniform change of the mass density of the fluid consequent to mixing together water and brine (the latter is a mixture of water and various salts). Exposing the fluid to thermal gradients or mixing together flows at different temperatures triggers temperature-driven flow. Thermohaline-driven flow is the combined effect of the just mentioned types of flow. In all these cases, in order to observe flow, it is necessary that the alteration of the density of the fluid favours motion. Within the context sketched above, some of the most typically studied problems are seawater intrusion into coastal aquifers, upconing of hypersaline water from deep aquifers, flow around salt domes or sedimentary rocks, and contamination of aquifers caused by waste. Some of the reasons for undertaking these studies are the management of

freshwater supplies (especially in arid or urbanized zones) and the solution of forecast and remediation problems (for example, in the neighbourhood of nuclear waste repositories).

Thermodiffusion is the coupling between mass and heat diffusion, which occurs in a multi-constituent system that is brought out of thermodynamic equilibrium. This phenomenon can be mathematically modelled by studying the expression of dissipation for the system under investigation, and determining generalized forms of both Fick's and Fourier's laws /DEG 69/. If the validity of Onsager's relations is accepted, these generalized laws feature, beyond the coefficients of mass and heat diffusion, two other phenomenological coefficients, referred to as cross coefficients /BEA 72/, which link the diffusive mass flux with the temperature gradient (Soret effect), and the heat flux with the concentration gradient (Dufour effect), respectively. Since Onsager's relations should be symmetric, the coefficients that describe the Soret and Dufour effects are related to each other.

Although there are several papers in which salinity- and temperature-driven flow are investigated simultaneously /BAL 88/, /DIE 02/, /GRA 07/, /OLD 98, 99, 00/, there are cases in which they are addressed separately in order to highlight some peculiar aspects of each of the two problems (cf., for example, /JOH 02/ and /HOL 01/). Density- and temperature-driven flow treated together, and the generalized forms of Fick's and Fourier's Laws that include the Soret and Dufour effects are considered. For the purposes of this project, Hybrid Mixture Theory (cf., for example, /BEN 00/ and the references therein) is used for the extension of the main results of the theory of thermodiffusion /DEG 69/, /LAN 84/ to porous media, and the study of dissipation is performed.

The idea of drawing the attention of researchers on thermodiffusion in the context of porous media of hydrogeological relevance is not new. Indeed, it can be found in the books /BEA 72/, /BEA 79/, /BEA 90/, where the authors, through the exploitation of the Onsager's relations, provide the mathematical expressions for the Soret and Dufour coefficients, i. e. the phenomenological quantities associated to the Soret and Dufour effect, respectively. There are also other publications in this field that account for thermodiffusion. For example, /HAS 86/ shall be cited here, where a generalization of Fick's law including thermal effects is obtained through a Coleman-Noll analysis of the dissipation inequality, /PAR 88/, where the instability of thermohaline flow with thermal diffusion and horizontal gradients is studied, /BEN 01/ and /COS 02/, where some measurement experiments of the Soret coefficient are given together with thermal and

solutal convection. These effects are, however, neglected in many applications. There are three major reasons for leaving the Soret and Dufour effects out of many practical problems related to porous media: firstly, the Soret and Dufour effects are often actually negligible in comparison with other phenomena; secondly, they noticeably complicate the equations to solve; thirdly, the mathematical expressions of the Soret and Dufour coefficients are based on the assumption that the Onsager's relations hold true also in the case of systems, or physical situations, in which no experimental feedback is available. This last point has led some authors to undertake investigations towards the validation of the Onsager's relations from both the theoretical and experimental point of view. For example, /ING 73/, and /ROW 80/ studied the Dufour effect in mixtures of non-electrolyte liquids, and related it to a thermodynamic quantity, named *heat of transport*, which measures the amount of heat transported by the concentration gradient of the solute in a given mixture. Rowley and Horne /ROW 80/ provided also an experimental validation of their theoretical model in order to support the definition of the heat of transport for the systems under investigation. On the other hand, Sugisaki /SUG 75/, Petit et al. /PET 86/, and Sanyal and Mukjerjee /SAN 88/ determined the Soret coefficient experimentally for electrolyte solutions, and determined the heat of transport by employing the theory of /AGA 63/.

More recently, the interest in the determination of the range of validity of the Onsager's relations, and the study of the Soret and Dufour effects has also turned to completely different problems, such as, for example, the modelling of solar ponds /CEL 05, 06/, and the physics of premelting solids /REM 01/. The feature that all the above cited publications have in common is that, quite independently on whether one deals with a non-electrolytic or an electrolytic solution, a premelting solid or a porous medium, the mathematical description of the diverse underlying physics is unavoidably the same, for it is built on the same formal treatment of irreversible thermodynamics. For this reason, in the formulation of density- and temperature-driven flow in a porous medium, the Soret and Dufour effects are not neglected a priori. Rather, both effects are included, and then the cases are tested, in which they are actually negligible, and those in which they become visible.

Technical note: for designing a numerical simulator it is useful to know the dependencies of the parameters on the primary variables temperature, pressure and salinity in order to select an appropriate underlying solution scheme. In a separate report the mathematical formulations for the fluid parameters as well as complementary information about the properties of rocks are compiled from the literature /KRO 10/. This

report shows that pressure has only little or no influence on the fluid properties so that variations in the pressure can safely be neglected in a numerical model.

4.1 Description of the mathematical model

A porous medium is macroscopically described by a biphasic mixture consisting of a fluid- and a solid-phase. The former is a two-constituent fluid experiencing single-phase flow, and the latter is a porous solid, which may be identified with the rocky matrix of a soil. In the following, the fluid- and the solid-phase are denoted by $\mathcal{F}_f$ and $\mathcal{F}_r$, respectively, and the saturation condition is assumed to apply at all times and all points in space, i. e. $\phi_f + \phi_r = 1$, where ϕ_j is the volume fraction of phase $\mathcal{F}_j$, and $j \in \{f, r\}$.

Whereas in the presence of transfer processes between the fluid- and the solid-phase, each phase $\mathcal{F}_j$ should be regarded as a mixture on its own, which comprises the same number of constituents as the other one /BEN 00/, in the absence of such processes it is possible to assume that $\mathcal{F}_f$ and $\mathcal{F}_r$ are a multi- and single-constituent system, respectively. Here, the solid-phase is considered as a single-constituent medium, and the fluid-phase is assumed to comprise two constituents, i. e. water and a solute (for example, brine or salt). Water and solute are referred to as $\mathcal{C}_w$ and $\mathcal{C}_s$, respectively.

Let ϱ_r denote the mass density of the solid-phase, and let the mass density of the fluid-phase be defined as $\varrho_f = \varrho_w + \varrho_s$. This definition is customary in the context of Mixture Theory (e. g., /BEN 00/, /HAS 86/). The composition of the fluid-phase is described through the mass fraction of its constituents, which are defined by $\omega_a = \varrho_a / \varrho_f$, with $a \in \{w, s\}$, and are linearly dependent through the constraint $\omega_w + \omega_s = 1$.

4.1.1 Balance laws

The general form of balance laws for the solid-phase and the constituent $\mathcal{C}_a$ is given by

$$\partial_t(\phi_r \varrho_r \psi_r) + \operatorname{div}(\phi_r \psi_r \boldsymbol{v}_r) + \operatorname{div}(\boldsymbol{\Phi}_r) = \phi_r \varrho_r (\mathcal{M}_r + \mathcal{G}_r), \tag{4.1}$$

$$\partial_t\big(\phi_f \varrho_f \omega_a \psi_a\big) + \operatorname{div}\big(\phi_f \varrho_f \omega_a \psi_a \boldsymbol{v}_a\big) + \operatorname{div}(\boldsymbol{\Phi}_a) = \phi_f \varrho_f \omega_a \psi_a (\mathcal{M}_a + \mathcal{G}_a). \tag{4.2}$$

Here, $\boldsymbol{v}_a$, $\boldsymbol{\Phi}_a$, $\mathcal{M}_a$, and $\mathcal{G}_a$ represent the velocity, flux, net production (or decay), and source (or sink) associated with the given field ψ_a, referred to the constituent $\mathcal{C}_a$ of the fluid-phase. Physical quantities with index "*r*" have the same physical meaning as those labelled with index "*a*", but are referred to the solid-phase. Since equations (4.1) and (4.2) are compact ways of writing balance of mass, momentum, energy, and entropy, the quantities ψ_a and ψ_r may represent either a scalar or a vector. In the case of balance of momentum, ψ_a and ψ_r are identified with $\boldsymbol{v}_a$ and $\boldsymbol{v}_r$, respectively, quantities $\mathcal{M}_a$, $\mathcal{M}_r$, $\mathcal{G}_a$ and $\mathcal{G}_r$ are vector fields, fluxes $\boldsymbol{\Phi}_a$ and $\boldsymbol{\Phi}_r$ are second-rank tensors, and the products $\psi_a\boldsymbol{v}_a$ and $\psi_r\boldsymbol{v}_r$ on the LHS of equations (4.1) and (4.2) are understood as the dyadic products $\psi_a\boldsymbol{v}_a = \boldsymbol{v}_a \otimes \boldsymbol{v}_a$ and $\psi_r\boldsymbol{v}_r = \boldsymbol{v}_r \otimes \boldsymbol{v}_r$, respectively. Balance laws for the solid-phase and the generic constituent $\mathcal{C}_a$ of the fluid-phase are obtained by substituting the quantities in Tab. 4.1 and Tab. 4.2 into equations (4.1) and (4.2). In Tab. 4.1, E_r is the internal energy density; $\boldsymbol{\sigma}_r$ is the Cauchy stress tensor; $\boldsymbol{J}_{rT}$ is the heat flux vector; Θ is absolute temperature; S_r is the entropy density; $\boldsymbol{g}$ is the gravity acceleration vector; and $\boldsymbol{T}_r$, Q_r, and η_r are sources (or sinks) of momentum, energy, and entropy due to exchange interactions occurring between the solid- and the fluid-phase. The physical quantities in Tab. 4.2 have the same physical meaning as above, but refer to the constituents of the fluid-phase.

Tab. 4.1 Thermodynamics quantities and related fluxes to substitute in eq. (4.1)

Quantity	ψ_r	$\boldsymbol{\Phi}_\mathrm{r}$	$\mathcal{M}_r$	$\mathcal{G}_r$
Mass	1	$\mathbf{0}$	0	0
Momentum	$\boldsymbol{v}_r$	$-\boldsymbol{\sigma}_r$	$\boldsymbol{g}$	$\boldsymbol{T}_r$
Energy	$E_r + \frac{1}{2}\lvert\boldsymbol{v}_r\rvert^2$	$(\boldsymbol{J}_{rT} - \boldsymbol{\sigma}_r^T\boldsymbol{v}_r)$	$\boldsymbol{g}\cdot\boldsymbol{v}_r$	$\boldsymbol{T}_r\cdot\boldsymbol{v}_r + Q_r$
Entropy	S_r	$\frac{1}{\Theta}J_{rT}$	Γ_r	η_r

Tab. 4.2 Thermodynamics quantities and related fluxes to substitute in eq. (4.2)

Quantity	ψ_α	$\boldsymbol{\Phi}_\alpha$	$\mathcal{M}_\alpha$	$\mathcal{G}_\alpha$
Mass	1	$\mathbf{0}$	0	0
Momentum	$\boldsymbol{v}_\alpha$	$-\boldsymbol{\sigma}_\alpha$	$\boldsymbol{g}$	$\boldsymbol{T}_\alpha$
Energy	$E_\alpha + \frac{1}{2}\lvert\boldsymbol{v}_\alpha\rvert^2$	$(\boldsymbol{J}_{\alpha T} - \boldsymbol{\sigma}_\alpha^T \boldsymbol{v}_\alpha)$	$\boldsymbol{g} \cdot \boldsymbol{v}_\alpha$	$\boldsymbol{T}_\alpha \cdot \boldsymbol{v}_\alpha + Q_\alpha$
Entropy	S_α	$\frac{1}{\Theta} J_{\alpha T}$	Γ_α	η_α

By performing the mass average of each source (or sink) listed in Tab. 4.2 (terms denoted by $\mathcal{G}_\alpha$), it is possible to define overall sources (or sinks) of momentum, energy, and entropy associated with the fluid-phase as a whole (/BEN 00/, /HAS 86/), i. e.

$$\boldsymbol{T}_f = \omega_w \boldsymbol{T}_w + \omega_s \boldsymbol{T}_s, \tag{4.3}$$

$$Q_f = \omega_w [Q_w + \boldsymbol{T}_w \cdot \boldsymbol{v}_w] + \omega_s [Q_s + \boldsymbol{T}_s \cdot \boldsymbol{v}_s], \tag{4.4}$$

$$\eta_f = \omega_w \eta_w + \omega_s \eta_s. \tag{4.5}$$

The requirement that the mixture is closed implies that the source (or sink) terms $\boldsymbol{T}_f, Q_f$, and η_f are related to their solid-phase counterparts through the following constraints:

$$\phi_f \varrho_f \boldsymbol{T}_f + \phi_r \varrho_r \boldsymbol{T}_r = \mathbf{0}, \tag{4.6}$$

$$\phi_f \varrho_f \left[Q_f + \boldsymbol{T}_f \cdot \boldsymbol{v}_f\right] + \phi_r \varrho_r [Q_r + \boldsymbol{T}_r \cdot \boldsymbol{v}_r] = 0, \tag{4.7}$$

$$\phi_f \varrho_f \eta_f + \phi_r \varrho_r \eta_r = 0, \tag{4.8}$$

where $\varrho_f = \varrho_w + \varrho_s$ and $\boldsymbol{v}_f = \omega_w \boldsymbol{v}_w + \omega_s \boldsymbol{v}_s$ are the mass density and velocity of the fluid-phase as a whole. Equations (4.6) - (4.8) define the fluid-solid interface as an ideal interface, i. e. a surface without thermodynamic properties /BEN 00/.

4.1.2 Dissipation inequality

It is convenient to write the dissipation inequality in a form similar to that proposed in /BEN 00/ and /HAS 86/. Yet, in order to focus the attention on the main aspects of the problem, a less general framework is adopted here, based on the following hypotheses:

- Absence of irradiative sources of energy in both the solid- and the fluid-phase constituents;
- Absence of mass-exchange phenomena between phases;
- Macroscopically inviscid fluid-phase;
- The solid-phase is rigid and at rest;
- The fluid-phase volume fraction is a given constant;
- The mass density of the solid-phase is a given constant.

By introducing the Helmholtz free energy densities $\Psi_r = E_r - \Theta S_r$ and $\Psi_a = E_a - \Theta S_a$ (with $a \in \{w, s\}$), it is possible to write the following expressions of entropy productions:

$$\phi_r \varrho_r \Theta \Gamma_r = -\phi_r \varrho_r \frac{\partial \Psi_r}{\partial t} - \phi_r \varrho_r S_r \frac{\partial \Theta}{\partial t} - \boldsymbol{J}_{rT} \cdot \frac{\mathrm{grad}(\Theta)}{\Theta} + \phi_r \varrho_r (Q_r - \Theta \eta_r) \tag{4.9}$$

$$\begin{aligned} \phi_f \varrho_f \omega_a \Theta \Gamma_a = \; & -\phi_f \varrho_f \omega_a \frac{\mathrm{D}_a \Psi_a}{\mathrm{D}t} - \phi_f \varrho_f \omega_a S_a \frac{\mathrm{D}_a \Theta}{\mathrm{D}t} - \boldsymbol{J}_{aT} \cdot \frac{\mathrm{grad}(\Theta)}{\Theta} \\ & -\boldsymbol{\sigma}_a : \mathrm{grad}(\boldsymbol{v}_a) + \phi_f \varrho_f \omega_a (Q_a - \Theta \eta_a). \end{aligned} \tag{4.10}$$

In order to obtain an expression for the entropy production of the mixture as a whole, the fluid-phase Helmholtz free energy density $\Psi_f = \sum_a \omega_a \Psi_a$ is introduced, and the relative velocity of the constituent $\mathcal{C}_a$ with respect to the mass average velocity $\boldsymbol{v}_f$, i. e. $\boldsymbol{u}_a = \boldsymbol{v}_a - \boldsymbol{v}_f$, with $a \in \{w, s\}$. Furthermore, the following quantities are defined:

$$S_f = \sum_a \omega_a S_a, \tag{4.11}$$

$$\boldsymbol{\sigma}_f = \sum_a \left[\boldsymbol{\sigma}_a - \phi_f \varrho_f \omega_a \boldsymbol{u}_a \otimes \boldsymbol{u}_a\right], \tag{4.12}$$

$$\boldsymbol{J}_{fT} = \sum_a \left\{\boldsymbol{J}_{aT} + \left[\phi_f \varrho_f \omega_a \left(E_a + \tfrac{1}{2}|\boldsymbol{u}_a|^2\right) u_a - \boldsymbol{\sigma}_a^{\mathrm{T}} \boldsymbol{u}_a\right]\right\}, \tag{4.13}$$

which represent the fluid-phase entropy density, Cauchy stress tensor, and energy flux, respectively. These definitions, and the notation introduced so far, enable to express the entropy production of the mixture as a whole, $\varrho\Gamma = \phi_f \varrho_f \sum_a \omega_a \Gamma_a + \phi_r \varrho_r \Gamma_r$, i. e.

$$\begin{aligned}
\varrho\Theta\Gamma = & -\phi_f \varrho_f \frac{\mathrm{D}_f \Psi_f}{\mathrm{D}t} - \phi_r \varrho_r \frac{\partial \Psi_r}{\partial t} - \phi_f \varrho_f S_f \frac{\mathrm{D}_f \Theta}{\mathrm{D}t} - \phi_r \varrho_r S_r \frac{\partial \Theta}{\partial t} \\
& + \sum_a \left[\boldsymbol{\sigma_a} - \phi_f \varrho_f \omega_a \Psi_a \mathbf{I} \right] : \mathrm{grad}(\boldsymbol{u}_a) \\
& - \sum_a \left[\mathrm{grad}\left(\phi_f \varrho_f \omega_a \Psi_a \right) + \phi_f \varrho_f \omega_a \boldsymbol{T}_a \right] \cdot \boldsymbol{u}_a \\
& + \left[\boldsymbol{\sigma}_f + \sum_a \phi_f \varrho_f \omega_a \boldsymbol{u}_a \otimes \boldsymbol{u}_a \right] : \mathrm{grad}(\boldsymbol{v}_f) - \phi_f \varrho_f \boldsymbol{T}_f \cdot \boldsymbol{v}_f \\
& - \frac{\mathrm{grad}(\Theta)}{\Theta} \cdot \left\{ \boldsymbol{J}_T - \sum_a \left[\phi_f \varrho_f \omega_a \left(\Psi_a + \frac{1}{2} |\boldsymbol{u}_a|^2 \right) \boldsymbol{u}_a - \boldsymbol{\sigma}_a^{\mathrm{T}} \boldsymbol{u}_a \right] \right\} \geq 0
\end{aligned} \tag{4.14}$$

where $\boldsymbol{J}_T = \boldsymbol{J}_{rT} + \boldsymbol{J}_{fT}$ is the heat flux vector of the mixture as a whole.

4.1.3 Constitutive framework

The unknowns featuring in the dissipation inequality (4.14) can be split into a set of free unknowns, $\mathcal{U}_{\mathrm{free}}$, and a set of dependent unknowns, $\mathcal{U}_{\mathrm{dep}}$. In the following is set:

$$\mathcal{U}_{\mathrm{free}} = \{\omega_s, \boldsymbol{u}_s, \boldsymbol{v}_f, \Theta\}, \tag{4.15}$$

$$\mathcal{U}_{\mathrm{dep}} = \{\Psi_s, \Psi_f, \Psi_r, S_f, S_r, \boldsymbol{\sigma}_s, \boldsymbol{\sigma}_f, \boldsymbol{T}_s, \boldsymbol{T}_f, \varrho_f, \boldsymbol{J}_T\}. \tag{4.16}$$

In equation (4.15), the mass fraction and relative velocity of water, denoted by ω_w and $\boldsymbol{u}_w$, respectively, do not feature because they are related to the quantities ω_s and $\boldsymbol{u}_s$ through the constraints $\sum_a \omega_a = 1$ and $\sum_a \omega_a \boldsymbol{u}_a = \mathbf{0}$.

In several publications dealing with the Theory of Mixtures, a more general framework is presented, in which the mass density of the fluid-phase, ϱ_f, is treated as an independent variable (/HAS 86/, /BEN 00/, /HAS 88/, /WIL 03/, /BEN 07/). This is done because neither constraints (such as incompressibility) not state laws are imposed on this field variable. Here, however, ϱ_f is considered as a dependent constitutive variable because, in consistency with what is usually done in modelling salinity- and tempera-

ture-driven flow, it is expressed through a function of the solute mass fraction, ω_s, and temperature, Θ, i. e.

$$\varrho_f = \varrho_f(\omega_s, \Theta). \tag{4.17}$$

Moreover, the solid-phase mass density, ϱ_r, is regarded as a constant and is therefore excluded from the lists of both free and independent variables.

Another remark is about multi-temperature models (cf., for example, /RUG 07/). Here, it is assumed that all constituents in the fluid-phase, i. e. $\mathcal{C}_w$ and $\mathcal{C}_s$, and all phases are in local thermal equilibrium. This means that, at each point of the mixture, they all share the same value of temperature, even in the case in which temperature is nonuniform over a given length-scale.

In this model, the set of independent constitutive variables (ICVs) is obtained through the union of $\mathcal{U}_{\text{free}}$ and a set of variables comprising the gradients of solute mass fraction and temperature, while the set of dependent constitutive variables (DCVs) is identified with $\mathcal{U}_{\text{dep}}$, i. e.

$$\text{ICV} = \mathcal{U}_{\text{free}} \cup \{\text{grad}(\omega_s), \text{grad}(\Theta)\}, \quad \text{DCV} = \mathcal{U}_{\text{dep}}. \tag{4.18}$$

Furthermore, it is assumed that the fluid-phase Helmholtz free energy density, Ψ_f, is a constitutive function of both ω_s and Θ, while the solid-phase Helmholtz free energy density, Ψ_r, depends on temperature only, i. e.

$$\Psi_f = \Psi_f(\omega_s, \Theta), \quad \Psi_r = \Psi_r(\Theta). \tag{4.19}$$

The study of the dissipation inequality is carried out by using Liu's Theorem /LIU 72/. In Liu's Theorem, each balance law is viewed as a constraint to append to the overall entropy production, and it is proven that the exploitation of the dissipation principle can be equivalently based on equation (4.14) or on a modified form of it, which is obtained by premultiplying each balance law by a Lagrange multiplier, adding together the consequent expressions, and summing the final result to $\varrho\Theta\Gamma$.

Here, fort the sake of conciseness, only mass balance laws are considered, and the constraint on relative velocities of the constituents of the fluid-phase. The latter is taken into account as in /BEN 00/, i. e.

$$C^u = \phi_f \mathbf{M}_f : \mathrm{grad}\left(\varrho_f \omega_s \boldsymbol{u}_s + \varrho_f \omega_w \boldsymbol{u}_w\right), \tag{4.20}$$

where $\mathbf{M}_f$ is a second-order tensor of Lagrange multipliers. By taking explicitly into account the incompressibility of the solid-phase, and the constitutive law expressing ϱ_f (cf. equation (4.17)), also the following constraint is considered, which is obtained by manipulation of the mass balance laws, i. e.

$$\begin{aligned} C^m = \; & p\phi_f \left\{ \frac{1}{\varrho_f} \frac{\partial \varrho_f}{\partial \omega_s} \frac{\mathrm{D}_f \omega_s}{\mathrm{D}t} + \frac{1}{\varrho_f} \frac{\partial \varrho_f}{\partial \Theta} \frac{\mathrm{D}_f \Theta}{\mathrm{D}t} + \mathrm{div}(\boldsymbol{v}_f) \right\} \\ & + m_s \left\{ \phi_f \varrho_f \frac{\mathrm{D}_f \omega_s}{\mathrm{D}t} + \mathrm{div}(\phi_f \varrho_f \omega_s \boldsymbol{u}_s) \right\}, \end{aligned} \tag{4.21}$$

where is referred to the Lagrange multiplier *p* as to the pressure of the mixture. By adding the constraints C^u and C^m to the RHS of equation (4.14), a modified expression of the entropy production is found, i. e.

$$\varrho \Theta \tilde{\Gamma} = \varrho \Theta \Gamma + C^u + C^m \geq 0. \tag{4.22}$$

For the problem under consideration, the study of the dissipation inequality written in equation (4.22) is performed according to the Coleman-Noll procedure and in /BEN 00/, /GRI 09a/, /GRI 09b/, /GRI 10/. The results are:

Lagrange multipliers

$$m_s = \frac{\partial \Psi_f}{\partial \omega_s} - \frac{p}{\varrho_f^2} \frac{\partial \varrho_f}{\partial \omega_s}, \tag{4.23}$$

$$\mathbf{M}_f = \Psi_w \mathbf{I} - \frac{1}{\phi_f \varrho_f \omega_w} \boldsymbol{\sigma}_w. \tag{4.24}$$

Entropies and Cauchy stress tensors

$$S_f = -\frac{\partial \Psi_f}{\partial \Theta} + \frac{p}{\varrho_f^2} \frac{\partial \varrho_f}{\partial \Theta}, \quad S_r = -\frac{\partial \Psi_r}{\partial \Theta}, \tag{4.25}$$

$$\boldsymbol{\sigma}_s = \phi_f \varrho_f \omega_s (\Psi_s - m_s) \mathbf{I} - \phi_f \varrho_f \omega_s \mathbf{M}_f, \tag{4.26}$$

$$\boldsymbol{\sigma}_f = -\phi_f p\mathbf{I} - \sum_a \phi_f \varrho_f \omega_a \boldsymbol{u}_a \otimes \boldsymbol{u}_a. \tag{4.27}$$

In equations (4.23) - (4.27), m_s is identified with the relative chemical potential of constituent $\mathcal{C}_s$, i. e. $m_s = \mu_{sw} = \mu_s - \mu_w$, where μ_s and μ_w are the chemical potentials of the solute, and water, respectively, Ψ_w is the Helmholtz free energy density of water, S_f and S_r are the entropy densities of the fluid- and the solid-phase, respectively, $\boldsymbol{\sigma}_s$ is the Cauchy stress tensor of the solute, and $\boldsymbol{\sigma}_f$ is the Cauchy stress tensor of the fluid-phase. In order for equation (4.26) to be consistent with the result (4.27), it is necessary that $\sum_a \boldsymbol{\sigma}_a = -\phi_f p\mathbf{I}$. By using equation (4.24), this condition can be applied to show that the absolute chemical potentials μ_a ($a \in \{w, s\}$) are given by

$$\mu_a = \Psi_a + \frac{p_a}{\varrho_f \omega_a}, \quad a \in \{w, s\}, \tag{4.28}$$

where p_a is said to be the partial pressure of the fluid-phase constituent $\mathcal{C}_a$. Furthermore, the Cauchy stress tensor of constituent $\mathcal{C}_a$ is given by $\boldsymbol{\sigma}_a = -\phi_f p_a \mathbf{I}$, and the partial pressure satisfy $\sum_a p_a = p$.

Notice that the only additional free unknown, which contributes to the set $\mathcal{U}_{\text{free}}$ is the pressure *p*. Indeed, this Lagrange multiplier must be determined by solving the mass balance equation of the fluid-phase as a whole. Finally, it shall be remarked that both the solute relative chemical potential $m_s = \mu_{sw}$ (cf. equation (4.23)) and the fluid-phase entropy density S_f (cf. equation (4.25a)) consist of two parts. The first part on the RHS of equations (4.23) and (4.25a) is related to the unconstrained Helmholtz free energy density, $\Psi_f = \Psi_f(\omega_s, \Theta)$, and is said to be constitutive. The second terms are contributions due to the relation $\varrho_f = \varrho_f(\omega_s, \Theta)$ and feature the Lagrange multiplier *p*.

The results (4.23) - (4.27), the constraint $\sum_a \omega_a \boldsymbol{u}_a = \mathbf{0}$, and the further assumption of smallness of both solute relative velocity and fluid-phase velocity (i. e. $|\boldsymbol{u}_s|^2 \ll 1$, and $|\boldsymbol{v}_f|^2 \ll 1$) yield the following form of the residual dissipation inequality:

$$\varrho\Theta\Gamma = -\boldsymbol{q}_f \cdot \boldsymbol{b}_f - \boldsymbol{J}_d \cdot \operatorname{grad}(\mu_{sw}) - \Theta^{-1}(\boldsymbol{J}_T - \mu_{sw}\boldsymbol{J}_d) \cdot \operatorname{grad}(\Theta) \geq 0, \tag{4.29}$$

where $\boldsymbol{b}_f$ is the dissipative part of the momentum exchange between the fluid- and the solid-phase, $\boldsymbol{q}_f = \phi_f \boldsymbol{v}_f$ is the specific discharge /BEA 72/, and $\boldsymbol{J}_d = \phi_f \varrho_f \omega_s \boldsymbol{u}_s$ is the diffusive mass flux of the solute.

4.2 Thermodiffusion

In order to focus attention on thermodiffusion, it is assumed here that the specific discharge $\boldsymbol{q}_f$ is coupled neither to the diffusive mass flux $\boldsymbol{J}_d$ nor to the thermal flux $\boldsymbol{J}_T$. If it is also assumed that the considered porous medium is isotropic, then, after dividing (4.29) by Θ, the Onsager's relations can be written as

$$q_f = -\frac{K}{\nu_f}\left[\mathrm{grad}(p) - \varrho_f \boldsymbol{g}\right], \tag{4.30}$$

$$\boldsymbol{J}_d = -\phi_f \varrho_f L_{dd} \mathrm{grad}\left(\frac{\mu_{sw}}{\Theta}\right) - \phi_f \varrho_f L_{dT} \frac{\mathrm{grad}(\Theta)}{\Theta^2}, \tag{4.31}$$

$$\boldsymbol{J}_T = -\phi_f \varrho_f L_{Td} \mathrm{grad}\left(\frac{\mu_{sw}}{\Theta}\right) - L_{TT} \mathrm{grad}(\Theta), \tag{4.32}$$

where K is the porous medium permeability, ν_f denotes the fluid-phase dynamic viscosity, L_{dd} and L_{TT} are the phenomenological coefficients of pure mass and thermal diffusion, respectively, and L_{dT} and L_{Td} are the Onsager's cross coefficients, which couple the fluxes $\boldsymbol{J}_d$ and $\boldsymbol{J}_T$, and give rise to Soret and Dufour effects. The form of the fluxes $\boldsymbol{J}_d$ and $\boldsymbol{J}_T$ is given by /DEG 69/ and is used, for example, by Ingle and Horne /ING 73/, Rowley and Horne /ROW 80/, Sugisaki /SUG 75/, Petit et al. /PET 86/, Sanyal and Mukherjee /SAN 88/, Agar /AGA 63/, and Rauch /RAU 06/. There is also an alternative formulation of the Onsager's relation describing thermodiffusion (cf., for example, /LAN 84/). However, equations (4.31) and (4.32) are more suitable for these purposes, because they explicitly feature the fluxes $\boldsymbol{J}_d$ and $\boldsymbol{J}_T$, that are measurable by experiments.

In the isothermal case, Bader and Kooi /BAD 05/ provided a theoretical framework in which $\boldsymbol{q}_f$ and $\boldsymbol{J}_d$ are coupled with each other through Onsager's reciprocal coefficients. Here, however, here it is preferred to leave $\boldsymbol{q}_f$ decoupled from $\boldsymbol{J}_d$ and $\boldsymbol{J}_T$ for the sake of simplicity.

In order to write Onsager's relations (4.30) - (4.32) in a form in which the coefficients L_{dd}, L_{dT}, L_{Td}, and L_{TT} are substituted by directly measureable quantities, the relative chemical potential of the solute, μ_{sw}, has to be determined as a function of the state variables p, ω_s, and Θ. For this purpose, a constitutive model has to be formulated.

4.2.1 Constitutive model

Since the mass densities of the fluid- and solid-phase are assumed to be independent on pressure, and equivalent formulation of the constitutive theory can be obtained by introducing the Gibbs free energy densities instead of the Helmholtz free energy densities prescribed in equation (4.19), i. e.

$$G_f(p,\omega_s,\Theta) = \Psi_f(\omega_s,\Theta) + \frac{p}{\varrho_f(\omega_s,\Theta)}, \tag{4.33}$$

$$G_r(p,\Theta) = \Psi_r(\Theta) + \frac{p}{\varrho_r}. \tag{4.34}$$

If the thermal capacity of the solid-phase, C_r, is a constant, the solid-phase free energy density reads

$$G_r(p,\Theta) = \frac{p}{\varrho_r} - C_r\left[\Theta \ln\left(\frac{\Theta}{\Theta_0}\right) - \Theta\right], \tag{4.35}$$

where Θ_0 is a constant reference temperature. For the fluid-phase, the determination of the Gibbs free energy density must be consistent with the following thermodynamic results:

$$\frac{\partial G_f}{\partial p}(p,\omega_s,\Theta) = \frac{1}{\varrho_f(\omega_s,\Theta)}, \tag{4.36}$$

$$\frac{\partial G_f}{\partial \omega_s}(p,\omega_s,\Theta) = \mu_{sw}(p,\omega_s,\Theta), \quad \frac{\partial G_f}{\partial \Theta}(p,\omega_s,\Theta) = -S_f(p,\omega_s,\Theta), \tag{4.37}$$

where both the relative chemical potential μ_{sw} and entropy density S_f are identified with functions of the state variables.

Following /OLD 98/, the fluid-phase mass density is prescribed to be given by

$$\varrho_f(\omega_s,\Theta) = \frac{\varrho_{pw}(\Theta)\varrho_{ps}(\Theta)}{\varrho_{ps}(\Theta) - [\varrho_{ps}(\Theta) - \varrho_{pw}(\Theta)]\omega_s}, \tag{4.38}$$

where $\varrho_{pw}(\Theta) = \varrho_{pw}^0 \exp\{-\alpha_w[\Theta - \Theta_0]\}$ and $\varrho_{ps}(\Theta) = \varrho_{ps}^0 \exp\{-\alpha_s[\Theta - \Theta_0]\}$ are the mass densities of pure water and pure solute as functions of temperature, ϱ_{ps}^0 and ϱ_{pw}^0 are constant densities corresponding to the reference temperature Θ_0, and α_w and α_s are the thermal expansion coefficients of water and solute in a neighbourhood of Θ_0.

Under the assumption that the heat capacity of the fluid-phase, C_f, is constant, it is required that the chemical potentials of the solute and water, μ_s and μ_w, are given by the following functions of state variables:

$$\mu_s(p,\omega_s,\Theta) = \mu_s^i(\omega_s,\Theta) + \frac{p}{\varrho_{ps}(\Theta)} - C_f\left[\Theta\ln\left(\frac{\Theta}{\Theta_0}\right) - \Theta\right], \tag{4.39}$$

$$\mu_w(p,\omega_s,\Theta) = \mu_w^i(\omega_s,\Theta) + \frac{p}{\varrho_{pw}(\Theta)} - C_f\left[\Theta\ln\left(\frac{\Theta}{\Theta_0}\right) - \Theta\right]. \tag{4.40}$$

If the solute under investigation is a salt, which in water dissociated into two ions (for example, NaCl), then the chemical potentials $\mu_s^i(\omega_s,\Theta)$ and $\mu_w^i\,(\omega_s,\Theta)$ are

$$\mu_s^i(\omega_s,\Theta) = \frac{2R\Theta}{M_s}\ln\left[\frac{M_w\omega_s}{(1-\omega_s)M_s+2M_w\omega_s}\right], \tag{4.41}$$

$$\mu_w^i(\omega_s,\Theta) = \frac{R\Theta}{M_w}\ln\left[\frac{M_s(1-\omega_s)}{(1-\omega_s)M_s+2M_w\omega_s}\right], \tag{4.42}$$

where M_s and M_w are the molar masses of the solute and water, respectively, and *R* ($[J\cdot\mathrm{mol}^{-1}\cdot\mathrm{K}^{-1}]$) is the gas constant. It shall be remarked that, although salt dissociates in water, its diffusion and transport can still be studied as if no dissociation took place, as long as no electric field is applied to the mixture (/HAS 86/, /CEL 05/, /CEL 06/, /GAJ 03/). It is also to notice that, since the definitions (4.41) and (4.42) do not take into account any activity coefficient, the chemical potentials $\mu_s^i(\omega_s,\Theta)$ and $\mu_w^i\,(\omega_s,\Theta)$ model an ideal solution /DEN 68/.

By virtue of (4.39) and (4.40), the relative chemical potential μ_{sw} becomes

$$\mu_{sw}(p,\omega_s,\Theta) = \mu_{sw}^i(\omega_s,\Theta) - \frac{\varrho_{ps}(\Theta)-\varrho_{pw}(\Theta)}{\varrho_{ps}(\Theta)\varrho_{pw}(\Theta)}p, \tag{4.43}$$

where $\mu_{sw}^i(\omega_s,\Theta) = \mu_s^i(\omega_s,\Theta) - \mu_w^i(\omega_s,\Theta)$. The construction of the differential form (4.36) - (4.37), and the enforcement of the integrability condition imply that the entropy density is given by

$$\begin{aligned} S_f(p,\omega_s,\Theta) = {} & C_f\ln\left(\frac{\Theta}{\Theta_0}\right) - \left[\frac{\alpha_w}{\varrho_{pw}(\Theta)} - \left(\frac{\alpha_w}{\varrho_{pw}(\Theta)} - \frac{\alpha_s}{\varrho_{ps}(\Theta)}\right)\omega_s\right]p \\ & -\omega_s\frac{2R}{M_s}\ln\left[\frac{M_w\omega_s}{\omega_w M_s+2M_w\omega_s}\right] - \omega_w\frac{R}{M_w}\ln\left[\frac{M_s\omega_w}{\omega_w M_s+2M_w\omega_s}\right], \end{aligned} \tag{4.44}$$

where the notation $\omega_w = 1 - \omega_s$ has been used. Results (4.38), (4.43) and (4.44) yield the following Gibbs free energy density

$$G_f(p, \omega_s, \Theta) = G_f^i(p, \omega_s, \Theta) + \frac{p}{\varrho_f(\omega_s, \Theta)} - C_f \left[\Theta \ln\left(\frac{\Theta}{\Theta_0}\right) - \Theta\right], \tag{4.45}$$

with

$$G_f^i(p, \omega_s, \Theta) = \omega_s \frac{2R\Theta}{M_s} \ln\left[\frac{M_w \omega_s}{\omega_w M_s + 2M_w \omega_w}\right] + \omega_w \frac{R\Theta}{M_w} \ln\left[\frac{M_s \omega_w}{\omega_w M_s + 2M_w \omega_w}\right]. \tag{4.46}$$

It must be remarked that both pairs of chemical potentials, μ_s and μ_w, and μ_s^i and μ_w^i, satisfy the main result of Gibbs' thermodynamics, i. e. their mass average equals the Gibbs free energy density:

$$G_f(p, \omega_s, \Theta) = \omega_s \mu_s(p, \omega_s, \Theta) + (1 - \omega_s)\mu_w(p, \omega_s, \Theta), \tag{4.47}$$

$$G_f^i(\omega_s, \Theta) = \omega_s \mu_s^i(\omega_s, \Theta) + (1 - \omega_s)\mu_w^i(\omega_s, \Theta). \tag{4.48}$$

Looking at the explicit forms of the relative chemical potential, μ_{sw}, and entropy, S_f, it is to be noticed that each of these quantities can be written as the sum of two functions: one of them depends only on the independent constitutive variables ω_s and Θ, and is said to be constitutive; the other one contains the Lagrange multiplier *p* as well as the independent constitutive variables, and is thus said to be constrained, for it represents the contribution of the constraint $\varrho_f = \varrho_f(\omega_s, \Theta)$ to both μ_{sw} and S_f. The constrained parts of μ_{sw} and S_f produce no dissipation. On the other hand, only the constitutive parts of μ_{sw} and S_f play a role in the determination of the material coefficients featuring in the Onsager's relations.

4.2.2 Phenomenological coefficients

After substituting the expression of the relative chemical potential (4.43) into Onsager's relations (4.31) and (4.32), and explicitly computing the gradient of μ_{sw}, one finds

$$\boldsymbol{J}_d = -\phi_f \varrho_f D \mathrm{grad}(\omega_s) - \phi_f \varrho_f D \frac{k_p}{p} \mathrm{grad}(p) - \phi_f \varrho_f D S \omega_s (1 - \omega_s) \mathrm{grad}(\Theta), \tag{4.49}$$

$$\boldsymbol{J}_T = -\phi_f \varrho_f D Q \mathrm{grad}(\omega_s) - \phi_f \varrho_f D Q \frac{k_p}{p} \mathrm{grad}(p) - [L_{TT} - A] \mathrm{grad}(\Theta). \tag{4.50}$$

Here, *D* and *Q* are the diffusivity coefficient and the heat of transport, i. e.

$$D = \frac{L_{dd}}{\Theta}\frac{\partial \mu_{sw}^{i}}{\partial \omega_s}, \quad Q = \frac{L_{Td}}{L_{dd}}, \tag{4.51}$$

the term *A* is given by

$$A = \phi_f \varrho_f \frac{DQh_{sw}}{\Theta\frac{\partial \mu_{sw}^{i}}{\partial \omega_s}}, \tag{4.52}$$

where h_{sw} and S are the relative specific enthalpy and the Soret coefficient, i. e.

$$h_{sw} = \mu_{sw} - \Theta\frac{\partial \mu_{sw}}{\partial \Theta}, \quad S\Theta\omega_s(1-\omega_s)\frac{\partial \mu_{sw}^{i}}{\partial \omega_s} = \frac{L_{dT}}{L_{dd}} - h_{sw}, \tag{4.53}$$

k_p is the baro-diffusion factor /LAN 84/

$$\frac{k_p}{p} = \frac{\partial \mu_{sw}/\partial p}{\partial \mu_{sw}^{i}/\partial \omega_s} = \frac{\partial \mu_{sw}^{c}/\partial p}{\partial \mu_{sw}^{i}/\partial \omega_s}, \tag{4.54}$$

and μ_{sw}^{c} is the constrained part of the relative chemical potential (second term on the RHS of equation (4.43)). The last equality follows from the fact that only the constrained part of the relative chemical potential, μ_{sw}^{c}, can depend on pressure. By computing the derivative $\partial \mu_{sw}^{i}/\partial \omega_s$, it can be seen that the baro-diffusion factor, k_p, vanishes identically when $\omega_s = 0$ or $\omega_s = 1$. Furthermore, because of the symmetry of Onsager's coefficients, it follows that $L_{dT} = L_{Td}$. This requirement also implies that the heat of transport, *Q*, and the Soret coefficient, *S*, are related to each other through

$$Q^* = Q - h_{sw} = S\Theta\omega_s(1-\omega_s)\frac{\partial \mu_{sw}^{i}}{\partial \omega_s}. \tag{4.55}$$

The factor $\omega_s(1-\omega_s)$ in equations (4.53) and (4.55) ensures that the Soret effect vanishes in the limit cases $\omega_s = 0$ or $\omega_s = 1$.

In conclusion, the fluxes $\boldsymbol{J}_d$ and $\boldsymbol{J}_T$ given in equations (4.49) and (4.50) are completely determined by the quantities *D*, k_p, *S*, *Q*, L_{TT}, and h_{sw}. Among those, the coefficients *D*, L_{TT}, and *S* (or *Q*) must be determined experimentally, while the remaining ones are determined by means of the constitutive model, and the symmetry of Onsager's relations (cf. equation (4.55)). In the case of isotropic porous media, the diffusivity coefficient, *D*, is defined as $D = D_m\tau$, where D_m is the coefficient of molecular diffusivity, and

τ is a nondimensional scalar quantity called tortuosity. More generally, also in isotropic porous media, the diffusivity coefficient is replaced by the diffusion-dispersion tensor, $\mathbf{D}_{dd}$, which takes into account the effect of fluid velocity on the trajectories of the solute particles. In the case of porous media which are transversely isotropic with respect to the flow, this tensor, introduced by Scheidegger, reads /BEA 72/

$$\mathbf{D}_{dd} = D\mathbf{I} + \alpha_t |\boldsymbol{q}_f| \mathbf{I} + (\alpha_\ell - \alpha_t) \frac{\boldsymbol{q}_f \otimes \boldsymbol{q}_f}{|\boldsymbol{q}_f|}, \tag{4.56}$$

where α_ℓ and α_t are said to be the longitudinal and transversal velocity correlation lengths, respectively. Sometimes, together with the introduction of $\mathbf{D}_{dd}$, the correction to Fick's law due to Forchhmeimer is used (cf., for example, /DIE 02/). Here, however, only the diffusivity coefficient *D*, and the "standard" form of Fick's law are considered for the sake of simplicity.

Following /OLD 99/, it was assumed here that L_{TT} is constant. However, it should be noticed that, since L_{TT} represents the coefficient of thermal conductivity of the mixture as a whole, it has to be defined through some averaged of the thermal conductivities of the solid- and the fluid-phase (the latter including all of its constants). For example, Holstad /HOL 01/ defines L_{TT} as $L_{TT} = \left(\Lambda_f\right)^{\phi_f} \left(\Lambda_r\right)^{\phi_f - 1}$, where Λ_r is the thermal conductivity of rock, and Λ_f is an effective thermal conductivity of the fluid-phase. Whereas it is licit to assume that Λ_r is constant within a certain temperature range, the thermal conductivity of the fluid-phase depends also on composition, i. e. the amount of solute and solvent. Moreover, if the dynamic regime of the fluid is such that also thermal dispersion effects have to be considered, Diersch and Kolditz /DIE 02/ consider the expression $L_{TT} = \phi_f \Lambda_f + \left(1 - \phi_f\right)\Lambda_r + f(\omega_s)(\mathbf{D}_{dd} - D\mathbf{I})$, where $f(\omega_s)$ is some function of the solute mass fraction, and $\mathbf{D}_{dd}$ is given in equation (4.56). Finally, the experimental values of the Soret coefficient are taken from /CEL 05/.

4.3 Field equations

Under the hypotheses outlined in section 1.2, and the assumption of very small fluid-phase velocity and solute relative velocity, the problem of salinity- and temperature-driven flow is entirely described by the following set of field equations

$$\phi_f \partial_t \varrho_f + \operatorname{div}\left(\varrho_f \boldsymbol{q}_f\right) = 0, \tag{4.57}$$

$$\phi_f \partial_t(\varrho_f \omega_s) + \mathrm{div}(\varrho_f \omega_s \boldsymbol{q}_f + \boldsymbol{J}_d) = 0, \tag{4.58}$$

$$\phi_f \varrho_f \Theta \frac{\mathrm{D}_f S_f}{\mathrm{D}t} + (1-\phi_f)\varrho_r \Theta \partial_t S_r + \mathrm{div}(\boldsymbol{J}_T - \mu_{sw}\boldsymbol{J}_d) = 0, \tag{4.59}$$

which represent the balance of mass of the fluid-phase, the balance of mass of the solute, and the balance of energy of the mixture as a whole, respectively. In equations (4.57) - (4.59), the free unknowns are pressure, *p*, mass fraction, ω_s, and temperature, Θ. Indeed, the mass density ϱ_f is given in equation (4.38), the entropies S_f and S_r are deducible from the constitutive model, and the specific discharge, $\boldsymbol{q}_f$, and the fluxes $\boldsymbol{J}_d$ and $\boldsymbol{J}_T$, are specified in equations (4.30), (4.49) and (4.50). The problem is therefore closed.

4.3.1 The Boussinesq-Oberbeck approximation

Equations (4.57) - (4.59) are coupled and highly nonlinear. A considerable simplification can be obtained by framing the problem under investigation as a free convection problem, in which a fluid out of mechanical equilibrium features internal currents tending to mix the fluid and bring it to a constant temperature and constant solute mass fraction /LAN 84/. In this respect, the Boussinesq-Oberbeck approximation is adopted. This approximation consists of replacing the mass density of the fluid-phase by a constant value (i. e. $\varrho_f \approx \varrho_f^0$) in all terms of equations (4.57) - (4.59) expect for the buoyancy contribution, $\varrho_f \boldsymbol{g}$, to Darcy's law. If this is done, the field equations become

$$\mathrm{div}(\boldsymbol{q}_f) = 0, \tag{4.60}$$

$$\phi_f \varrho_f^0 \partial_t \omega_s + \mathrm{div}(\varrho_f^0 \omega_s q_f + \boldsymbol{J}_d^{BO}) = 0, \tag{4.61}$$

$$C_{\mathrm{eff}} \partial_t \Theta + \mathrm{div}(\varrho_f^0 C_f \Theta \boldsymbol{q}_f) + \mathrm{div}(\boldsymbol{J}_T^{BO}) = 0, \tag{4.62}$$

where $C_{\mathrm{eff}} = [\phi_f \varrho_f^0 C_f + (1-\phi_f)\varrho_r C_r]$ is the effective thermal capacity of the mixture, and $\boldsymbol{J}_T^{BO}$ and $\boldsymbol{J}_d^{BO}$ are the heat and mass fluxes computed within the Boussinesq-Oberbeck approximation, i. e.

$$\boldsymbol{J}_d^{BO} = -\phi_f \varrho_f^0 D[\mathrm{grad}(\omega_s) + S\omega_s(1-\omega_s)\mathrm{grad}(\Theta)], \tag{4.63}$$

$$\boldsymbol{J}_T^{BO} = -\phi_f \varrho_f^0 D Q^* \mathrm{grad}(\omega_s) - L_{TT}\mathrm{grad}(\Theta) = 0. \tag{4.64}$$

It should be noticed that the baro-diffusion coefficient, k_p, and the relative specific enthalpy, h_{sw}, are negligibly small with respect to the other quantities, and vanish identically is the fluid compressibility is neglected or, equivalently, if the relative chemical potential, μ_{sw}, is approximated by its constitutive part μ_{sw}^i. For the same reason, if $h_{sw} = 0$, the heat of transport Q can be set equal to Q^* (cf. equation (4.55)).

4.3.2 The Dufour effect

A further simplification concerns the elimination of the Dufour effect in the heat flux $\boldsymbol{J}_T^{BO}$. This is possible because of the material properties (cf. Tab. 4.3), and the magnitude of the temperature and composition gradients used here. Therefore, the heat flux in equation (4.64) can be approximated by

$$\boldsymbol{J}_T^{BO} \approx L_{TT}\mathrm{grad}(\Theta). \tag{4.65}$$

In order to see that, Q^* is computed explicitly for the case of sodium-chloride (whose molar mass is $M_s = 58.44\ \mathrm{g}\cdot\mathrm{mol}^{-1}$). By using equation (4.55) and the definition of μ_{sw}^i, one obtains

$$Q^* = S\Theta\frac{2R\Theta}{(1-\omega_s)M_s+\omega_s 2M_w} \approx \frac{2RS\Theta^2}{M_s}, \tag{4.66}$$

where the last term is the heat of transport of a very diluted solution.

Let be assumed porosity $\phi_f = 0.1$, reference mass density $\varrho_f^0 \sim 10^3\ \mathrm{kg}\cdot\mathrm{m}^{-3}$, and diffusion coefficient $D \sim 10^{-7}\ \mathrm{m}^2\cdot\mathrm{s}^{-1}$. For the physical range of temperature and mass fraction gradients, it can be shown that the coefficient giving rise to the Dufour effect in (4.64) is negligible with respect to that of pure thermal diffusion. Therefore, for the physical situations to be modelled here, the Dufour effect will be neglected in the computation of the heat flux. In conclusion, the mass flux given in equation (4.63) and the heat flux in equation (4.65) will be used for numerical simulations.

4.4 Results and discussion

All numerical examples performed here were carried out by using the software package d³f. This is a finite volume simulator for density-driven flow /FEI 99/. The software d³f is built on the toolbox UG, which provides functionality for unstructured grids, adaptive

local grid refinement, robust multigrid methods for fast solution of systems of linear equations, and parallelization of all these algorithms on MIMD-style (Multiple-Instruction-Multiple-Data) machines.

Simulations were carried out by applying the consistent velocity approximation put forward by Knabner and Frolkovič /KNA 96/.

4.4.1 Elder problem

The Elder problem /ELD 67/ is an often encountered benchmark problem in hydrogeology. It illustrates a free convection phenomenon, in which fluid-flow originates as a consequence of the variability of the fluid-phase mass density in response to composition and/or temperature inhomogeneities. Although in its original formulation, it aimed to account for thermal convection only, the Elder problem was transformed into a solute convection problem by Diersch /DIE 81/, and Voss and Souza /VOS 87/. More information can be read in /DIE 05/.

Here, for numerical simulation the scheme reported in Fig. 4.1 is considered, in which a 2D-Elder problem is shown. A domain of length 600 m and depth 150 m represents a portion of a porous medium initially filled with a fluid at rest, but exposed to a temperature gradient (see Fig. 4.2a). The resulting system is assumed to be isotropic with respect to permeability, K, and tortuosity, τ. A portion of the top boundary of the domain is in contact with a brine reservoir at a prescribed mass fraction. The brine intrudes the domain mainly because of gravity. As soon as the fluid receives the brine, a motion occurs. Since, at this stage, the motion is actually density-driven, it cannot be a potential flow and, consequently, vortices appear (see Fig. 4.2b). The brine, in turn, is mainly convected by the fluid, although it is also subject to diffusion. For this reason, here the Elder example is referred to as to a convection-dominated problem.

Since the temperature at the upper boundary of the domain (including the temperature of the brine in the reservoir) is lower than that in the domain, the intruding brine tends to cool the system down (blue regions in Fig. 4.2b). In the upper part of the domain, the temperature distribution tends to “follow” the brine, but the time-scale associated with temperature variations is larger than that characterizing the motion of the brine. Looking at equation (4.62), this can be understood by introducing the retardation factor /OLD 99/

$$R^h = \frac{\phi_f \varrho_f C_f + (1-\phi_f)\varrho_r C_r}{\phi_f \varrho_f C_f} = \frac{C_{\text{eff}}}{\phi_f \varrho_f C_f}, \tag{4.67}$$

which can be defined by dividing the thermal convection term by the effective thermal capacity of the mixture. Physically, it describes the fact that the rock is able to store thermal energy whereas it is not able to store the brine. Oldenburg and Pruess /OLD 99/ have shown that the retardation factor is $R^h = 6.7$, this meaning that the thermal front moves at approximately 1/7 the velocity of the brine front.

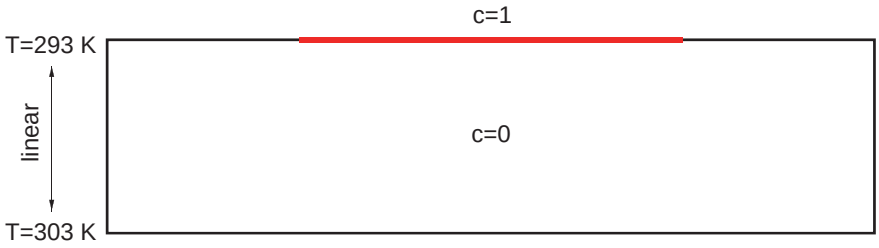
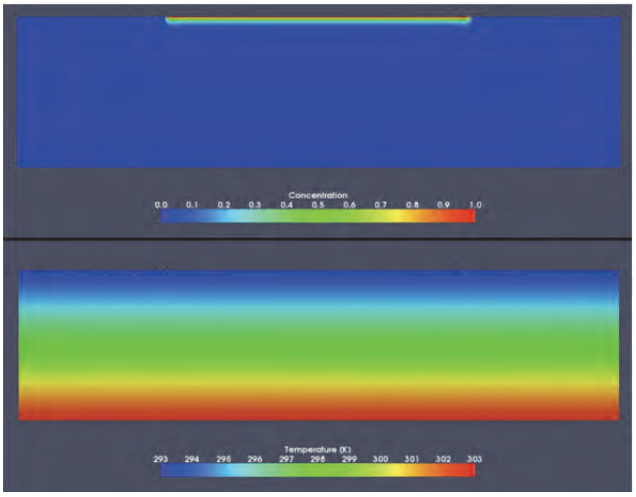


Fig. 4.1 Scheme of an Elder problem

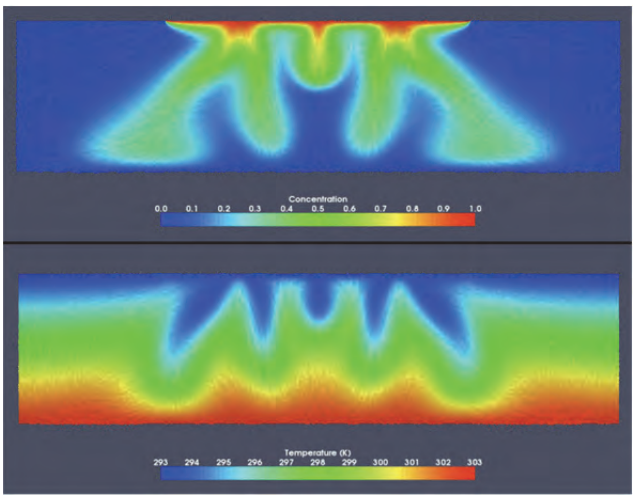
The evolution of temperature, however, contributes to change the mass-density of the fluid, this resulting into a salinity- and temperature-driven flow. In this example, equation (4.38) is linearized, and used the following expression for the mass density of the fluid-phase

$$\varrho_f(\omega_s, \Theta) = 1000 + 200\,\omega_s - 0.3\,(\Theta - 293), \tag{4.68}$$

with $[\varrho_f] = \mathrm{kg} \cdot \mathrm{m}^{-3}$. It should be remarked that, since the Soret effect represents only a small contribution to the diffusive flux $\boldsymbol{J}_d^{BO}$, it turns out to be completely negligible in a convection-dominated problem like the one studied in this example.



a)



b)

Fig. 4.2 Thermohaline Elder problem:
a) Initial configuration
b) evolution of brine mass fraction and temperature

4.4.2 Evolution of a solute parcel

This example was taken as a benchmark problem from Oldenburg and Pruess /OLD 99/ and consists of two test cases, in each of which the evolution of a parcel of solute is studied. The differences between the results in /OLD 99/ and those reported here are due to the better refinement of the grid used in the numerical computations.

The experiments are prepared by inserting, for each case, a square-shaped parcel of fluid at a specified temperature, Θ^p, and solute mass fraction, ω_s^p, in a domain of length 2500 m and depth 2500 m. The domain represents a 2D reservoir consisting of an iso-

tropic porous medium saturated by an initially homogeneous fluid (i. e. $\omega_s^R = 0$) at uniform temperature ($\Theta^R = 473\ \mathrm{K}$), whose mass density is $\varrho_f^R = 875\ \mathrm{kg \cdot m^{-3}}$ (see Fig. 4.3). The experimental set-up is designed so to expect negative buoyancy in the first case and positive buoyancy in the second case.

The left and right boundaries of the domain are assumed to be adiabatic (i. e. $\boldsymbol{J}_T \cdot \boldsymbol{n} = 0$), and impermeable for both fluid-phase as a whole and brine (i. e. $\varrho_f \boldsymbol{q}_f \cdot \boldsymbol{n} = 0$ and $\boldsymbol{J}_d \cdot \boldsymbol{n} = 0$), while the upper and lower boundaries are kept at constant temperature, $\Theta = 473\ \mathrm{K}$, and mass fraction $\omega_s = 0.0$. Furthermore, no-flow conditions are assumed at the upper and lower boundaries (i. e. $\varrho_f \boldsymbol{q}_f \cdot \boldsymbol{n} = 0$), and the pressure, *p*, is set equal to atmospheric pressure only at the upper corners of the domain.

In the first case, the parcel is characterized by the following data: $\omega_s^p = 0.55$, $\Theta^p = 573\ \mathrm{K}$, and $\varrho_f^p = 919\ \mathrm{kg \cdot m^{-3}}$. Since $\varrho_f^p > \varrho_f^R$, and initial negative buoyancy is observed, i. e. the motion of the parcel is initially directed downwards. Fig. 4.4 and Fig. 4.5 show the distribution of mass fraction, temperature, and mass density at two different times. Each picture is symmetric with respect to the vertical symmetry axis, which ideally cuts the domain into a left and right region. Because of the thermal retardation facto defined in equation (4.67), the solute is transported downward faster than heat (cf. Fig. 4.4 and Fig. 4.5). The distribution of the solute mass fraction (denoted by *C* in the figure) features eighth symmetric “fingers” (cf. Fig. 4.5a), which are generated by negative buoyancy, and are accelerated by density-driven flow. The leading phenomenon is thus convection. The picture showing the distribution of mass density is very similar (cf. Fig. 4.5c). The fingers are due to the brine distribution, while the upper part of the picture reflects the density configuration in response to both temperature and solute distributions. Because of the delay with which temperature evolves in time, there is separation of the brine plume from the thermal plume /OLD 99/. In other words, it is possible to observe a “splitting” of the original parcel in two parcels. The first one, which describes the evolution of the brine, moves downward because of negative buoyancy, while the second one, which describes the evolution of temperature, tends to move upward because of thermal buoyancy (see Fig. 4.5b). For simulations, the mass density given in equation (4.38) was used. The parameters were extrapolated from the data provided by Oldenburg and Pruess /OLD 99/.

In the second case, the parcel is characterized by the following data: $\omega_s^p = 0.47$, $\Theta = 573\ \mathrm{K}$, and $\varrho_f^p = 831\ \mathrm{kg \cdot m^{-3}}$. Since $\varrho_f^p < \varrho_f^R$, an initial positive buoyancy is ob-

served, i. e. the motion of the parcel is initially directed upward (Fig. 4.6). This initial upward motion occurs in the middle of the parcel. Fig. 4.7 and Fig. 4.8 show the distribution of mass fraction, temperature, and mass density at two different times. Also in this case, the pictures are symmetric with respect to the vertical axis passing through the centre of the domain. The transport of brine occurs over a time-scale faster than that of heat because of the thermal retardation factor (cf. Fig. 4.7 and Fig. 4.8). The distribution of the brine mass fraction features two symmetric fingers in the lower part of the parcel, and a “fountain-like” motion in the upper part of the parcel and on its left and right sides.

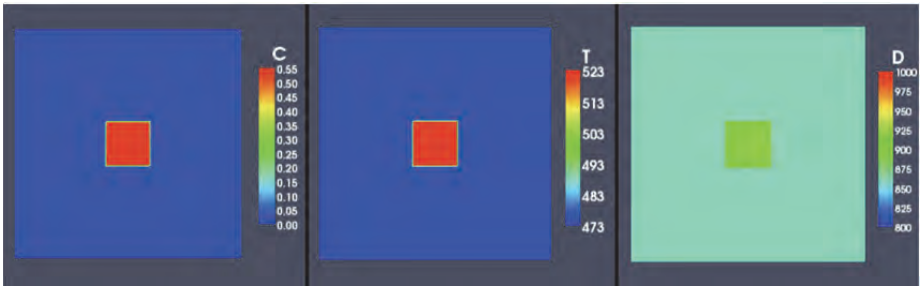


Fig. 4.3 Initial configuration of negative buoyancy:
a) mass fraction; b) temperature; c) fluid density

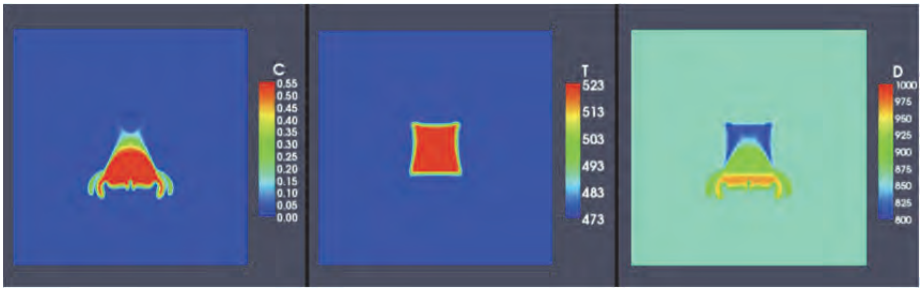


Fig. 4.4 Evolution of the parcel after 10 yrs.

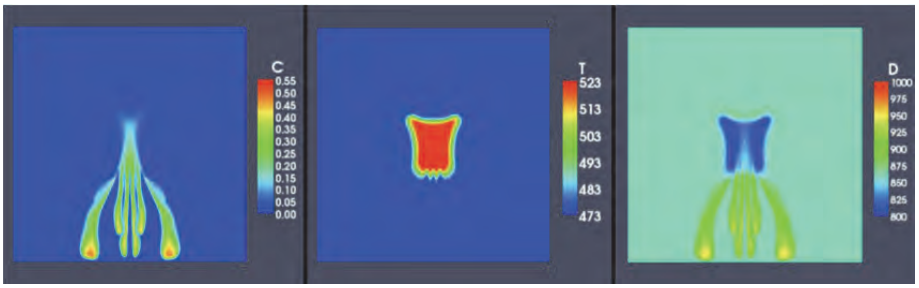


Fig. 4.5 Evolution of the parcel after 20 yrs:
a) mass fraction; b) temperature; c) density

This is due to the vortices that accompany the upward motion of the brine, and persist in the interior of the parcel. This behaviour, and the delayed evolution of temperature are responsible for forming a marked lid in the density plot (see Fig. 4.8c), which shows a strong contrast between the inner region of the parcel (where the density is lower) and its boundary (where the density is higher). A mathematical explanation of this result can be obtained by showing that the curl of the specific discharge $\mathrm{curl}(\boldsymbol{q}_f)$, is non-zero because of the gradients of temperature and mass fraction.

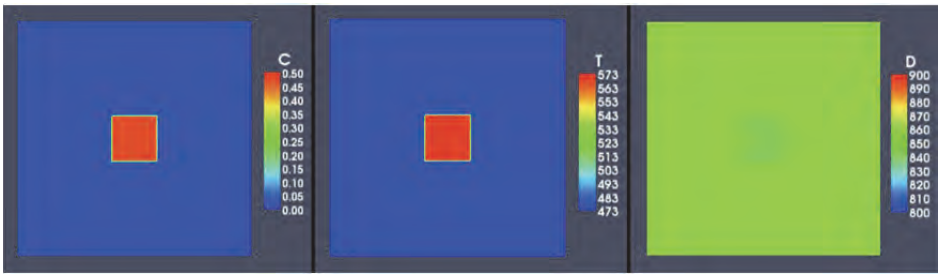


Fig. 4.6 Initial configuration of the parcel for positive buoyancy:
a) mass fraction; b) temperature; c) fluid density

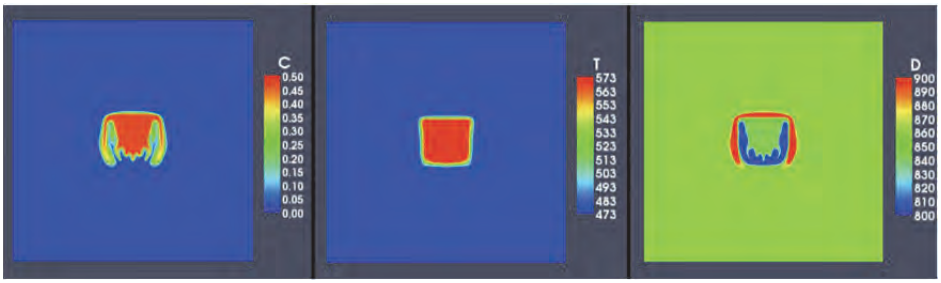


Fig. 4.7 Evolution of the parcel after 10 yrs:
a) mass fraction; b) temperature; c) fluid density

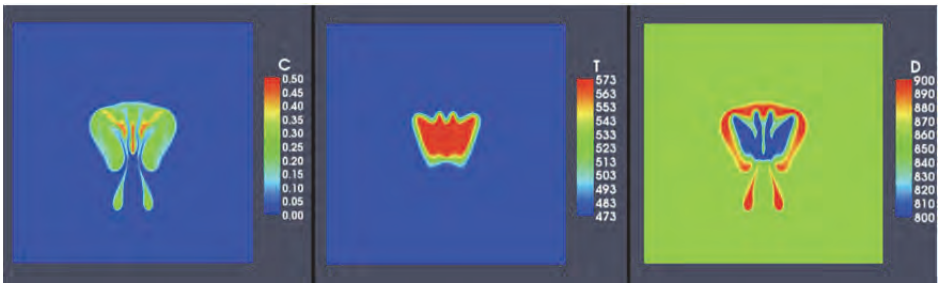


Fig. 4.8 Evolution of the parcel after 20 yrs:
a) mass fraction; b) temperature; c) fluid density

Also in these two cases, the Soret effect is negligible because it represents only a small contribution to diffusion in a convection-dominated problem.

4.4.3 Mixing problem

This benchmark problem was taken from /OLD 00/, and consists of studying the evolution of the mixing zone formed between two layers of solution at different temperature and with different mass fraction (see Fig. 4.9). The main difference between /OLD 00/ and this approach is that the Soret effect is considered in the diffusive flux $\boldsymbol{J}_d$. It should be remarked that, since the lower layer is heavier than the upper one, the problem under investigation is expected to be diffusion-dominated

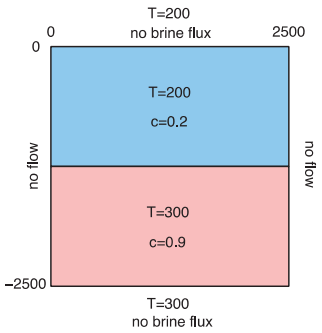


Fig. 4.9 Initial configuration of the mixing problem

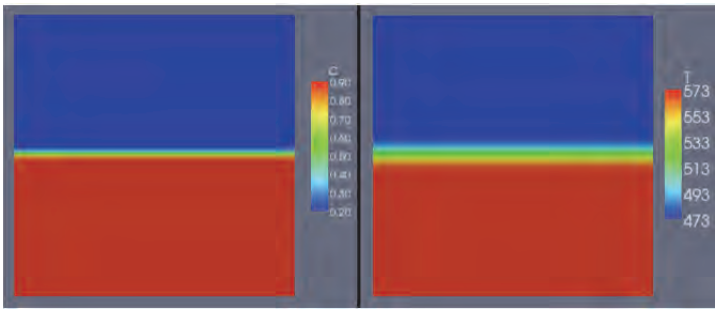


Fig. 4.10 Distribution of mass fraction and temperature after t = 150 yrs. The diffusion of heat (right) is faster

In Fig. 4.10, the distribution of mass fraction and temperature are shown at time *t* = 150 yrs. Note that the time-scale of thermal diffusion is smaller than that associated with the brine. This is due to the contrast between the diffusivity coefficient, $D \sim$

$10^{-7}\ \mathrm{m}^2 \cdot \mathrm{s}^{-1}$, and the thermal diffusivity $D_T = L_{TT}/C_{eff} \sim 6.42 \cdot 10^{-7}\ \mathrm{m}^2 \cdot \mathrm{s}^{-1}$ (cf. Tab. 4.3).

A qualitative estimate of the role of thermodiffusion is shown in Fig. 4.11, which was obtained by considering the Soret effect in the numerical calculations.

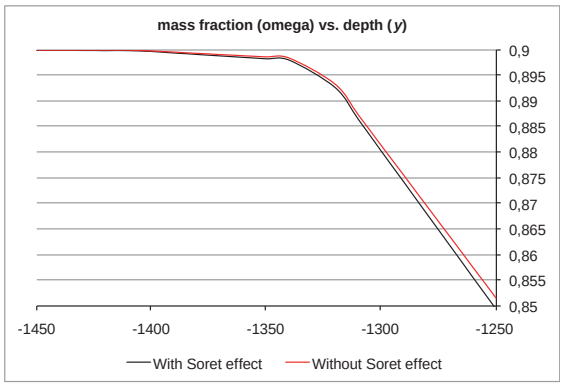


Fig. 4.11 Mass fraction vs. depth (t = 150 yrs)

In Fig. 4.11, the profile of brine mass fraction is shown as a function of height. Computations were performed in two cases: the Soret effect was disregarded in the first case (red curve in Fig. 4.11), and considered in the second case (black curve in Fig. 4.11). The Soret coefficient in equation (4.63) was taken equal to $S = 5 \cdot 10^{-3}\ ^{\circ}\mathrm{C}^{-1}$. This value was extrapolated from the values given in /CEL 05/. The difference in the profiles of mass fraction obtained with and without the Soret effect is barely visible. However, a closer inspection reveals that the mixing zone in the second case is slightly wider than that in the first case.

4.4.4 Soret cell

A Soret cell can be thought of as a cubic device containing a binary fluid mixture initially of uniform composition (i. e. uniform solute mass fraction), enclosed between two horizontal plates at different temperatures. In the case of thermodiffusion in porous media, a similar problem was studied in /BEN 01/. In order to minimize the convection currents in the mixture, and ease the attainment of the steady state (i. e. the state in which the concentration and temperature gradients balance each other), the upper plate is kept at a temperature higher than the lower one /TYR 56/. Here, this “experimental” set-up is slightly modified by letting the Soret cell be occupied by a porous me-

dium of constant porosity and saturated by a fluid-phase made of water and brine. Furthermore, the physical dimension of the Soret cell is big enough to enclose a porous medium oh hydrogeological interest. Also in this case, the problem under investigation is diffusion-dominated.

For the sake of simplicity, the case of a 2D Soret cell is considered. Initially, the interior of the domain possesses a uniform mass fraction $\omega_s^{in} = c = 0.5$, and uniform temperature $\Theta^{in} = 323\ \mathrm{K}$. The upper and lower plates are kept at temperature $\Theta^{upper} = 353\ \mathrm{K}$ and $\Theta^{lower} = 293\ \mathrm{K}$, respectively, and no solute flux is assumed, i. e. $\varrho_f \boldsymbol{q}_f \cdot \boldsymbol{n} = 0$, and $\boldsymbol{J}_d \cdot \boldsymbol{n} = 0$. For the given range of temperature, the Soret coefficient was set equal to $S = 2 \cdot 10^{-3}\ °\mathrm{C}^{-1}$. On the other hand, the lateral boundaries are assumed to be impermeable (i. e. $\varrho_f \boldsymbol{q}_f \cdot \boldsymbol{n} = 0$, and $\boldsymbol{J}_d \cdot \boldsymbol{n} = 0$), and adiabatic ($\boldsymbol{J}_T \cdot \boldsymbol{n} = 0$). Furthermore, mass fraction is kept constant to its initial value at the middle points of the lateral boundaries. By these initial and boundary conditions (see Fig. 4.12) the problem is actually one-dimensional.

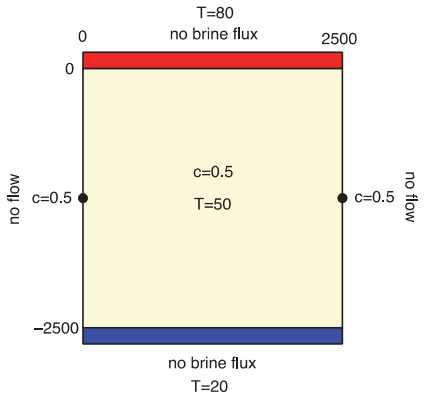


Fig. 4.12 Soret cell. Initial and boundary conditions used in simulations

Although the performed simulations provide results for pressure, *p*, temperature, Θ, and mass fraction, ω_s, here for brevity only the profile of mass fraction is shown (see Fig. 4.13). The result is a “snapshot” in time towards the steady state solution. From the upper boundary of the Soret cell, the mass fraction increases until a local maximum is reached. For symmetry reasons, the mass fraction decreases from the lower boundary until a local minimum is reached, and then increases towards the middle of the cell. This behaviour depends on the interaction between the mass fraction and the thermodiffusion of the solute. Indeed, although thermal diffusion is faster than the thermodiffusion of the solute, time is required in order to “build” up the temperature gradient compatible with boundary conditions, and characterizing the steady state of the system.

Therefore, the temperature distribution remains close to its initial value near the centre of the cell, while two gradients are generated in a neighbourhood of the upper and lower plates. These gradients, and the form of the diffusion flux $\boldsymbol{J}_d^{BO}$, are responsible for moving the solute downward. Consequently, on its way to the steady state, the profile of the solute mass fraction features two symmetric "bumps", which tend to disappear as soon as temperature reaches the steady-state solution. At the steady state, if the specific discharge, $\boldsymbol{q}_f$, is neglected, the profile of mass fraction can be computed analytically. In consistence with the boundary conditions, one finds

$$\omega_s(y) = \frac{\omega_s^{in} \exp\left[-\frac{S\Theta_c}{\ell}(y-y_m)\right]}{1+\omega_s^{in}\left\{\exp\left[-\frac{S\Theta_c}{\ell}(y-y_m)\right]-1\right\}}, \tag{4.3}$$

where $\Theta_c = \Theta_{upper} - \Theta_{lower}$, ℓ is the depth of the domain, and $y_m = \ell/2$.

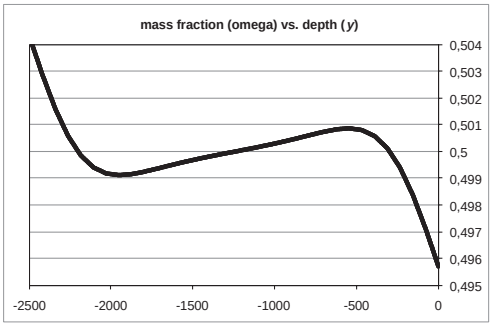


Fig. 4.13 Numerically computed profile of mass fraction in a Soret cell

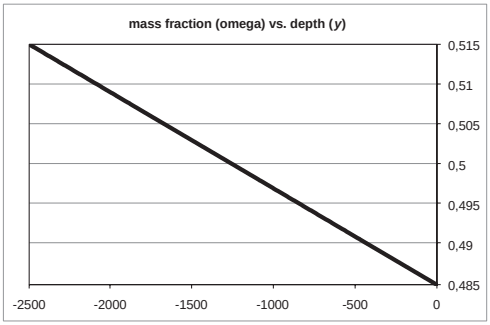


Fig. 4.14 Analytically computed profile of mass fraction in a Soret cell

The plot of the stationary solution, shown in Fig. 4.14, resembles a straight line because the coefficient is much smaller than unity. This shows that, in this set-up, the

coefficient $(S\Theta_c)/\ell$ produces a maximum deviation from the initial solute concentration of 3 %, and it is thus observable.

The results obtained in the sections above should be regarded as qualitative. Indeed, the scarcity of published experimental data in the context of thermodiffusion and variable-density flow has forced that values to the Soret coefficient used in these simulations are given, which may not be realistic.

Tab. 4.3 Parameters used for the numerical simulations

Quantity	Units	Value
Porosity, ϕ_f	–	0.10
Permeability, $\boldsymbol{K} = K\mathbf{I}$	m^2	$5.0 \cdot 10^{-12}$
Viscosity, μ	Nsm^{-2}	$1.0 \cdot 10^{-4}$
Molecular diffusivity (including porosity), *D*, D_m	$m^2 s^{-1}$	$1.0 \cdot 10^{-8}$
Longitudinal dispersion length, α_ℓ	m	0.00
Transversal dispersion length, α_t	m	0.00
Thermal conductivity, L_{TT}, Λ_c	$Wm^{-1}K^{-1}$	1.8
Tortuosity, $\boldsymbol{\tau} = \tau\mathbf{I}$	–	1.0
Heat capacity of rock, C_r	$J\ kg^{-1}K^{-1}$	1000
Heat capacity of the fluid-phase, C_f	$J\ kg^{-1}\ K^{-1}$	4184
Mass density of rock, ϱ_r	$kg\ m^{-3}$	2650
Gravity acceleration, g	$m\ s^{-2}$	9.81
Reference density of pure water, $\bar{\varrho}_{pw}$	$kg\ m^{-3}$	800
Reference density of pure brine, $\bar{\varrho}_{pb}$	$kg\ m^{-3}$	1000
Thermal expansion coefficient of water, α_w	K^{-1}	$1.45 \cdot 10^{-3}$

Quantity	Units	Value
Thermal expansion coefficient of brine, α_b	K^{-1}	$8.13 \cdot 10^{-4}$
Reference temperature for water, Θ_w	K	523.15
Reference temperature for brine, Θ_b	K	563.15

4.5 Three-dimensional simulation of the thermohaline-driven buoyancy of a brine parcel

A three-dimensional numerical simulation of the thermohaline-driven buoyancy of a brine parcel immersed in an initially homogeneous porous medium of hydrogeological interest is provided here. The purpose is to improve the understanding of the thermohaline flow through the 3D visualization of the evolving patterns generated by the distribution of brine, temperature and fluid density in the porous medium. A possible physical interpretation of the results is proposed, which are obtained within the approximations usually employed in the context of salinity- and temperature-driven flow.

Our purpose is to study the solutal and thermal buoyancy of brine in the presence of strong thermal gradients. In order to do that, the same numerical experiment as in /OLD 99/ is re-proposed. However, three-dimensional numerical simulations on very fine grids are provided. The experiment consists of placing a relative small brine parcel in an idealized porous medium saturated by a fluid, which is assumed to be homogeneous and in thermo-mechanic equilibrium prior to the insertion of the parcel. The brine parcel is actually a mixture of brine and pure water with specified brine mass fraction. The strong thermal gradients are generated by requiring that the parcel temperature is greater than that of the surrounding system (homogeneous fluid and porous medium).

Following /OLD 99/, two different experimental set-ups were prepared which, depending on the initial temperature and brine mass fraction of the brine parcel, yield either negative buoyancy or positive buoyancy. The 3D visualization of the evolving patterns generated by the distribution of brine, temperature, and fluid density in the region of observation are obtained by solving the mass balance equations of the brine and the fluid-phase as a whole, and the energy balance equation of the whole system (fluid-phase and porous medium). The numerical simulations were performed by using an

enhanced version of the software package d³f, extended to include temperature in addition to pressure and brine mass fraction.

4.5.1 Mathematical model

In the case of single-phase flow, the fundamental kinematic entities associated with the fluid-phase are given by the specific discharge, $\boldsymbol{q}$, and the brine diffusive mass flux $\boldsymbol{J}_d$. The former vector field describes the filtration velocity of the fluid-phase as a whole, while the mass flux $\boldsymbol{J}_d$ accounts for the motion of the brine relative to the centre of mass of the fluid-phase. If the dissipative interaction forces between the fluid and the solid-phase are assumed to be of genuine mechanical nature (i. e. they are independent on the macroscopic thermal gradient), and the hypotheses of negligible inertial forces (laminar motion), macroscopically inviscid fluid-phase (absence of Brinkmann's correction), and small fluid-phase velocity (absence of Forchheimer's correction) are imposed, the momentum balance law of the fluid-phase allows for expressing the specific discharge, $\boldsymbol{q}$, by means of Darcy's law, i. e.

$$\boldsymbol{q} = -\frac{\mathbf{K}}{\mu}[\mathrm{grad}(p) - \varrho \boldsymbol{g}], \tag{4.69}$$

where **K** is the permeability tensor of the porous medium, μ is the fluid-phase dynamic viscosity, p is pressure, and μ is the fluid-phase mass density. Although the viscosity μ is a function of temperature and the composition of the fluid-phase, it will be considered constant in the following. A reasoning analogous to that leading to equation (4.69) yields to express the brine diffusive mass flux, $\boldsymbol{J}_d$, through Fick's law, i. e.

$$\boldsymbol{J}_d = -\varrho \mathbf{D}\mathrm{grad}(\omega), \tag{4.70}$$

where ω is the brine mass fraction, and **D** is the diffusion-dispersion tensor. The tensor **D** is taken as the sum of a diffusive and a dispersive contribution. The Scheidegger's form of the dispersive contribution is commonly used, and the tensor **D** is written as

$$\mathbf{D} = D_m \boldsymbol{\tau} + \mathbf{D}_{\boldsymbol{M}} = D_m \boldsymbol{\tau} + a_t |\boldsymbol{q}| \mathbf{I} + (a_\ell - a_t) \frac{\boldsymbol{q} \otimes \boldsymbol{q}}{|\boldsymbol{q}|}, \tag{4.71}$$

where D_m is the brine scalar molecular diffusivity, $\boldsymbol{\tau}$ is the tortuosity tensor, and $\mathbf{D}_{\boldsymbol{M}}$ is the Scheidegger's mechanical dispersion tensor. In the definition of $\mathbf{D}_{\boldsymbol{M}}$, the quantities a_t and a_ℓ are the transversal and longitudinal dispersion lengths, respectively, and $\mathbf{I}$

denotes the identity tensor. Accepting all approximations introduced so far, the problem of thermohaline flow requires to consider the following three equations: the balance of mass of the fluid-phase as a whole, the balance of mass of the brine, and the energy balance law for the mixture as a whole. These three scalar equations read

$$\partial_{\boldsymbol{t}}(\phi\varrho) + \operatorname{div}(\varrho\boldsymbol{q}) = 0, \tag{4.72}$$

$$\partial_{\boldsymbol{t}}(\phi\varrho\omega) + \operatorname{div}(\varrho\omega\boldsymbol{q} + \boldsymbol{J}_d) = 0, \tag{4.73}$$

$$\partial_{\boldsymbol{t}}\left[\left(\phi C_f + (1-\phi)\varrho_s C_s\right)\Theta\right] + \operatorname{div}\left(\varrho C_f \Theta\boldsymbol{q} + \boldsymbol{J}_T\right) = 0, \tag{4.74}$$

where ϕ is porosity, C_f and C_s are the heat capacities per unit mass of the fluid- and solid-phase, respectively, ϱ_s is the constant density of the solid-phase, and $\boldsymbol{J}_T$ is the heat flux of the mixture as a whole. If the validity of Fourier's law of heat transfer is assumed, then $\boldsymbol{J}_T$ is expressed by

$$\boldsymbol{J}_T = -\boldsymbol{\Lambda}\operatorname{grad}(\Theta), \tag{4.75}$$

where $\boldsymbol{\Lambda}$ is the hydrodynamic thermo-dispersion tensor of the mixture as a whole. Following /DIE 02/, the positive-definite tensor $\boldsymbol{\Lambda}$ can be written as

$$\boldsymbol{\Lambda} = \boldsymbol{\Lambda}_c + \boldsymbol{\Lambda}_M = \boldsymbol{\Lambda}_{\boldsymbol{c}} + C_f\varrho\mathbf{D}_M, \tag{4.76}$$

where $\boldsymbol{\Lambda}_c$ and $\boldsymbol{\Lambda}_M$ represent the conductive and mechanical part of the thermo-dispersion tensor, respectively. Since the tensor $\boldsymbol{\Lambda}_c$ represents the thermal conductivity of the whole mixture, it has to enclose the thermal conduction properties of both the fluid- and the solid-phase. For this reason, the tensor $\boldsymbol{\Lambda}_c$ is expressed as the mean value of the thermal conductivities of the two phases. Bear /BEA 72/ defined the tensor $\boldsymbol{\Lambda}_c$ as the weighted sum of the thermal conductivities of the fluid- and the solid-phase, the weights being given by the volumetric mass fractions of the phases, i. e.

$$\boldsymbol{\Lambda}_c = \phi\boldsymbol{\Lambda}_f + (1-\phi)\boldsymbol{\Lambda}_s. \tag{4.77}$$

However, other definitions are also used (cf., for example, /HOL 01/).

The saturation constraint, and the hypotheses of incompressible and rigid solid-phase imply that the volume fraction of the fluid-phase is constant in time and equal to the porosity ϕ. A further simplifying assumption is that ϕ is also constant in space. Equa-

tions (4.69) - (4.77) make up a system of simplified field equations to be solved. The closure of this system of equations is obtained by prescribing either pressure or the fluid-phase mass density as a constitutive function of the other state variables. In order to model thermohaline flow, the fluid-phase mass density, ϱ, is chosen as a constitutive function of temperature, Θ, and brine mass fraction, ω, i. e. $\varrho = \varrho(\omega, \Theta)$. This choice implies that the free unknowns in equations (4.72) - (4.74) are given by pressure, *p*, mass fraction, ω, and temperature, Θ. Furthremore, pressure has to be understood as a Lagrange multiplier, and the heat capacities C_f and C_s are defined at constant pressure.

4.5.2 Description of the problem and specific assumptions

4.5.2.1 Preparation oft he experiment: Initial and boundary conditions

A fixed cubic region, $\mathcal{R}$, is considered, filled with a porous medium of uniform porosity, ϕ, and saturated by a fluid (see Fig. 4.15). Let *L* be the length of the edge of the cube, and imagine to place in the centre of the region $\mathcal{R}$ a cubic “brine parcel”, $\mathcal{P}$, which occupies the volume $\mathrm{Vol}(\mathcal{P}) = \ell^3$, and has its faces parallel to the faces of the outer cube $\mathcal{R}$. The brine parcel, $\mathcal{P}$, consists of a very concentrated mixture of brine and water characterized by uniform brine mass fraction, $\omega_{\mathcal{P}}$, and uniform initial temperature $\Theta_{\mathcal{P}}$. The partial domain $\mathcal{Q} = \mathcal{R}/\mathcal{P}$ is filled with pure water (i. e. $\omega_{\mathcal{Q}} = 0$), and is assumed to be in thermal equilibrium with the porous medium at temperature $\Theta_{\mathcal{Q}} = 0$. It is required that the initial condition $\Theta_{\mathcal{Q}} \neq \Theta_{\mathcal{P}}$ be satisfied. Consistently with the description given so far, the initial conditions of the prepared numerical experiments are:

$$\omega(0, \boldsymbol{r}) = \begin{cases} \omega_{\mathcal{P}}, & \text{if } \boldsymbol{r} \in \mathcal{P}, \\ \omega_{\mathcal{Q}}, & \text{if } \boldsymbol{r} \in \mathcal{Q}, \end{cases} \tag{4.78}$$

$$\Theta(0, \boldsymbol{r}) = \begin{cases} \Theta_{\mathcal{P}}, & \text{if } \boldsymbol{r} \in \mathcal{P}, \\ \Theta_{\mathcal{Q}}, & \text{if } \boldsymbol{r} \in \mathcal{Q}. \end{cases} \tag{4.79}$$

It should be remarked that, because of the constitutive assumption specifying the fluid-phase mass density, the time derivative of the pressure does not feature in equations (4.72) - (4.74). Therefore, no initial condition is required for determining the unknown field *p*. The initial conditions (4.78) and (4.79) reflect the 3D-generalization of the benchmark problem formulated by Oldenburg and Pruess /OLD 99/.

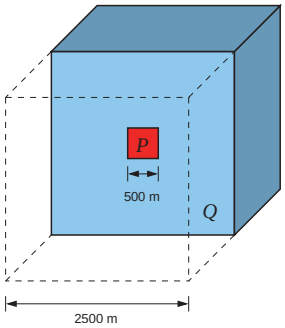


Fig. 4.15 Computation domain and parcel “immersed” in it

The four lateral faces of the outer cube are assumed to be impermeable and adiabatic. This implies that the following boundary conditions apply:

$$\varrho \boldsymbol{q} \cdot \boldsymbol{n} = 0, \quad (\varrho \omega \boldsymbol{q} + \boldsymbol{J}_d) \cdot \boldsymbol{n} = 0, \quad \text{on } \mathcal{B} = \cup_{i=1}^{4} \mathcal{F}_i \text{ (impermeable walls)}, \tag{4.80}$$

$$\boldsymbol{J}_T \cdot \boldsymbol{n} = 0, \quad \text{on } \mathcal{B} = \cup_{i=1}^{4} \mathcal{F}_i \text{ (adiabatic walls)}, \tag{4.81}$$

where $\mathcal{F}_i$ is the *i*-th face of the lateral boundary $\mathcal{B}$ of the outer cube $\mathcal{R}$.

The upper and lower faces of the outer cube, identified by the planes $z = 0$ and $z = L$, respectively, are equipped with the Dirichlet boundary conditions written below:

$$\Theta = \Theta_{\mathcal{Q}}, \quad \omega = \omega_{\mathcal{Q}} = 0, \quad \text{on the upper face}, \mathcal{F}_0, \tag{4.82}$$

$$\Theta = \Theta_{\mathcal{Q}}, \quad \omega = \omega_{\mathcal{Q}} = 0, \quad \text{on the lower face}, \mathcal{F}_L, \tag{4.83}$$

$$\varrho \boldsymbol{q} \cdot \boldsymbol{n} = 0, \quad \text{on } \mathcal{F}_0 \cup \mathcal{F}_L, \text{and } p = p_0 = 0 \text{ if } \boldsymbol{r} \in \mathcal{V}, \tag{4.84}$$

where $\mathcal{V} = \cup_{k=1}^{4} \{\boldsymbol{r}_k\}$ is the set of all vertices of the upper face $\mathcal{F}_0$.

Following /OLD 99/, two different physical situations are considered associated with initial positive and negative buoyancy, respectively. In the first case, the initial mass fraction and temperature of the parcel are set equal to $\omega_{\mathcal{P}} = 0.47$ and $\Theta_{\mathcal{P}} = 573.15\ \mathrm{K}$; accordingly, the fluid-phase mass density in the parcel is $\varrho_{\mathcal{P}} = 831.0\ \mathrm{kg \cdot m^{-3}}$. In the case of negative buoyancy, the initial values $\omega_{\mathcal{P}} = 0.55$ and $\Theta_{\mathcal{P}} = 523.15\ \mathrm{K}$ are chosen, which amount to an initial fluid-phase mass density $\varrho_{\mathcal{P}} = 919.0\ \mathrm{kg \cdot m^{-3}}$. In both cases, the initial temperature of the subdomain $\mathcal{Q}$ is chosen to be $\Theta_{\mathcal{Q}} = 473.15\ \mathrm{K}$, corresponding to a mass density of $\varrho_{\mathcal{Q}} = 875.0\ \mathrm{kg \cdot m^{-3}}$. The numerical values of the parameters

used here are summarized in Tab. 4.4, and are taken from /OLD 99/. It should be remarked, however, that the boundary condition (4.84) is different from that chosen in /OLD 99/. This is due to a small difference in the formulation of the problem. Indeed, similarly to Elder's problem, pressure is prescribed to satisfy Neumann conditions almost everywhere on the boundary surfaces (i. e. on $\partial\mathcal{R}/\mathcal{V}$), and to satisfy Dirichlet conditions on the set of upper vertices denoted by $\mathcal{V}$. The reason for such a choice of boundary conditions is that, while here a flow region with impermeable boundary is considered, Oldenburg and Pruess /OLD 99/ impose a hydrostatic pressure distribution in a domain in which only the lateral walls are impervious. Furthermore, since only the pressure gradient features in the governing equations and no dependence on the pressure itself is accounted for, the quantity *p* can be defined up to an arbitrary additive constant, such that $p(\boldsymbol{r}_k) = 0$ for all $\boldsymbol{r}_k \in \mathcal{V}$.

4.5.2.2 Specific assumptions

In order to simplify the problem at hand, the following further hypotheses are employed: (a) permeability, **K**, tortuosity, $\boldsymbol{\tau}$, and thermal conductivity, $\boldsymbol{\Lambda}_c$, are spherical and homogeneous tensors; (b) the role of mechanical dispersion is neglected; (c) the assumption of volume additivity is enforced for the fluid-phase constituents.

The hypotheses (a) and (b) allow for rewriting Darcy's, Fick's and Fourier's laws as:

$$\boldsymbol{q} = -\frac{K}{\mu}[\mathrm{grad}(p) - \varrho\boldsymbol{g}], \tag{4.85}$$

$$\boldsymbol{J}_d = -\varrho D_m \tau\, \mathrm{grad}(\omega), \tag{4.86}$$

$$\boldsymbol{J}_T = -\Lambda_c \mathrm{grad}(\Theta), \tag{4.87}$$

where K, D_m, Λ_c are given scalar constants. The hypothesis (c) leads to the following definition of the fluid-phase mass density:

$$\varrho(\omega, \Theta) = \frac{\varrho_{pw}(\Theta)\varrho_{pb}(\Theta)}{\varrho_{pb}(\Theta) - [\varrho_{pb}(\Theta) - \varrho_{pw}(\Theta)]\omega}. \tag{4.88}$$

By introducing two uniform reference temperatures, Θ_w and Θ_b, and letting $\bar{\varrho}_{pw}$, $\bar{\varrho}_{pb}$ and α_w, α_b denote the reference mass densities and constant thermal expansion coefficients of water and brine, respectively, the functions $\varrho_{pw}(\Theta)$ and $\varrho_{pb}(\Theta)$ can be ex-

pressed as $\varrho_{pw}(\Theta) = \bar{\varrho}_{pw} \exp\{-\alpha_w[\Theta - \Theta_w]\}$ and $\varrho_{pb}(\Theta) = \bar{\varrho}_{pb} \exp\{-\alpha_b[\Theta - \Theta_b]\}$, respectively.

A further reduction of the complexity of the computation is gained by solving equations (4.72) - (4.74) within the Boussinesq-Oberbeck approximation /DIE 05/, /JOH 02/. This approximation consists of maintaining the compressibility of the fluid-phase only in the buoyancy term $\varrho \boldsymbol{g}$ featuring in Darcy's law, while replacing the mass density ϱ with a constant reference value anywhere else it appears. Although the Boussinesq-Oberbeck approximation may be quite useful in several cases, it is not used here.

4.5.3 Numerical simulations

The governing equations (4.72) - (4.74) constitute a set of three scalar, coupled and non-linear partial differential equations in the three unknowns p, ω, and Θ. This set of equations is solved in the flow region $\mathcal{R}$ numerically by employing the "vertex centred" Finite Volume Method, because it carries the conversation properties of the continuous equations over their discretized form. For the sake of conciseness the method is exposed in the two-dimensional case, although the flow region, $\mathcal{R}$, and the results shown here are three-dimensional. The procedure reported below is based on the paper by Frolkovic /FRO 96/ and was extended here in order to include temperature.

It shall be assumed that a conforming grid covers the flow region, and that the elements of this grid are quadrilaterals. In order to perform the "vertex centred" finite volume scheme, a dual grid is constructed by connecting the barycentre of each element with the midpoints of the element edges (cf. Fig. 4.16). By following this procedure, the generic vertex $\boldsymbol{x}_i$ of the grid is associated with a finite volume V_i, which represents a control volume of the dual grid. The index i ranges from 1 to M, where M is the number of grid nodes.

Equations (4.72) - (4.74) share a similar structure. Indeed, each of them can be written in the form

$$\partial_t\big(F(\sigma)\big) + \mathrm{div}\big(\boldsymbol{\Phi}(\eta, \boldsymbol{\kappa})\big) = 0, \tag{4.89}$$

where $F(\sigma)$ and $\boldsymbol{\Phi}(\eta, \boldsymbol{\kappa})$ are two "auxiliary" functions denoting a generalized "storage" term and flux, respectively, while σ, η, and $\boldsymbol{\kappa}$ define the following collections of fields:

$\sigma = (\omega, \Theta)$, $\eta = (p, \omega, \Theta)$, and $\boldsymbol{\kappa} = \mathrm{grad}(\eta)$. Equation (4.89) is integrated over the generic finite volume V_i, and Gauss' Theorem is applied. This yields

$$\int_{V_i} \partial_t(F(\sigma))\mathrm{d}v + \int_{\partial V_i} \boldsymbol{\Phi}(\eta, \boldsymbol{\kappa}) \cdot \boldsymbol{n}\, \mathrm{d}a = 0. \tag{4.90}$$

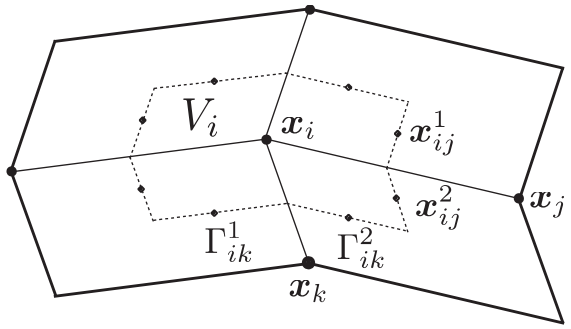


Fig. 4.16 Exemplified two-dimensional grid and dual mesh

Let now $\sigma_i(t) = \sigma(t, \boldsymbol{x}_i)$ denote the time-dependent value of σ at the node $\boldsymbol{x}_i$, and the piecewise constant interpolations

$$\sigma(t, \boldsymbol{x}_i) \approx \sum_{i=1}^{M} \sigma_i(t)\chi_i(\boldsymbol{x}), \tag{4.91}$$

are introduced, where $\chi_i(\boldsymbol{x})$ is the characteristic function of the finite volume V_i (i. e. $\chi_i(\boldsymbol{x}) = 1$ for $\boldsymbol{x} \in V_i$ and zero elsewhere). Equations (4.91) allows for computing the volume integral in equation (4.89) within the following approximation

$$\int_{V_i} \partial_t(F(\sigma))\mathrm{d}v = \mathrm{d}_t \int_{V_i} F(\sigma)\mathrm{d}v = |V_i|\, \mathrm{d}_t F(\sigma_i), \tag{4.92}$$

where d_t denotes the total derivative with respect to time, and $|V_i|$ is the measure of the finite volume V_i.

In order to compute the surface integral in equation (4.90), the bilinear trial functions $\{N_1 \ldots N_M,\}$ are considered, each of which is required to satisfy the conditions $N_i(\boldsymbol{x}_j) = \delta_{ij}$, and expand η and $\boldsymbol{\kappa}$ as

$$\bar{\eta}(t, \boldsymbol{x}) = \sum_{i=1}^{M} \eta_i(t) N_i(\boldsymbol{x}), \text{ and } \bar{\boldsymbol{\kappa}}(t, \boldsymbol{x}) = \sum_{i=1}^{M} \eta_i(t)\mathrm{grad}(N_i)(\boldsymbol{x}). \tag{4.93}$$

The evaluation of $\bar{\eta}$ and $\bar{\boldsymbol{\kappa}}$ on the boundary ∂V_i of the finite volume V_i is done by selecting a set of integration points, here denoted by I_i. A given element of I_i is labeled by $\boldsymbol{x}_{ij}^{\alpha}$, where the index *j* refers to the *j*-th mesh node sharing an edge with $\boldsymbol{x}_i$, and the in-

dex α enumerates the α-th portion of ∂V_i emanating from the *j*-th edge. This portion of ∂V_i is denoted by Γ_{ij}^{α}. It follows that Γ_{ij}^{α} can be expressed as $\partial V_i = \cup_j \cup_\alpha \Gamma_{ij}^{\alpha}$.

The computation of $\bar{\eta}$ and $\bar{\boldsymbol{\kappa}}$ at a given integration point $\boldsymbol{x}_{ij}^{\alpha}$ leads to the following results:

$$\boldsymbol{\Phi}_{ij}^{\alpha} = \boldsymbol{\Phi}\left(\eta_{ij}^{\alpha}, \boldsymbol{\kappa}_{ij}^{\alpha}\right). \tag{4.94}$$

The quantities η_{ij}^{α} and $\boldsymbol{\kappa}_{ij}^{\alpha}$ are calculated by using the following approximation of the interpolations defined in equation (4.93):

$$\eta_{ij}^{\alpha} \equiv \eta_{ij}^{\alpha}(t) = \bar{\eta}\left(t, \boldsymbol{x}_{ij}^{\alpha}\right) = \eta_i(t) N_i\left(\boldsymbol{x}_{ij}^{\alpha}\right) + \textstyle\sum_k \eta_k(t) N_k\left(\boldsymbol{x}_{ij}^{\alpha}\right), \tag{4.95}$$

$$\kappa_{ij}^{\alpha} \equiv \kappa_{ij}^{\alpha}(t) = \bar{\kappa}\left(t, \boldsymbol{x}_{ij}^{\alpha}\right) = \eta_i(t) \mathrm{grad}(N_i)\left(\boldsymbol{x}_{ij}^{\alpha}\right) + \textstyle\sum_k \eta_k(t) \mathrm{grad}(N_k)\left(\boldsymbol{x}_{ij}^{\alpha}\right), \tag{4.96}$$

where the index *k* ranges through all nodes neighbouring $\boldsymbol{x}_i$, at the element which contains the integration point $\boldsymbol{x}_{ij}^{\alpha}$. By virtue of the definitions introduced so far, the surface integral in equation (4.89) can be written as

$$\int_{\partial V_i} \boldsymbol{\Phi}(\eta, \boldsymbol{\kappa}) \cdot \boldsymbol{n} \, \mathrm{d}a = \textstyle\sum_j \sum_\alpha \left|\Gamma_{ij}^{\alpha}\right| \boldsymbol{\Phi}_{ij}^{\alpha} \cdot \boldsymbol{n}_{ij}^{\alpha}, \tag{4.97}$$

where $\boldsymbol{n}_{ij}^{\alpha}$ is the unit vector normal to the boundary Γ_{ij}^{α}. The application of the results (4.90) and (4.97) to the governing equations (4.72) - (4.74) leads to the following system of non-linear ordinary differential equations, which can be solved by any of the known techniques (an implicit backward Euler scheme was used in the examples):

$$|V_i| \mathrm{d}_t[\phi \varrho(\omega_i, \Theta_i)] + \textstyle\sum_j \sum_\alpha \left|\Gamma_{ij}^{\alpha}\right| \varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right) \boldsymbol{q}_{ij}^{\alpha} \cdot \boldsymbol{n}_{ij}^{\alpha} = 0, \tag{4.98}$$

$$|V_i| \mathrm{d}_t[\phi \varrho(\omega_i, \Theta_i) \omega_i] + \textstyle\sum_j \sum_\alpha \left|\Gamma_{ij}^{\alpha}\right| \left\{\varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right) \omega_{ij}^{\alpha} \boldsymbol{q}_{ij}^{\alpha} + (\boldsymbol{J}_d)_{ij}^{\alpha}\right\} \cdot \boldsymbol{n}_{ij}^{\alpha} = 0, \tag{4.99}$$

$$|V_i| \mathrm{d}_t\left[C\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right) \Theta_i\right] + \textstyle\sum_j \sum_\alpha \left|\Gamma_{ij}^{\alpha}\right| \left\{\varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right) C_f \Theta_{ij}^{\alpha} \boldsymbol{q}_{ij}^{\alpha} + (\boldsymbol{J}_T)_{ij}^{\alpha}\right\} \cdot \boldsymbol{n}_{ij}^{\alpha} = 0, \tag{4.100}$$

where $C = \phi \varrho C_f + (1 - \phi) \varrho_s C_s$ is the effective thermal capacity of the mixture, and the vector quantities $\boldsymbol{q}_{ij}^{\alpha}$, $(\boldsymbol{J}_d)_{ij}^{\alpha}$, and $(\boldsymbol{J}_T)_{ij}^{\alpha}$ are given by

$$\boldsymbol{q}_{ij}^{\alpha} = -\frac{K}{\mu}\left[\mathrm{grad}(p)\left(t, \boldsymbol{x}_{ij}^{\alpha}\right) - \varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right) \boldsymbol{g}\right], \tag{4.101}$$

$$(\boldsymbol{J}_d)_{ij}^{\alpha} = -\varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right) D_m \tau \, \mathrm{grad}(\omega)\left(t, \boldsymbol{x}_{ij}^{\alpha}\right), \tag{4.102}$$

$$(\boldsymbol{J}_T)_{ij}^{\alpha} = -\Lambda_c \mathrm{grad}(\Theta)\left(t, \boldsymbol{x}_{ij}^{\alpha}\right). \tag{4.103}$$

With respect to the finite volume V_i, the products $\varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right)\omega_{ij}^{\alpha}$ and $\varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right)C_f\Theta_{ij}^{\alpha}$ represent the quantity of brine and energy, respectively, that are carried by the flow through the convective term $\boldsymbol{q}_{ij}^{\alpha}$. Since the calculation of these products is influenced by all nodal values of the unknown fields but disregards the direction of the flow, non-physical oscillations of the numerical solution may arise. In order to circumvent this difficulty, some upwind scheme is usually applied, an example being full upwinding using the following conventions:

Tab. 4.4 Initial and boundary values of the variables featuring in the model

Quantity	Subdomain	Symbol	Units	Value
Side of the domain	-	L	m	2500
Side of the initial parcel	-	ℓ	m	500
Initial mass fraction	Subdomain $\mathcal{Q}$	$\omega_{\mathcal{Q}}$	$-$	0.00
Initial temperature	Subdomain $\mathcal{Q}$	$\Theta_{\mathcal{Q}}$	K	473.15
Pressure	$\mathcal{V} = \cup_{k=1}^{4}\{\boldsymbol{r}_k\}$	p_0	Pa	0.00
Initial fluid-phase mass density	Subdomain $\mathcal{Q}$	$\varrho_{\mathcal{Q}}$	$kg \cdot m^{-3}$	875.00
Initial positive buoyancy				
Initial mass fraction	Parcel $\mathcal{P}$	$\omega_{\mathcal{P}}$	$-$	0.47
Initial temperature	Parcel $\mathcal{P}$	$\Theta_{\mathcal{P}}$	K	573.15
Initial fluid-phase mass density	Parcel $\mathcal{P}$	$\varrho_{\mathcal{P}}$	$kg \cdot m^{-3}$	831.00
Initial negative buoyancy				
Initial mass fraction		$\omega_{\mathcal{P}}$	$-$	0.55

Quantity	Subdomain	Symbol	Units	Value
Initial temperature		$\Theta_{\mathcal{P}}$	K	523.15
Initial fluid-phase mass density		$\varrho_{\mathcal{P}}$	$kg \cdot m^{-3}$	919.00

$$\varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right)\omega_{ij}^{\alpha} = \begin{cases} \varrho(\omega_i, \Theta_i)\omega_i, & \text{if } \boldsymbol{q}_{ij}^{\alpha} \cdot \boldsymbol{n}_{ij}^{\alpha} > 0, \\ \varrho\left(\omega_j, \Theta_j\right)\omega_j, & \text{if } \boldsymbol{q}_{ij}^{\alpha} \cdot \boldsymbol{n}_{ij}^{\alpha} < 0, \end{cases} \tag{4.104}$$

$$\varrho\left(\omega_{ij}^{\alpha}, \Theta_{ij}^{\alpha}\right)C_f\Theta_{ij}^{\alpha} = \begin{cases} \varrho(\omega_i, \Theta_i)C_f\Theta_i, & \text{if } \boldsymbol{q}_{ij}^{\alpha} \cdot \boldsymbol{n}_{ij}^{\alpha} > 0, \\ \varrho\left(\omega_j, \Theta_j\right)C_f\Theta_j, & \text{if } \boldsymbol{q}_{ij}^{\alpha} \cdot \boldsymbol{n}_{ij}^{\alpha} < 0. \end{cases} \tag{4.105}$$

Our simulations were performed on an in-house parallel computer on 64 processor cores. Grid resolution was $8^8 \approx 16$ million elements.

4.5.4 Discussion and outlook

In the following discussion, the retardation factor

$$R = \frac{\phi\varrho C_f + (1-\phi)\varrho_s C_s}{\phi\varrho C_f} \tag{4.106}$$

Is used, i. e. the quotient of the effective thermal capacity of the mixture and the thermal convection term /OLD 99/. Physically, it describes the fact that the solid is able to store thermal energy but not brine. With the parameters chosen for these simulations, one gets $R \approx 6.7$, which means that the thermal front moves at approximately 1/7th the velocity of the brine front.

Two cases were considered: one in which the initial parcel is subject to negative buoyancy (cf. Fig. 4.17) and one in which it is subject to positive buoyancy (cf. Fig. 4.18). Each of these two figures shows (from left to right) brine mass fraction, temperature, and density, with the initial values depicted in the first row, the evolution after 10 years in the second row, and the final result after 20 years in the third row. In these pictures, the values of these quantities are shown as a colour plot on the surface of the computation domain with the front cuboid (from the top to bottom, front quarter of the top/bottom face) removed.

In the case of negative buoyancy, brine is transported downward faster than heat because of the retardation factor. The distribution of the brine mass fraction features several “fingers”, which are generated by the interplay of down- and upswelling of fluid motions. Heat, on the other hand, is (as it is delayed) mainly confined to the initial parcel. When the brine starts to leave the initial parcel, fluid density in the parcel becomes smaller and eventually this now low-brine heat parcel begins to show even positive buoyancy. Thus a separation of brine and heat from the initial parcel is observed: brine moving downward and heat slowly upward. Since the formation of brine “fingers” occurs mainly at the sides of the initial parcel, there is a “cylindrical” zone in the middle of the parcel that contains less brine than at the beginning, but has still preserved (almost) its initial temperature. In this zone, a small “fountain”-like motion is observed: brine, which is moved upward by temperature, enters colder regions where it again leads to higher fluid density, and is consequently convected downward.

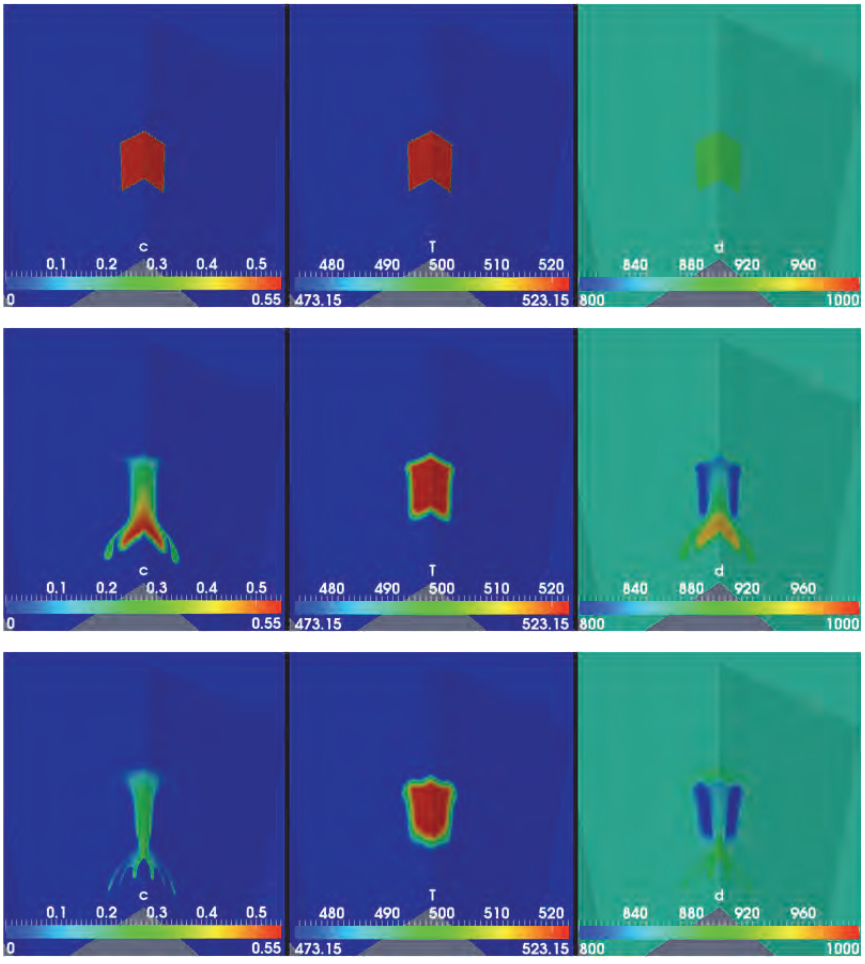


Fig. 4.17 Evolution of the parcel in the case of negative buoyancy:
a) initial configuration; b) parcel after 10 yrs; c) parcel after 20 yrs

In the case of positive buoyancy, again due to the retardation factor, brine is transported upward faster than heat. Above the initial parcel, brine is cooled down, leading to higher density of the fluid. The induced flow moves brine downward along the sides of the parcel. This is comparable to what was described as the “fountain”-like motion above, but is now the main phenomenon. Despite positive buoyancy of the initial parcel, brine leaves this parcel and moves downward in this roundabout way. Since the heat is again confined mostly to the initial parcel because of retardation, fluid density in this zone is decreasing, which even intensifies its positive buoyancy. However, the separation of heat from the initial parcel is hindered by a density “lid”, which is built up by the cold brine (highest density) flowing around the parcel (hot, low-brine, relatively low density).

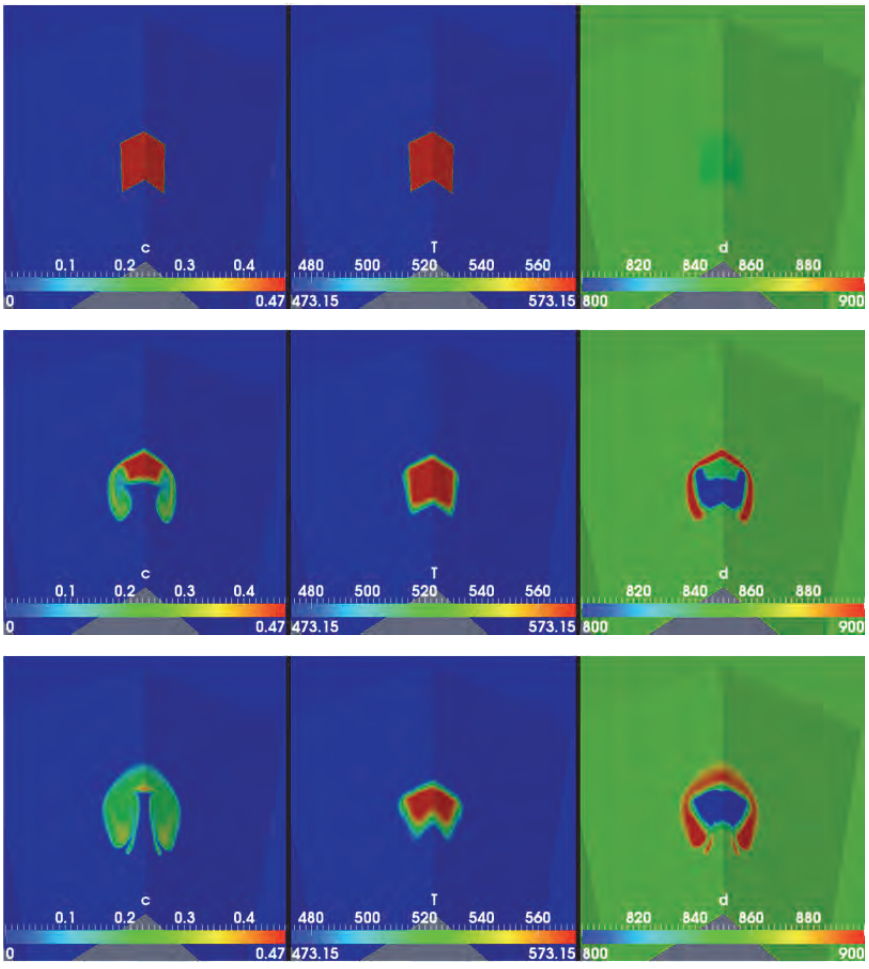


Fig. 4.18 Evolution of the parcel in the case of positive buoyancy:
a) initial configuration; b) parcel after 10 yrs; c) parcel after 20 yrs

It may be noted that the central “fountain”-like motion in the case of negative buoyancy also produces such a density “lid”, although it is less marked.

To highlight the three-dimensional nature of the processes simulated, also isosurfaces of the brine mass fraction for $\omega = 0.15$ are shown. This value seems to be a good indicator for the location of the brine front. Fig. 4.19 shows the case of negative buoyancy at the beginning, after 7, 13, and 20 years (left to right, top to bottom). Fig. 4.20 shows the case of positive buoyancy in the same arrangement.

In /GRI 10b/ it is showed that, in the two-dimensional case, corresponding conditions lead to the formation of several “fingers” in the brine pattern. It is admitted that, in analogy with Elder’s problem, the generation of “fingers” is triggered out by the combination of downswelling and upswelling fluid motions, and if those motions are detected by refining the mesh levels /FRO 01/, then the results of computations do depend on the grid refinement. In forthcoming works a physical interpretation of the formation of “fingers” based on considerations about the stability of the flow will be given, in order to understand whether the number of generated “fingers” tends to increase indefinitely at increasing grid refinement. The question is whether one gets an increasingly precise description of the flow or whether one runs into unphysical results.

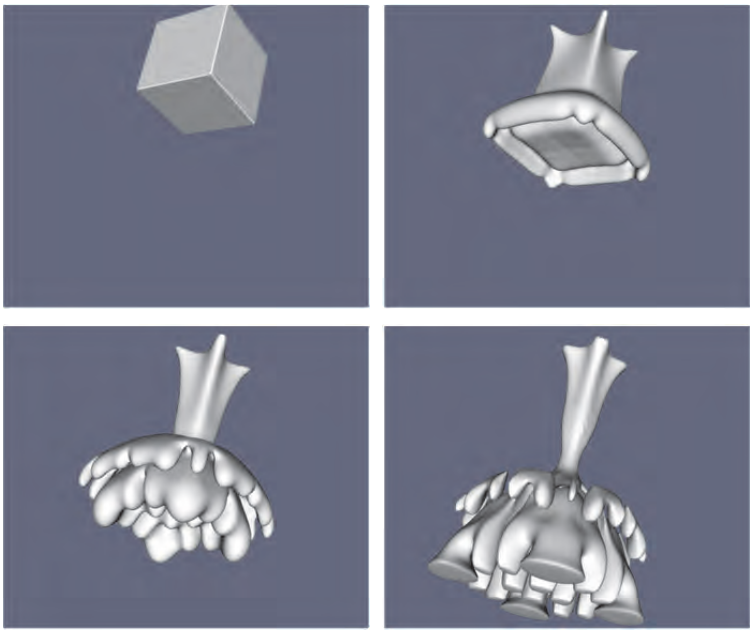


Fig. 4.19 Isosurfaces of brine mass fraction $\boldsymbol{\omega = 0.15}$ in case of negative buoyancy

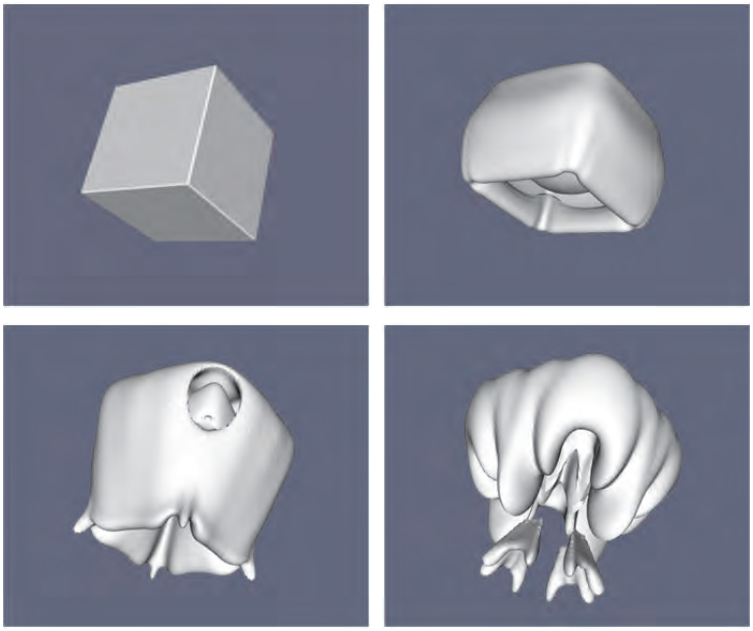


Fig. 4.20 Isosurfaces of brine mass fraction $\boldsymbol{\omega = 0.15}$ in case of positive buoyancy

5 Density-driven flow in fractured media

5.1 Introduction

The study of fractured porous media is an important and challenging problem in hydrogeology. One of the difficulties is that mathematical models have to account for heterogeneity introduced by fractures in hydrogeological media. Heterogeneity may strongly influence the physical processes taking place in these media. The goal of this part of the project was to extend the packages d^3f and r^3t with the possibility to perform the computations with the geometries containing thin fractures filled with porous medium.

The thickness of the fractures, which is usually negligible in comparison with the size of the whole domain, and the complicated geometry of fracture networks reduce essentially the efficiency of numerical methods. In order to overcome these difficulties, fractures are considered as objects of reduced dimensionality (surfaces in three dimensions), and the field equations are averaged along the fracture width (cf., for example, /ANG 09/, BAS 00/, /BAS 99/, /MAR 06/, /SOR 01/). This consideration is also motivated by the geological data that usually do not contain enough geometrical information for the exact representation of the fractures as thin layers.

From a descriptive point of view, a fracture is a portion $\mathcal{F}$ of a region of observation, $\Omega \subset \mathbb{R}^d$ (where $d = 2$ or $d = 3$), characterized by a shape such that one of its geometric dimensions is much smaller than the other ones /BEA 93/. In Ω, there can be either isolated fractures or networks of fractures. Fracture networks usually provide a resistance to the motion of groundwater lower than that of unfractured rock, and may thus lead to the movement of a comparatively large amount of water and the substances dissolved in it. A quantification of flow and transport in fractured rocks was undertaken by Neumann /NEU 05/. Because of its influence on the environment (e. g., pollution of aquifers), and its connection with industrial problems (e. g., modelling of fractured reservoirs 0/), the flow in fractured porous media has received particular attention /MUR 79/, /MAR 06/, /SHI 98/, /GRA 05a/, GRA 09/, /SHA 09/. Mathematical and numerical models have been developed in order to predict fluid flow and contaminant transport in highly heterogeneous domains, the heterogeneity being given by the abrupt change of permeability when passing from the fracture to the surrounding domain, and vice versa.

Together with heterogeneity, the spatial distribution of the fractures in a given network may also influence other flow properties. For example, even though the permeability of the medium embedding the fractures is isotropic, the effective permeability of the equivalent system, made of the network and the embedding medium, may be anisotropic.

The evaluation of the effective flow properties of heterogeneous media is based upon upscaling methods such as, for example, volume-averaging /BRE 62/, /HAS 86/, /WHI 99/, asymptotic expansions /BEN 78/, stochastic modelling, and coarse graining and Renormalization Group Theory /ATT 02/. In the context of volume-averaging, the method known as average-along-the-vertical /BEA 79/ is applied to thin flow regions. The description of flow and transport provided by this method is based on averaging the equations defined in the flow region along its width. Here, the thin flow region is represented by a fracture.

Among the modelling approaches for fractured porous media two approaches may be classified: the near-field and far-field. The first one considers a relatively small domain with a small number of well-defined fractures whose location and shape is known. The second approach is based on the concept of overlapping continua /BEA 93/, which are identified by the fluid in the fractures and the fluid in the embedding medium. If the region of investigation is large with respect to the size of the fractures but not big enough to allow for the introduction of the overlapping continua, then the near field approach is used. The model presented belongs to this case. It is assumed that the fractures are filled by a porous medium whose permeability is bigger than the permeability of the medium enclosing them. The fractures and the embedding medium preserve their identity, and their mutual interaction is studied by means of interface balance laws. Therefore, in the resulting picture, the overall medium is heterogeneous with respect to permeability, and the more permeable medium is inside the fractures.

Here, a fracture has the following two properties: (a) its thickness is much smaller than the smallest characteristic length scale of the embedding medium, and (b) it is filled by a porous medium whose porosity and permeability differ from those of the embedding medium. The fractures are thus distinguishable from the medium. For a given fracture, it is assumed that $K_f \gg K_m$, where K_f and K_m are the permeabilities of the fractures and medium, respectively. The fractures and the embedding medium are considered at the same scale of observation. Since density-driven flow is modeled, it is necessary to describe flow and transport in the whole region of observation. In order to do that, the

same equations are used both in the fracture and in the embedding medium. Flow in the fracture is such that Darcy's law is still applicable. In order to account for the geometric properties of the fracture, Bear's procedure /BEA 79/ is adopted and the equations of density-driven flow are averaged over the fracture width. After averaging, the fractures remain distinguishable from the embedding medium but are regarded as objects of reduced dimensionality.

The equations obtained within the $(d-1)$-dimensional model are similar to the equations determined by Cermelli et al. /CER 05/, in the study of transport relations for surface integrals defined over evolving surfaces. In /CER 05/, the authors elaborate two-dimensional transport models, in which the effect of curvature is accounted for.

Our model is a set of partial differential equations in two different dimensionalities: d and $d-1$. Furthermore, it contains the interface conditions describing the flows between the fractures and the embedding medium, i. e. coupling the PDEs of different dimensionalities. For the numerical solution of this system of equations, a special finite volume discretization is developed and implemented, as well as special methods for the preparation of the computation grid.

Note that the main difficulties arising in the averaging technique in development of the model, as well in the implementation of the numerical methods, relate to those terms in the PDEs that describe the spatial, geometrically non-local phenomena. Thus, the package d^{3}f is more suitable for the demonstration of this technique. In fact, the model lying in its base describes two non-local phenomena: the density-driven movement of the fluid and the transport of the salt. The model in the package r^{3}t contains more equations. But its main geometrically non-local phenomenon is the transport of different species which is from the point of view of the derivation of the model similar to that in d^{3}f. For this, in the present report, the main attention is paid to the description of the extension of d^{3}f.

The model was numerically verified. To this end, simulations with two different representations of a fracture were performed. In the first one, the fractures have the same geometric dimension as the embedding bulk medium and are thus said to be d-dimensional, with $d=2$ in 2D and $d=3$ in 3D. In the second representation, the fractures are considered as $(d-1)$-dimensional manifolds, and the averaged model is used. The first approach is well-established, more general, but computationally more

expensive and practically applicable only for very simple geometries of the domain and the fractures. The second approach, instead, requires some working hypotheses but is computationally essentially cheaper. It could be shown that the results obtained by the two methods are in good agreement with each other for sufficiently small fracture widths.

In the context of the flow simulations in fractured porous media, the following methodological and computational results are presented:

1. The derivation of the equations modelling flow and transport in the fractures that are independent on the equations used in the embedding medium. The unknown functions in the fractures are not assumed to be equal to the unknown functions in the embedding medium.
2. The use of the concept of "excess mass" in the formulation of the $(d-1)$-dimensional representation of fractures.
3. The development of numerical schemes for solving the equations of density-driven flow as formulated in the $(d-1)$-dimensional model.
4. The verification of the proposed $(d-1)$-dimensional model through numerical experiments conducted by using the d-dimensional formulation.

The concept of "excess mass", used in the model of the density-driven flow in the fractures, was used in /HAS 90/ for modelling interfaces, and in the works by Murdoch and Soliman /MUR 99/ and Murdoch /MUR 05/, where it was indicated as one of the most important aspects of modelling transport on lower-dimensional domains embedded in three-dimensional regions. Excess quantities are defined only for extensive quantities, and should be taken into account when the real, three-dimensional interface between two regions is replaced with an equivalent domain of lower dimensionality. This should be done, for example, in order not to "gain" an unphysical mass after ideally "shrinking" the fracture to its mean plane $\mathcal{S}$ (see /MUR 05/ for details).

5.2 Model of the density-driven flow in fractured porous media

The model was developed for the density-driven flow in fractured porous media using Hybrid Mixture Theory /BEN 00/, where a porous medium is macroscopically modelled as a mixture of solids and fluids, that co-exist in a given region of space $\Omega \subset \mathbb{R}^d$ ($d = 2$ or $d = 3$). In many hydrogeological applications, it is assumed that the mixture consists

of a single solid-phase, for example the porous rocky matrix of a soil, and either a multi-phase or a single-phase fluid. The fluid-phase comprises constituents which undergo physico-chemical processes such as advection, diffusion, chemical reactions, and exchange with the solid-phase.

Here, the model is restricted to the case in which a two-constituent fluid experiences single-phase flow through the porous matrix of the solid-phase. The two constituents of the fluid-phase are assumed to be water and brine, the latter being a chemical compound consisting of salts and water. In the presence of exchange processes between the fluid- and the solid-phase, each of these phases should be regarded as a mixture in which each phase is composed of the same constituents /BEN 00/. In the absence of such an assumption it is possible to assume that the fluid- and the solid-phase are a two- and single-constituent system, respectively.

Under the hypothesis that the whole mixture is subject to the saturation condition, the porosity ϕ coincides with the volume fraction of the fluid-phase. The volume fraction of the solid-phase is thus given by $(1-\phi)$.

5.2.1 Governing equations

Here, a fracture $\mathcal{F}$ is a region occupied by a porous medium whose permeability is bigger than the permeability of the medium $\mathcal{M}$ in which it is embedded. It is assumed that the same flow and transport processes occur both in the fracture and in embedding medium. The regions $\mathcal{F}$ and $\mathcal{M}$ interact through exchange processes. To be consistent with the macroscopic continuum description, the partial differential equations governing density-driven flow are obtained by means of the balance laws of mass, momentum, and energy, and the Second Principle of Thermodynamics. These laws have to be written for each constituent of the fluid-phase (i. e. water and brine), and for the solid-phase. However, suitable hypotheses allow for a considerable reduction of the number of equations to be solved. It is assumed that the fracture and the surrounding porous medium satisfy the following requirements:

1. they are subject to a uniform temperature field;
2. pore-scale mass exchange processes between the fluid- and the solid-phase are absent everywhere in Ω;
3. the solid-phase is undeformable and at rest;

4. inertial terms are negligible in the balance laws of momentum; and
5. the porosities of the medium and the fracture are constant but, in general, different from each other.

In order to model the interaction between $\mathcal{F}$ and $\mathcal{M}$, it is necessary to provide a description of the interface separating the fracture from the embedding medium. For simplicity, it is assumed that this interface is ideal. This means that mass and momentum are transferred from the fractures to the medium (and vice versa) without undergoing further processes at the interfaces. The detailed treatment of the interface exchange processes is presented in section 5.2.6.

According to these simplifying hypotheses, the problem of fluid flow and brine transport in a fractured porous medium is macroscopically governed by the laws of mass balance of the brine and the fluid-phase as a whole. These equations must be written for both the porous medium and the fracture. By renaming the mass fraction of the brine by ω, these equations read

$$\partial_t(\phi_\alpha \rho_\alpha) + \nabla \cdot (\rho_\alpha \mathbf{q}_\alpha) = \rho_{\alpha,S} S_\alpha \tag{5.1}$$

$$\partial_t(\phi_\alpha \rho_\alpha \omega_\alpha) + \nabla \cdot (\rho_\alpha \omega_\alpha \mathbf{q}_\alpha + \mathbf{J}_\alpha) = \omega_{\alpha,S} \rho_{\alpha,S} S_\alpha \tag{5.2}$$

The index $\alpha \in \{f, m\}$ specifies whether a physical quantity is defined in the fracture, $\mathcal{F}$, or in the surrounding porous medium, $\mathcal{M} \coloneqq \Omega \setminus \mathcal{F}$. The quantities ϕ_α, ρ_α, and $\mathbf{q}_\alpha$ are the fluid-phase volume fraction (porosity), mass density, and specific discharge, while ω_α and $\mathbf{J}_\alpha$ are the mass fraction and mass flux of the brine, respectively. In the right-hand side, S_α denotes the power of the source/sink, $\omega_{\alpha,S}$ is the mass fraction of the brine in the source (for a sink, $\omega_{\alpha,S} = \omega_\alpha$), $\rho_{\alpha,S} \coloneqq \rho_\alpha(\omega_{\alpha,S})$.

Under the assumption of negligible inertial terms, the momentum balance laws of the brine and the fluid-phase as a whole enable to express $\mathbf{q}_\alpha$ and $\mathbf{J}_\alpha$ in terms of the quantities ϕ_α, ρ_α, ω_α (already present in (5.1 - 5.2)), and the pressure, p_α.

If the validity of Darcy's and Fick's laws is assumed for the problem at hand, then $\mathbf{q}_\alpha$ and $\mathbf{J}_\alpha$ are given by

$$\mathbf{q}_\alpha = -\mu^{-1} \mathbf{K}_\alpha (\nabla p_\alpha - \rho_\alpha \mathbf{g}), \tag{5.3}$$

$$\mathbf{J}_\alpha = -\rho_\alpha \mathbf{D}_\alpha \nabla \omega_\alpha, \tag{5.4}$$

where $\mathbf{K}_\alpha$, μ, $\mathbf{g}$, and $\mathbf{D}_\alpha$ are the permeability tensor, fluid viscosity, gravity acceleration vector, and diffusion-dispersion tensor, respectively. Equations (5.3) - (5.4) can be obtained through the exploitation of the dissipation inequality for both $\mathcal{F}$ and $\mathcal{M}$ /BEA 93/, /BEN 00/, /GRO 54/.

Requiring the validity of Darcy's law in the fracture may be a strong assumption. Indeed, since the fluid is expected to flow faster in the fracture than in the surrounding medium, the Forchheimer's correction term /DIE 05/ may become necessary for a more precise description of the flow in the fracture. The assumption of Fick-type diffusion may be questionable too, when big values of brine mass fraction are involved. Here, the choices of modelling are motivated by simplicity only.

In order to close the system (5.3) - (5.4), it further is assumed that the mass density of the fluid-phase, ρ_α, is a given constitutive function of the brine mass fraction ω_α, i. e. $\rho_\alpha \coloneqq \rho_\alpha(\omega_\alpha)$. In particular, following Oldenburg and Pruess /OLD 98/, ρ_α is prescribed by

$$\frac{1}{\rho_\alpha(\omega_\alpha)} \coloneqq \frac{1-\omega_\alpha}{\rho^{pW}} + \frac{\omega_\alpha}{\rho^{pB}}, \tag{5.5}$$

where the mass densities of “pure water”, ρ^{pW}, and “pure brine”, ρ^{pB}, are given constants. Note that (5.5) holds true under the hypothesis of additivity of volumes, which means that the volume of a sample of mixture, made by water and the brine, equals the sum of the volume of water and the volume of brine present in the sample. Since the mass density $\rho_\alpha(\omega_\alpha)$ varies in response to the mass fraction, the resulting non-potential flow /BEA 72/ is said to be density-driven. The pressure, p_α, is therefore the Lagrange multiplier associated to the constraint $\rho_\alpha = \rho_\alpha(\omega_\alpha)$. More details about the derivation of the constitutive relation (5.5) are given in /OLD 98/. It should only be remark that (5.5) is consistent with the definition given in Mixture Theory $\rho \coloneqq \rho^W + \rho^B$ (see, for example, Hassanizadeh /HAS 86/ and Bennethum et al. /BEN 00/), where ρ^W and ρ^B are the apparent densities of water and brine in the fluid-phase. These quantities are sometimes referred to as “concentrations”. The intrinsic, or “pure”, densities of water and brine are related to ρ^W and ρ^B through $\rho^J = \varphi^J \rho^{pJ}$, with $J \in \{B, W\}$. The composition of the fluid-phase is measured by the mass fractions of its constituents. By definition $\omega^B \equiv \omega = \rho^B/\rho$, and $\omega^W = 1 - \omega$.

5.2.2 A change of variables

Our purpose is to average (5.1) - (5.4) in order to study density-driven flow in a thin fracture. However, the form of these equations makes the averaging procedure very cumbersome because of the many products of fluctuations and the non-linearity of (5.5).

In order to overcome this problem, a change of variables is proposed which reduces the number of fluctuations featuring in the averaged form of (5.1) - (5.4). In these equations, the set of free unknowns, given by mass fraction and pressure, is $\mathcal{U} \coloneqq \{\omega_\alpha, p_\alpha\}_{\alpha=f,m}$.

Our approach consists of replacing the brine mass fraction, ω_α, with the apparent mass density of the brine, ρ_α^B (hereafter called concentration). For ease of notation, the latter quantity will be denoted by c_α from here on (i. e. $c_\alpha \equiv \rho_\alpha^B$ $[\mathrm{kg \cdot m^{-3}}]$), so that the transformed set of unknowns reads $\mathcal{U}^* \coloneqq \{c_\alpha, p_\alpha\}_{\alpha=f,m}$.

By using the definitions of brine concentration, c_α, and mass fraction, $\omega_\alpha \coloneqq c_\alpha/\rho_\alpha$, it can be shown that (5.5) can be equivalently reformulated as

$$\rho_\alpha(c_\alpha) \coloneqq \rho^{pW} + \frac{\rho^{pB}-\rho^{pW}}{\rho^{pB}} c_\alpha. \tag{5.6}$$

When c_α attains its maximum possible value (i. e. $c_\alpha = \rho_\alpha^B$), the fluid-phase mass density equals the mass density of “pure brine”. The constitutive law (5.6) was used in the paper by Henry /HEN 64a/. Other relations can be found, for example, in Holzbecher /HOL 98/.

Equation (5.6) was obtained without linearizing any constitutive law of the form $\rho_\alpha = \rho_\alpha(c_\alpha)$. Furthermore, since ρ^{pW} and ρ^{pB} are fixed parameters here, (5.6) does not introduce any new coefficient to be determined experimentally. The change of variables, however, is useful as long as thermal phenomena are neglected. Indeed, if the thermal expansion were taken into account, the concentration would no longer be an appropriate free unknown because it would depend on temperature.

A constitutive law similar to (5.6) was given in /BEA 72/ as a hint for solving problems analogous to ours. In /BEA 72/, however, the proposed law is $\rho(c) = a + b\,c$, where a and b are coefficients to be determined experimentally.

With respect to the variables $\mathcal{U}^* \coloneqq \{c_\alpha, p_\alpha\}_{\alpha=f,m}$, and after denoting

$$\rho' \coloneqq \frac{\rho^{pB}-\rho^{pW}}{\rho^{pB}}, \tag{5.7}$$

the quantities $\mathbf{q}_\alpha$ and $\mathbf{J}_\alpha$ in (5.3) and (5.4) transform into

$$\mathbf{q}_\alpha = -\mu^{-1}\mathbf{K}_\alpha(\nabla p_\alpha - \rho_\alpha(c_\alpha)\mathbf{g}), \tag{5.8}$$

$$\mathbf{J}_\alpha = -\left(\frac{\rho^{pW}}{\rho^{pW}+\rho' c_\alpha}\mathbf{D}_\alpha\right)\nabla c_\alpha, \tag{5.9}$$

Finally, the governing equations take on the form

$$\nabla\cdot(\rho^{pW}\mathbf{q}_\alpha - \rho'\mathbf{J}_\alpha) = \rho^{pW}S_\alpha, \tag{5.10}$$

$$\phi_\alpha\partial_t c_\alpha + \nabla\cdot(c_\alpha\mathbf{q}_\alpha + \mathbf{J}_\alpha) = c_{\alpha,S}S_\alpha, \tag{5.11}$$

with $\alpha \in \{f, m\}$. Equation (5.10) is obtained under the assumption of constant porosity by substituting (5.6) into the RHS of (5.1), multiplying (5.2) by ρ', and substracting the resulting expression from (5.1). In the right-hand side of (5.11), $c_{\alpha,S}$ denotes the inflow concentration of the brine in the source. For a sink, $c_{\alpha,S} = c_\alpha$.

When a fracture is considered as a d-dimensional object, the system (5.10) - (5.11) and quantities (5.6) - (5.9) have the same form both in the fracture, $\mathcal{F}$, and in the surrounding porous medium, $\mathcal{M} = \Omega \setminus \mathcal{F}$. When the fracture is treated as an equivalent $(d-1)$-dimensional object, (5.9) - (5.11) will be averaged.

It should be remarked that the Boussinesq-Oberbeck approximation is retrieved by setting $\rho' = 0$ in (5.10).

From now on, the fluid viscosity, μ, is assumed to be independent on concentration, and the permeability tensor, $\mathbf{K}_\alpha$, is assumed to be constant and isotropic, i. e. $\mathbf{K}_\alpha = K_\alpha\mathbf{I}$, both in the fracture and the surrounding porous medium. Since the dependency of fluid viscosity on the brine concentration is not negligible in general, the hypothesis of constant fluid viscosity is employed here just to simplify calculations. On the other hand, the hypothesis of isotropic permeability can be physically motivated by assuming that the internal structure of the porous media filling the fractures and the embedding

medium does not induce preferential flow directions. An example of a porous medium with anisotropic permeability is given by the presence of sediments which, when deposited, lead to a higher permeability in one direction /BEA 79/.

In general, the tensor $\mathbf{D}_\alpha$ $[\frac{\mathrm{m}^2}{\mathrm{s}}]$ describes diffusion and mechanical dispersion, i. e.

$$\mathbf{D}_\alpha \coloneqq \mathbf{D}_\alpha^d + \mathbf{D}_\alpha^{md}. \tag{5.12}$$

The diffusion tensor, $\mathbf{D}_\alpha^d$, accounts for tortuosity, and is therefore defined by $\mathbf{D}_\alpha^d \coloneqq D^d\mathbf{T}_\alpha$, where D^d is the scalar molecular diffusivity, and $\mathbf{T}_\alpha$ is the tortuosity tensor. In the following, only the case of isotropic tortuosity (i. e. $\mathbf{T}_\alpha = T_\alpha\mathbf{I}$) is considered. Furthermore, by postulating isotropic dispersivity, the tensor of mechanical dispersion, $\mathbf{D}_\alpha^{md}$, is transversely isotropic, and admits the expression given by Scheidegger /SCH 74/, i. e.

$$\mathbf{D}_\alpha^{md} \coloneqq a_\alpha^t|\mathbf{q}_\alpha|\mathbf{I} + (a_\alpha^\ell - a_\alpha^t)\frac{\mathbf{q}_\alpha\otimes\mathbf{q}_\alpha}{|\mathbf{q}_\alpha|}, \tag{5.13}$$

where a_α^t and a_α^ℓ are the transversal and longitudinal dispersivity lengths, respectively, and the symmetry axis which generates the transverse isotropy is given by the direction of flow, $\mathbf{q}_\alpha$, or, equivalently, by the second-order symmetric tensor $\mathbf{q}_\alpha\otimes\mathbf{q}_\alpha$ /BEA 90/.

Because of the different properties of the subregions $\mathcal{F}$ and $\mathcal{M}$, the permeability, K_α, tortuosity, T_α, and the dispersivities a_α^t and a_α^ℓ feature the index $\alpha \in \{f, m\}$. With respect to the global region of observation Ω, these physical quantities are regarded as piecewise constant.

5.2.3 Geometric model of a single fracture

For modelling purposes, a fracture $\mathcal{F}$ is considered as a shell-shaped d-dimensional domain, in which one of the d geometric dimensions, the width, is much smaller than the other two. Since the fracture $\mathcal{F}$ is embedded in Ω and surrounded by $\mathcal{M} = \Omega \setminus \mathcal{F}$, the boundary of the subregion $\mathcal{F}$ coincides with the inner boundary of $\mathcal{M}$. This common boundary is referred to as $\partial\mathcal{F}$. Because of the shell-type geometry, the fracture is delimited by the two surfaces, $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$, and possesses a band-shaped lateral boundary, denoted by $\mathcal{B}$. Accordingly, the boundary of the fracture is $\partial\mathcal{F} = \mathcal{B} \cup \mathcal{S}^{(1)} \cup \mathcal{S}^{(2)}$.

In this derivation, the case is regarded in which the surfaces $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ are parallel planes. Moreover, by denoting by $\{O,(X,Y,Z)\}$ and $\{o,(x,y,z)\}$ a global and a local right-handed coordinate frame, respectively, $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ are required to have the same projection onto the (x,y)-plane, and to be described by the following two equations

$$\mathcal{S}^{(1)}\colon \sigma^{(1)}(\mathbf{r}) = z - z^{(1)} = 0, \quad \mathcal{S}^{(2)}\colon \sigma^{(2)}(\mathbf{r}) = z - z^{(2)} = 0, \tag{5.14}$$

see Fig. 5.1. The width of the fracture is defined by the distance between $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$:

$$\epsilon \coloneqq z^{(2)} - z^{(1)} > 0. \tag{5.15}$$

The fracture mean plane is denoted by $\mathcal{S}$, and is assumed to be identified by the equation $z = 0$ (this implies: $z^{(1)} = \epsilon/2$, and $z^{(2)} = -\epsilon/2$).

For consistency with the upscaling procedure adopted in Hybrid Mixture Theory, the distance ϵ has to satisfy the inequalities $\ell \ll \epsilon \ll L$, with ℓ and L being the pore-scale and macro-scale characteristic lengths in Ω, respectively.

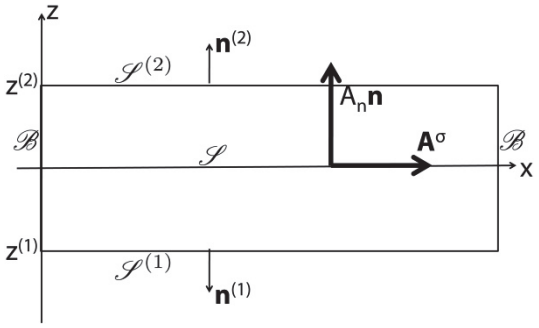


Fig. 5.1 Scheme of a planar *d*-dimensional fracture in the averaging process

5.2.4 Average along the thickness

The theory shown can be extended to more complicated fracture geometries. Furthermore, although the average is treated here along the thickness in the case of a single fracture, the computations done here are also applied to fractures with piecewise flat delimiting surfaces, and fracture networks. Example of intersections between two fractures, which may be of interest for investigating fracture networks, are presented in Fig.

5.6 and Fig. 5.7 below. An approach similar to ours has been recently adopted by Angot et al. /ANG 09/.

It is averaged (5.8) - (5.11), with $\alpha = f$, for a fracture having the shape described in section 5.2.3 (see Fig. 5.4).

The average is performed along the thickness of the fracture according to the averaging method introduced in /BEA 72/, /BEA 77/, /BEA 79/, /BEA 90/. For a given field F (either a scalar, a vector, or a tensor field), this averaging technique is based on the definition of the operator

$$\langle F\rangle(t,x,y) \coloneqq \frac{1}{\epsilon}\int_{-\epsilon/2}^{\epsilon/2} F(t,x,y,z)dz. \tag{5.16}$$

The following notation is used for the averaged values of the concentration and pressure in the fracture:

$$\bar{c}_f \coloneqq \langle c_f\rangle, \text{ and } \bar{p}_f \coloneqq \langle p_f\rangle. \tag{5.17}$$

It should be remarked that the definition (5.16) is less general than the one used in /BEA 77/.

If F and $\mathbf{A}$ are a scalar and a vector field, respectively, then one obtains

$$\langle F\mathbf{A}\rangle = \langle F\rangle\langle\mathbf{A}\rangle + \langle\tilde{F}\widetilde{\mathbf{A}}\rangle, \tag{5.18}$$

where $\tilde{F} \coloneqq F - \langle F\rangle$ and $\widetilde{\mathbf{A}} \coloneqq \mathbf{A} - \langle\mathbf{A}\rangle$ are the fluctuations of the fields F and $\mathbf{A}$. The following results hold also true:

$$\langle\partial_t F\rangle(t,x,y) = \partial_t\langle F\rangle(t,x,y), \tag{5.19}$$

$$\langle\nabla\cdot\mathbf{A}\rangle = \nabla^\sigma\cdot\langle\mathbf{A}^\sigma\rangle + \frac{\mathbf{A}^{(2)}\cdot\mathbf{n}^{(2)} - \mathbf{A}^{(1)}\cdot\mathbf{n}^{(1)}}{\epsilon}. \tag{5.20}$$

In (5.20), $\mathbf{A}^\sigma = \mathbb{P}.\mathbf{A} = \sum_{k=x,y} A_k\mathbf{e}_k$ is the projection of $\mathbf{A} = \sum_{j=x,y,z} A_j\mathbf{e}_j$ onto the (x,y)-plane, which is spanned by the unit vectors $\mathbf{e}_x$ and $\mathbf{e}_y$ of the $\mathbb{R}^3$-canonical basis $\{\mathbf{e}_x,\mathbf{e}_y,\mathbf{e}_z\}$, $\nabla^\sigma = \mathbb{P}.\nabla$ is the surface divergence operator defined on this plane, and $\mathbb{P} \coloneqq (\mathbf{I} - \mathbf{e}_z\otimes\mathbf{e}_z)$ is the projection operator. The unit vectors $\mathbf{n}^{(1)}$ and $\mathbf{n}^{(2)}$ are antipar-

allel, normal to the surfaces $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$, respectively, and are directed in both cases from the fracture, $\mathcal{F}$, into the surrounding medium, $\mathcal{M}$. The quantities $\mathbf{A}^{(1)}$ and $\mathbf{A}^{(2)}$ are the restrictions of the vector field $\mathbf{A}$ onto the surfaces $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$, respectively.

Equation (5.20) becomes

$$\langle \nabla \cdot \mathbf{A} \rangle = \nabla^{\sigma} \cdot \langle \mathbf{A}^{\sigma} \rangle + \frac{A_n^{(2)} - A_n^{(1)}}{\epsilon}, \tag{5.21}$$

with $A_n^{(2)} := \mathbf{A}^{(2)} \cdot \mathbf{n}^{(2)}$, $A_n^{(1)} := \mathbf{A}^{(1)} \cdot \mathbf{n}^{(1)}$. In the following, the local reference frame $\{o, (x, y, z)\}$ is chosen such that $\mathbf{e}_z = \mathbf{n} = \mathbf{n}^{(2)} = -\mathbf{n}^{(1)}$, and

$$\mathbf{A}^{(1)} \coloneqq \mathbf{A}\left(t, x, y, -\frac{\epsilon}{2}\right), \quad \mathbf{A}^{(2)} \coloneqq \mathbf{A}\left(t, x, y, \frac{\epsilon}{2}\right). \tag{5.22}$$

When the parameter ϵ is very small, the second term on the right-hand side of (5.21) can be regarded as the approximation of the normal derivative of A_n.

5.2.5 Averaged equations – Further simplifying assumptions

The constitutive law (5.6) defines the fluid-phase mass density as an affine function of the brine concentration. Since the mass densities of “pure water” and “pure brine”, ρ^{pW} and ρ^{pB}, are given constants in the formulation used here, the average of the function $\rho_f(c_f)$ reads

$$\langle \rho_f(c_f) \rangle = \rho_f\big(\langle c_f \rangle\big) \equiv \rho_f\big(\bar{c}_f\big) = \rho^{pW} + \rho' \bar{c}_f. \tag{5.23}$$

For ease of notation, two “auxiliary” vector fields are introduced:

$$\mathbf{Q}_\alpha \coloneqq \rho^{pW} \mathbf{q}_\alpha - \rho' \mathbf{J}_\alpha, \quad \mathbf{P}_\alpha \coloneqq c_\alpha \mathbf{q}_\alpha + \mathbf{J}_\alpha. \tag{5.24}$$

By virtue of (5.24), the governing equations (5.10) (5.11) become

$$\nabla \cdot \mathbf{Q}_\alpha = \rho^{pW} S_\alpha, \tag{5.25}$$

$$\phi_\alpha \partial_t c_\alpha + \nabla \cdot \mathbf{P}_\alpha = c_{\alpha,S} S_\alpha. \tag{5.26}$$

By applying the average operator (5.16) to (5.10) - (5.11) (or, equivalently, to (5.25) - (5.26)), and using (5.18) - (5.20), the averaged form of the governing equations is:

$$\nabla^\sigma \cdot \left(\rho^{pW}\langle \mathbf{q}_f^\sigma\rangle - \rho'\langle \mathbf{J}_f^\sigma\rangle\right) + \frac{Q_{fn}^{(2)}+Q_{fn}^{(1)}}{\epsilon} = \rho^{pW}\langle S_f\rangle, \tag{5.27}$$

$$\phi_f \partial_t\langle c_f\rangle + \nabla^\sigma \cdot \left(\langle c_f\rangle\langle \mathbf{q}_f^\sigma\rangle + \langle \widetilde{c_f}\,\widetilde{\mathbf{q}_f^\sigma}\rangle + \langle \mathbf{J}_f^\sigma\rangle\right) + \frac{P_{fn}^{(2)}+P_{fn}^{(1)}}{\epsilon} = \langle c_{f,s} S_f\rangle. \tag{5.28}$$

The averaged vector fields in (5.27) - (5.28) may be written as

$$\langle \mathbf{q}_f^\sigma\rangle = -\mu^{-1}K_f\left[\nabla^\sigma \bar{p}_f - \left(\rho^{pW} + \rho'\bar{c}_f\right)\mathbf{g}^\sigma\right], \tag{5.29}$$

$$\langle \widetilde{c_f}\,\widetilde{\mathbf{q}_f^\sigma}\rangle = -\langle \widetilde{c_f}\mu^{-1}K_f\left[\nabla^\sigma \widetilde{p_f} - \rho'\widetilde{c_f}\mathbf{g}^\sigma\right]\rangle, \tag{5.30}$$

$$\langle \mathbf{J}_f^\sigma\rangle = -\langle \mathbb{P}\left[\frac{\rho^{pW}}{\rho^{pW}+\rho' c_f}\mathbf{D}_f \nabla c_f\right]\rangle, \tag{5.31}$$

where the projection operator $\mathbb{P}$ has been applied to the vectors $\mathbf{q}_f$ and $\mathbf{J}_f$ in order to obtain $\mathbf{q}_f^\sigma$ and $\mathbf{J}_f^\sigma$, respectively.

The use of (5.31) renders the equations to be solved very difficult. In order to overcome this difficulty, the dispersion part of the diffusion-dispersion tensor from the original model is neglected here. Moreover, by assuming that D^d and T_f are given constants, and linearizing the ratio $\rho^{pW}/(\rho^{pW}+\rho' c_f)$, the tangential component of the diffusive mass flux is rewritten as

$$\mathbf{J}_f^\sigma = -D_f\left(1 - \frac{\rho'}{\rho^{pW}}c_f\right)\nabla^\sigma c_f, \tag{5.32}$$

where $D_f \coloneqq D^d T_f$. In order to compute the average of the diffusive mass flux (5.32), it is further assumed that the term $\langle \widetilde{c_f}\,\widetilde{\nabla^\sigma c_f}\rangle$ is negligible, and one gets the approximated averaged expression

$$\langle \mathbf{J}_f^\sigma\rangle = -D_f\left(1 - \frac{\rho'}{\rho^{pW}}\bar{c}_f\right)\nabla^\sigma \bar{c}_f. \tag{5.33}$$

The last quantity to be examined is the advective term $\langle \widetilde{c_f}\,\widetilde{\mathbf{q}_f^\sigma}\rangle$, which gives rise to mechanical macrodispersion (cf. /BEA 79/, /BEA 90/ for details). In the case of flow of a

single fluid phase in an aquifer, Bear and Bachmat /BEA 90/ assume that the macro-dispersion associated with the total mass flux can be neglected, and thus write

$$\langle \rho_f \mathbf{q}_f^\sigma \rangle = \langle \rho_f \rangle \langle \mathbf{q}_f^\sigma \rangle + \langle \widetilde{\rho_f} \widetilde{\mathbf{q}_f^\sigma} \rangle \approx \langle \rho_f \rangle \langle \mathbf{q}_f^\sigma \rangle, \tag{5.34}$$

this meaning that $\langle \widetilde{\rho_f} \widetilde{\mathbf{q}_f^\sigma} \rangle$ is small in comparison with $\langle \rho_f \rangle \langle \mathbf{q}_f^\sigma \rangle$. However, by using the constitutive expression (5.6), and substituting it into (5.34), it should be noticed that (5.34) implies

$$\langle c_f \mathbf{q}_f^\sigma \rangle = \langle c_f \rangle \langle \mathbf{q}_f^\sigma \rangle + \langle \widetilde{c_f} \widetilde{\mathbf{q}_f^\sigma} \rangle \approx \langle c_f \rangle \langle \mathbf{q}_f^\sigma \rangle. \tag{5.35}$$

The mathematical assumption of neglecting fluctuations amounts to neglect dispersion in the averaged form of the brine mass balance law. Note however that the neglected terms in (5.35) model mainly the part of the dispersion in the normal direction to the fracture. The tangential dispersion in (5.31) was neglected for the sake of the mathematically rigorous derivation. To compensate this simplification, the dispersion, neglected in (5.35), is included into the definition of $\langle \mathbf{J}_f^\sigma \rangle$, replacing (5.33) by

$$\langle \mathbf{J}_f^\sigma \rangle = -\widehat{\mathbf{D}}_f \left(1 - \frac{\rho'}{\rho^{pW}} \bar{c}_f\right) \nabla^\sigma \bar{c}_f, \tag{5.36}$$

where $\widehat{\mathbf{D}}_f \coloneqq D_f \mathbf{I} + \widehat{\mathbf{D}}_f^{md}$,

$$\widehat{\mathbf{D}}_f^{md} \coloneqq a_f^t \left|\mathbf{q}_f^\sigma\right| \mathbf{I} + (a_f^\ell - a_f^t) \frac{\mathbf{q}_f^\sigma \otimes \mathbf{q}_f^\sigma}{|\mathbf{q}_f^\sigma|}. \tag{5.37}$$

Note that $\widehat{\mathbf{D}}_f^{md}$ is a $(d-1) \times (d-1)$-tensor.

The averaged source/sink term in (5.28) has to be approximated, too. For this purpose is set

$$\langle c_{f,S} S_f \rangle \approx \langle c_{f,S} \rangle \langle S_f \rangle. \tag{5.38}$$

Note that for the sources, $c_{f,S}$ and S_f are known, so that (5.38) can be considered as a redefinition of $\bar{c}_{f,S} \coloneqq \langle c_{f,S} \rangle$. For sinks, (5.38) becomes an exact equality if S_f is constant along the fracture width.

The simplified form of the averaged governing equations in the fracture is thus given by

$$\nabla^{\sigma} \cdot \left(\rho^{pW} \epsilon \langle \mathbf{q}_f^{\sigma} \rangle - \rho' \epsilon \langle \mathbf{J}_f^{\sigma} \rangle\right) + \left(\hat{Q}_{fn}^{(2)} + \hat{Q}_{fn}^{(1)}\right) = \epsilon \rho^{pW} \bar{S}_f, \tag{5.39}$$

$$\phi_f \epsilon \partial_t \bar{c}_f + \nabla^{\sigma} \cdot \left(\epsilon \bar{c}_f \langle \mathbf{q}_f^{\sigma} \rangle + \epsilon \langle \mathbf{J}_f^{\sigma} \rangle\right) + \left(\hat{P}_{fn}^{(2)} + \hat{P}_{fn}^{(1)}\right) = \epsilon \bar{c}_{f,S} \bar{S}_f, \tag{5.40}$$

where $\bar{S}_f \coloneqq \langle S_f \rangle$ does not depend on the unknown functions and is considered to be given, $\langle \mathbf{q}_f^{\sigma} \rangle$ and $\langle \mathbf{J}_f^{\sigma} \rangle$ are defined in (5.29) and (5.36) - (5.37), respectively, whereas $\hat{Q}_{fn}^{(k)}$ and $\hat{P}_{fn}^{(k)}$ (with $k = 1{,}2$) are approximations of the fluxes $Q_{fn}^{(k)}$ and $P_{fn}^{(k)}$ in (5.27) - (5.28). The quantities $\hat{Q}_{fn}^{(k)}$ and $\hat{P}_{fn}^{(k)}$ are defined in section 5.2.6, where interface balance laws are discussed. Equations (5.39) - (5.40) amount to say that averaged quantities are considered constant throughout the fracture thickness. This property follows from the definition of the averaging operator (5.16), and is the goal of the procedure. Within this approach, a given fracture is regarded as a “shell”, i. e. an object whose balance laws are written with respect to the tangent space to its mean surface.

(5.39) - (5.40) describe density-driven flow and brine diffusion by means of the averaged quantities $\langle \mathbf{q}_f^{\sigma} \rangle$ and $\langle \mathbf{J}_f^{\sigma} \rangle$, which are defined on the mean plane of the fracture, $\mathcal{S}$, that means they provide an $(d-1)$-dimensional representation of the phenomena taking place in the fracture. Equations (5.39) - (5.40) in the unknowns $\bar{c}_f$ and $\bar{p}_f$ are thus defined on $\mathcal{S}$ (not in $\mathcal{F}$), and have to be coupled to the set of equations for the concentration and the pressure defined in $\widehat{\mathcal{M}} \coloneqq \Omega \setminus \mathcal{S}$, not in $\mathcal{M}$. These new unknowns are denoted by $\hat{c}_m \colon \widehat{\mathcal{M}} \to \mathbb{R}$ and $\hat{p}_m \colon \widehat{\mathcal{M}} \to \mathbb{R}$. They approximate the original unknown functions $c_m \colon \mathcal{M} \to \mathbb{R}$ and $p_m \colon \mathcal{M} \to \mathbb{R}$ defined in the smaller domain $\mathcal{M} \subset \widehat{\mathcal{M}}$. For $\hat{c}_m$ and $\hat{p}_m$, equations to (5.10) - (5.11) are formulated analogously:

$$\nabla \cdot \left(\rho^{pW} \hat{\mathbf{q}}_m - \rho' \hat{\mathbf{J}}_m\right) = \rho^{pW} S_m, \tag{5.41}$$

$$\phi_m \partial_t \hat{c}_m + \nabla \cdot \left(\hat{c}_m \hat{\mathbf{q}}_m + \hat{\mathbf{J}}_m\right) = \hat{c}_{m,S} S_m, \tag{5.42}$$

where

$$\hat{\mathbf{q}}_m = -\mu^{-1} K_m \left(\nabla \hat{p}_m - (\rho^{pW} + \rho' \hat{c}_m)\mathbf{g}\right), \tag{5.43}$$

$$\hat{\mathbf{J}}_m = -\mathbf{D}_m \left(\frac{\rho^{pW}}{\rho^{pW} + \rho' \hat{c}_m}\right) \nabla \hat{c}_m. \tag{5.44}$$

Equations (5.41) - (5.44) are defined in $\widehat{\mathcal{M}}$ (not only in $\mathcal{M}$). Boundary conditions at the fracture-medium interface have to be prescribed.

Equations (5.39) - (5.40), obtained by adopting Bear's method 0, are similar to those determined by Hassanizadeh and Gray /HAS 89a/, which are based on the averaging procedure put forward by Gray /GRA 82/, /GRA 83/. In /HAS 89a/, the authors modelled transport across zones with “reduced dynamics”, and formulated conditions linking the dynamic processes among different flow regions. Their resulting equations represented vertically averaged balance laws of physical quantities defined in a **nonsimple** interface (e. g., a fracture) between two domains. The constitutive prescriptions as well as Darcy and Fick's laws were assigned to the macroscopic quantities featuring in the averaged equations.

5.2.6 Balance laws at the fracture-medium interface

The conditions at the fracture-medium interface express the continuity of mass and momentum for both the brine and the overall fluid-phase across the fracture boundary $\partial\mathcal{F} = \mathcal{B} \cup \mathcal{S}^{(1)} \cup \mathcal{S}^{(2)}$. The boundary $\partial\mathcal{F}$ is modelled as a simple interface, i. e. an interface that satisfies the following requirements:

1. it is a narrow zone whose thickness is of the order of the REV length scale;
2. it does not constitute an actual barrier between the two adjacent media;
3. the two adjacent media are in direct thermodynamic contact.

The unit vector normal to $\mathcal{B}$ and tangent to the (x, y)-plane will be denoted by $\boldsymbol{\tau}$.

Firstly, the case of the full-dimensional (i. e. d-dimensiona) fracture is considered. The mass balance at the fracture-medium interfaces is given by the continuity of the normal components of the mass fluxes of both the fluid-phase as a whole and the brine. When referred to $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$, the explicit form of these conditions reads

$$-\rho(c_f)\frac{K_f}{\mu}\left(\partial_n p_f - \rho(c_f) g_n\right) = -\rho(c_m)\frac{K_m}{\mu}\left(\partial_n p_m - \rho(c_m) g_n\right), \tag{5.45}$$

$$-D_f\left(\frac{\rho^{pW}}{\rho^{pW}+\rho' c_f}\right)\partial_n c_f = -D_m\left(\frac{\rho^{pW}}{\rho^{pW}+\rho' c_m}\right)\partial_n c_m. \tag{5.46}$$

For the band-shaped lateral boundary, $\mathcal{B}$, one obtains

$$-\rho(c_f)\frac{K_f}{\mu}\left(\partial_\tau p_f - \rho(c_f) g_\tau\right) = -\rho(c_m)\frac{K_m}{\mu}\left(\partial_\tau p_m - \rho(c_m) g_\tau\right), \tag{5.47}$$

$$-D_f\left(\frac{\rho^{pW}}{\rho^{pW}+\rho' c_f}\right)\partial_\tau c_f = -D_m\left(\frac{\rho^{pW}}{\rho^{pW}+\rho' c_m}\right)\partial_\tau c_m. \tag{5.48}$$

The balance of momentum, under the hypotheses of macroscopically inviscid fluid and negligible advective contributions, implies that both p_α and c_α are continuous across $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ /HAS 89/, i. e.

$$p_f = p_m, \quad \text{and} \quad c_f = c_m, \quad \text{on } \mathcal{S}^{(1)} \text{ and } \mathcal{S}^{(2)}. \tag{5.49}$$

The balance laws (5.45) - (5.49) for modelling the interaction between the fracture, $\mathcal{F}$, and the surrounding porous medium, $\mathcal{M}$, hold when $\mathcal{F}$ is regarded as a d-dimensional object. Since the concentration of the brine is continuous across the interfaces, and the fluid-phase mass density is continuous too, the discontinuity of the normal derivatives of pressure and concentration is due to the abrupt changes of permeability and diffusivity when passing from the fracture to the embedding medium, and vice versa.

Consider now the $(d-1)$-dimensional representation of the fracture. In (5.27) and (5.28), the mass fluxes normal to the surfaces $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ are combined in the auxiliary quantities $Q_{fn}^{(k)}$ and $P_{fn}^{(k)}$, where $k = 1{,}2$. Equations (5.45) and (5.46) imply that each of these quantities is conserved when crossing the fracture-medium interface, i. e.

$$Q_{fn}^{(k)} = Q_{mn}^{(k)} \quad \text{and} \quad P_{fn}^{(k)} = P_{mn}^{(k)} \quad \text{with } k = 1{,}2. \tag{5.50}$$

When the fracture is considered as a $(d-1)$-dimensional object, $\mathcal{S}^{(1)}$ and $\mathcal{S}^{(2)}$ "ideally" lie one upon the other, and coincide with the mean surface $\mathcal{S}$. Thus, the band-shaped boundary, $\mathcal{B}$, collapses to a closed line, i. e. the contour of $\mathcal{S}$, which is denoted by $\widehat{\mathcal{B}} \coloneqq \mathcal{B} \cap \mathcal{S}$. Although the equivalent fracture is now represented by $\mathcal{S}$, the sums

$$Q_{mn}^{(2)} + Q_{mn}^{(1)} = \left(\rho^{pW} q_{mn}^{(2)} - \rho' J_{mn}^{(2)}\right) + \left(\rho^{pW} q_{mn}^{(1)} - \rho' J_{mn}^{(1)}\right), \tag{5.51}$$

$$P_{mn}^{(2)} + P_{mn}^{(1)} = \left(c_m^{(2)} q_{mn}^{(2)} + J_{mn}^{(2)}\right) + \left(c_m^{(1)} q_{mn}^{(1)} + J_{mn}^{(1)}\right), \tag{5.52}$$

which are computed on the two sides of $\mathcal{S}$, do not vanish in general, for they represent the "jump" of physical quantities that are processed in the fracture and exchanged with the surrounding medium at the two sides of $\mathcal{S}$. In other words, the surface $\mathcal{S}$ has to be treated as a discontinuity surface. This situation is modeled by assigning to each point of $\mathcal{S}$ three values of concentration, $\hat{c}_m^{(1)} \neq \bar{c}_f \neq \hat{c}_m^{(2)}$, and three values of pressure,

$\hat{p}_m^{(1)} \neq \bar{p}_f \neq \hat{p}_m^{(2)}$, where $\hat{c}_m^{(k)}$ and $\hat{p}_m^{(k)}$ are approximations of $c_m^{(k)}$ and $p_m^{(k)}$ introduced in section 5.2.5. Then, the normal derivatives of pressure and concentration featuring in (5.45) and (5.46) are approximated in the following way:

$$\partial_n p_f\big|_{\mathcal{S}^{(k)}} \approx \frac{\hat{p}_m^{(k)} - \bar{p}_f}{\epsilon/2}, \quad \text{and} \quad \partial_n c_f\big|_{\mathcal{S}^{(k)}} \approx \frac{\hat{c}_m^{(k)} - \bar{c}_f}{\epsilon/2}, \quad k = 1{,}2. \tag{5.53}$$

Furthermore, since (5.51) - (5.52) require only the evaluation of the Darcy's specific discharge and Fick's diffusive flux at the two sides of $\mathcal{S}$, the following approximations for numerical computations are used:

$$\hat{q}_{fn}^{(k)} = -\frac{K_f}{\mu}\left[\frac{\hat{p}_m^{(k)} - \bar{p}_f}{\epsilon/2} - \left(\rho\left(\hat{c}_m^{(k)}\right) - \rho\left(\bar{c}_f\right)\right) g_n^{(k)}\right], \tag{5.54}$$

$$\hat{J}_{fn}^{(k)} = -D_f\left(1 - \frac{\rho'}{\rho^{pW}}\hat{c}_m^{(k)}\right)\frac{\hat{c}_m^{(k)} - \bar{c}_f}{\epsilon/2}, \tag{5.55}$$

where $k = 1{,}2$. The term $\left(\rho(\hat{c}_m^{(k)}) - \rho(\bar{c}_f)\right) g_n^{(k)}$ on the right-hand side of (5.54) requires some explanations.

Whereas the jumps (5.51) and (5.52) can be physically meaningful at the inner points of the surface $\mathcal{S}$, they may produce unphysical artifacts at $\widehat{\mathcal{B}}$. In fact, these artifacts affect the pressure distribution across the fracture, and the evaluation of the Darcy's specific discharge. The origin of this problem is the approximation of the buoyancy term $\rho(c_f)\mathbf{g}$ featuring in Darcy's law. In order to see that, the following consideration was made. A horizontal fracture (i. e. $\mathbf{g} \perp \mathcal{S}$) is considered, placed at the height Z_f of an absolute coordinate frame, and $c_m = c_f = \text{const}$. Because of averaging, the concentration c_f is replaced by its averaged value, $\bar{c}_f$, which, by its own definition, is constant along the z-axis of the fracture. Consequently, the mass density of the fluid-phase is constant along the z-axis too. This means that, in the hydrostatic case, the pressure drop across the surface $\mathcal{S}$ is $\epsilon\rho(\bar{c}_f)g$. This result, however, disagrees with the pressure distribution in the medium. Indeed, at the height Z_f, the pressure in the medium has a determined value and there is no jump. This disagreement generates parasite flows, whose velocity is proportional to ϵ. It should be underlined that the reason of the disagreement is the shrinking of the d-dimensional fracture to a surface (or a line in 2D), after which the physically correct pressure drop between the sides of the fracture becomes a discontinuity at the surface $\mathcal{S}$.

In order to reduce the parasite flows, the pressure field is corrected, and the buoyancy term normal to $\mathcal{S}$ in the approximated expression of Darcy's law (5.54). A physical explanation for this correction might be based on the concept of *excess mass* put forward by Murdoch /MUR 99/, MUR 05/. When the fracture is regarded as a d-dimensional object, pressure and brine concentration are defined both in $\mathcal{F}$ and in $\mathcal{M}$ and are continuous across the fracture-medium interface. However, for a $(d-1)$-dimensional fracture, pressure and brine concentration in the medium are prolonged over the wider region $\widehat{\mathcal{M}} = \mathcal{M} \cup \mathcal{F} \setminus \mathcal{S}$, while the fracture reduces to the surface $\mathcal{S}$. The correction consists of compensating for the pressure prolonged over $\mathcal{M}^+$ by accounting for the mass that is "removed" from the fracture and "given" to the surrounding medium. For example, for the half-region above $\mathcal{S}$, the reasoning here leads to the definition of the *excess* mass density:

$$\rho_f^{ex} := \rho(\bar{c}_f) - \rho\left(c_m\left(\frac{\epsilon}{2}\right)\right) = \rho(\bar{c}_f) - \rho\left(\hat{c}_m^{(2)}\right). \tag{5.56}$$

Use of (5.56) in the definition of Darcy's law, and approximating the normal derivative of the pressure as in (5.53) lead to (5.54). The concept of excess mass density is used in order to reformulate the normal component of Darcy's velocity at the fracture-medium interface of a $(d-1)$-dimensional fracture. For example, for $k = 2$, it is written

$$q_{fn}^{(2)} = -\frac{K_f}{\mu}\left[\partial_n p_f + \left(\rho(\bar{c}_f) - \rho\left(\hat{c}_m^{(2)}\right)\right) g_n\right]. \tag{5.57}$$

The approximated normal velocity $\hat{q}_{fn}^{(2)}$ in (5.54) is obtained by expanding the normal derivative of the pressure as explained in the first equality from (5.53).

An alternative explanation may be based on mimicking the consistent-velocity approximation. In order to illustrate the procedure leading to (5.54), a horizontal fracture is considered, and the pseudo-potential is defined by

$$\Psi_f(z) := p_f(z) + g\int_0^z \rho\left(c_f(\eta)\right) d\eta, \tag{5.58}$$

where $z = 0$ identifies the mean plane of the fracture. The component q_{fz} of $\mathbf{q}_f$ can be rewritten as

$$q_{fz}(z) = -\frac{K_f}{\mu}\frac{\mathrm{d}\Psi_f}{\mathrm{d}z}(z). \tag{5.59}$$

The value $q_{fz}(0)$ is approximated by the following finite difference

$$\begin{aligned} q_{fz}\left(\frac{\epsilon}{2}\right) &\approx -\frac{K_f}{\mu}\frac{\Psi_f(\epsilon/2)-\Psi_f(0)}{\epsilon/2} \\ &= -\frac{K_f}{\mu}\left[\frac{p_f(\epsilon/2)-p_f(0)}{\epsilon/2}+\frac{g}{\epsilon/2}\int_0^{\epsilon/2}\rho\left(c_f(\eta)\right)d\eta\right]. \end{aligned} \tag{5.60}$$

By approximating the brine concentration in the fracture by its boundary value, i. e. $c_f(\eta) \approx c_m(\epsilon/2)$, and using the fact that pressure is continuous across the surface $z = \epsilon/2$, (5.60) may rewritten as

$$q_{fz}\left(\frac{\epsilon}{2}\right) \approx -\frac{K_f}{\mu}\left[\frac{p_f(\epsilon/2)-p_f(0)}{\epsilon/2}+g\rho\left(c_m\left(\frac{\epsilon}{2}\right)\right)\right]. \tag{5.61}$$

It should be remarked that, at this stage, the fracture is regarded as very thin, but has not yet degenerated to a surface.

When the fracture "shrinks" to an equivalent surface, the pressure p_m is actually computed at $z = 0^+$. In order to see that, the hydrostatic pressure distribution π_m in the whole medium (i. e. in $\mathcal{M} \cup \mathcal{F}$). is considered. In the region above the fracture, one obtains (in global coordinates):

$$\pi_m(Z) = \pi_m\left(Z_f+\frac{\epsilon}{2}\right) - g\int_{Z_f+\frac{\epsilon}{2}}^{Z}\rho\big(c_m(Z')\big)dZ'. \tag{5.62}$$

where Z_f is the value of Z at which the fracture mean plane is located. It should be remarked that the function π_m is defined on the interval $Z \in \left[Z_f + \epsilon/2\,, L\right]$, where L is the height of the global domain.

When the fracture "shrinks" to a surface, the pressure in the medium above the fracture is actually defined on the wider interval $Z \in [Z_f, L]$. In order to account for that, the extension of π_m to the interval $Z \in [Z_f, L]$ is defined in the following way

$$\pi_m^+(Z) = \pi_m\left(Z_f+\frac{\epsilon}{2}\right) - g\int_{Z_f+\frac{\epsilon}{2}}^{Z}\rho\left(c_f(Z')\right)dZ'. \tag{5.63}$$

In "shrinking" the fracture to its mean plane, the quantity π_m^+ is are actually computed on the plane $Z = Z_f$. Since ϵ is small, the calculation of $\pi_m^+(Z_f)$ is approximated as follows:

$$\pi_m^+\big(Z_f^+\big) \approx \pi_m\left(Z_f+\frac{\epsilon}{2}\right) + g\,\frac{\epsilon}{2}\rho(\bar{c}_f). \tag{5.64}$$

Requiring (5.64) to be valid also in the dynamic case means that the pressure difference $\left(p_m^+(Z_f) - p_m\left(Z_f + \frac{\epsilon}{2}\right)\right)$ does not lead to a fluid motion. This leads to the condition

$$p_m^+(Z_f^+) \approx p_m\left(Z_f + \frac{\epsilon}{2}\right) + g\frac{\epsilon}{2}\rho(\bar{c}_f). \tag{5.65}$$

By substituting result (5.65) into (5.61), one obtains (in global coordinates)

$$q_{fz}(Z_f^+) \approx -\frac{K_f}{\mu}\left[\frac{p_m^+(Z_f^+) - p_f(Z_f)}{\epsilon/2} + g\left(\rho\left(c_m\left(Z_f + \frac{\epsilon}{2}\right)\right) - \rho(\bar{c}_f)\right)\right]. \tag{5.66}$$

This equation was found by considering the half of the fracture above the mean plane $Z = Z_f$. By extending this treatment also to the half of the fracture below the mean plane, and making the identifications $p_m^+(Z_f) \equiv \hat{p}_m^{(k)}$, $p_f(Z_f) \approx \bar{p}_f$, and $c_m(Z_f + \epsilon/2) \equiv \hat{c}_m^{(k)}$, the approximated normal velocity is obtained as

$$\hat{q}_{fz}^{(k)} := -\frac{K_f}{\mu}\left[\frac{\hat{p}_m^{(k)} - \bar{p}_f}{\epsilon/2} - \mathbf{g}\cdot\mathbf{n}^{(k)}\left(\rho\left(\hat{c}_m^{(k)}\right) - \rho(\bar{c}_f)\right)\right]. \tag{5.67}$$

Equation (5.54) is found by generalizing these results to the case of an oblique fracture.

By (5.54) - (5.58), the fluxes to be introduced in (5.39) - (5.40) are thus defined by

$$\begin{aligned}\hat{Q}_{fn}^{(k)}(\hat{p}_m^{(k)}, \bar{p}_f; \hat{c}_m^{(k)}, \bar{c}_f) = &-\rho^{pW}\frac{K_f}{\mu}\left[\frac{\hat{p}_m^{(k)} - \bar{p}_f}{\epsilon/2} - \left(\rho\left(\hat{c}_m^{(k)}\right) - \rho(\bar{c}_f)\right)g_n^{(k)}\right] \\ &+\rho' D_f\left(1 - \frac{\rho'}{\rho^{pW}}\hat{c}_m^{(k)}\right)\frac{\hat{c}_m^{(k)} - \bar{c}_f}{\epsilon/2},\end{aligned} \tag{5.68}$$

$$\begin{aligned}\hat{P}_{fn}^{(k)}(\hat{p}_m^{(k)}, \bar{p}_f; \hat{c}_m^{(k)}, \bar{c}_f) = &-c_\beta\frac{K_f}{\mu}\left[\frac{\hat{p}_m^{(k)} - \bar{p}_f}{\epsilon/2} - \left(\rho\left(\hat{c}_m^{(k)}\right) - \rho(\bar{c}_f)\right)g_n^{(k)}\right] \\ &-D_f\left(1 - \frac{\rho'}{\rho^{pW}}\hat{c}_m^{(k)}\right)\frac{\hat{c}_m^{(k)} - \bar{c}_f}{\epsilon/2},\end{aligned} \tag{5.69}$$

and the continuity conditions at the fracture-medium interface read

$$\hat{Q}_{fn}^{(k)} = Q_{mn}^{(k)}, \quad \text{and} \quad \hat{P}_{fn}^{(k)} = P_{mn}^{(k)}, \quad \text{with } k = 1{,}2. \tag{5.70}$$

The dependence of the quantities $\hat{Q}_{fn}^{(k)}$ and $\hat{P}_{fn}^{(k)}$ on their full list or arguments has been written explicitly. It should be noted that, in the first term on the right-hand side of

(5.69), the variable c_β has been introduced. This is done in order to use the **upwind** method for the computation of $\hat{P}_{fn}^{(k)}$. When this mass flux is "leaving" the fracture, i. e. $\hat{q}_{fn}^{(k)} \geq 0$, the variable c_β is set equal to the averaged concentration in the fracture, i. e. $c_\beta \equiv \bar{c}_f$. In the other case, i. e. $\hat{q}_{fn}^{(k)} < 0$, $c_\beta \equiv \hat{c}_m^{(k)}$ is set /BEA 90/.

In the case of permeable boundary $\mathcal{B}$, the interface conditions (5.47) - (5.48) have to be written in averaged form. This noticeably complicates the treatment of the problem. This difficulty is circumvented by requiring that the band-shaped lateral boundary of the fracture is impervious /ANG 09/. Accordingly, (5.47) - (5.48) are substituted with the following interface conditions valid on $\widehat{\mathcal{B}} = \mathcal{B} \cap \mathcal{S}$:

$$-\rho(\bar{c}_f)\frac{K_f}{\mu}\left(\partial_\tau \bar{p}_f - \rho(\bar{c}_f)g_\tau\right) = 0, \quad -D_f\left(1 - \frac{\rho'}{\rho^{pW}}\bar{c}_f\right)\partial_\tau \bar{c}_f = 0, \tag{5.71}$$

$$-\rho(\hat{c}_m)\frac{K_f}{\mu}(\partial_\tau \hat{p}_m - \rho(\hat{c}_m)g_\tau) = 0, \quad -D_f\left(1 - \frac{\rho'}{\rho^{pW}}\hat{c}_m\right)\partial_\tau \hat{c}_m = 0. \tag{5.72}$$

In this case, the brine concentration does not need to be continuous on $\widehat{\mathcal{B}}$.

In summary, the density-driven flow in a domain filled with porous medium with the $(d-1)$-dimensional fractures is modeled by equations (5.41) - (5.42) in $\widehat{\mathcal{M}} = \Omega \setminus \mathcal{S}$ and (5.39) - (5.40) on the surfaces $\mathcal{S}$ (the fracture network). The interface conditions are described by (5.70) - (5.72).

5.3 Model of contaminant transport in fractured porous medium

The model used in the program package r^{3}t consists only of the transport equations of the general form

$$\Theta_{\alpha,i}\partial_t c_{\alpha,i} + \nabla \cdot \left(\mathbf{V}_\alpha S_{\alpha,i} c_{\alpha,i} - \mathbf{D}_{\alpha,i}\nabla\left(S_{\alpha,i} c_{\alpha,i}\right)\right) = F_{\alpha,i}(c_\alpha), \tag{5.73}$$

where the index $\alpha \in \{m, f\}$ distinguishes between the surrounding bulk medium and the fracture, $c_{\alpha,i} = c_{\alpha,i}(t,x,y,z)$ denote the unknown concentrations for contaminant i, $\Theta_{\alpha,i}$ describes effects like the porosity of medium, the density of fluid and/or retardation factor, $S_{\alpha,i}$ describes the solubility (precipitation), $\mathbf{V}_\alpha$ is the transport velocity, $\mathbf{D}_{\alpha,i}$ the diffusion and dispersion tensors, and $F_{\alpha,i}(c_\alpha)$ represents the sources or sinks, in partic-

ular due to the reactions and the radioactive decay. For every i and α the term $F_{\alpha,i}$ may depend on $c_{\alpha,j}$ for all j, and the set of all these concentrations is denoted by c_α.

For the development of the model for the fractures represented as low-dimensional manifolds for (5.73) is followed the same approach as for the transport equation (5.11) in the model of the density-driven flow, see section 5.2 for details. To do this, it is assumed that the model coefficients like the porosity, retardation factors, precipitation coefficients etc. are piecewise constant in Ω. The rigorous mathematical derivation supposes also some special conditions on the diffusion and the dispersion tensors, but in the implementation, a more general form is considered. The application of this approach yields three sets of equations. Equations for the surrounding medium $\mathcal{M}^+ = \Omega \setminus \mathcal{S}$

$$\Theta_{m,i}\partial_t c_{m,i} + \nabla \cdot \left(\mathbf{V}_m S_{m,i} c_{m,i} - \mathbf{D}_{m,i}\nabla\left(S_{m,i}c_{m,i}\right)\right) = F_{m,i}(c_m), \tag{5.74}$$

equations for the fracture $\mathcal{S}$

$$\epsilon\Theta_{f,i}\partial_t \bar{c}_{f,i} + \nabla^\sigma \cdot \left(\epsilon\overline{\mathbf{V}}_f S_{f,i}\bar{c}_{f,i} - \epsilon\widehat{\mathbf{D}}_{f,i}\nabla^\sigma\left(S_{f,i}\bar{c}_{f,i}\right)\right) + \left(\hat{P}^{(2)}_{fn,i} + \hat{P}^{(1)}_{fn,i}\right) = \epsilon F_{f,i}\left(\bar{c}_f\right), \tag{5.75}$$

and the interface conditions

$$\left(\mathbf{V}_m S_{m,i} c_{m,i} - \mathbf{D}_{m,i}\nabla\left(S_{m,i}c_{m,i}\right)\right)\cdot \mathbf{n}^{(k)} = \hat{P}^{(k)}_{fn,i} \quad \text{with } k = 1,2, \tag{5.76}$$

on $\mathcal{S}$. In (5.75) - (5.76),

$$\hat{P}^{(k)}_{fn,i} \coloneqq V^{(k)}_{fn} S_{f,i} c_{\beta,i} - D_{f,i}\frac{S^{(k)}_{m,i}c^{(k)}_{m,i} - S_{f,i}\bar{c}_{f,i}}{\epsilon/2}, \quad k = 1,2. \tag{5.77}$$

In (5.74) - (5.77), the analogous notation as in section 5.2. is used. In particular, $\bar{c}_{f,i} \coloneqq \langle c_{f,i}\rangle$ and $c^{(k)}_{m,i}$ denotes the concentration on one side of the fracture. $\overline{\mathbf{V}}_f$ denotes the transport velocity along the fracture, and the normal velocities of the fluid at the sides of the fracture are denoted by $V^{(k)}_{fn}$. For $V^{(k)}_{fn} \geq 0$, the $c_{\beta,i}$ is set equal to the averaged concentration in the fracture, i. e. $c_{\beta,i} \equiv \bar{c}_{f,i}$, whereas for $V^{(k)}_{fn} < 0$, $c_{\beta,i} \equiv c^{(k)}_{m,i}$ is set. The $(d-1)\times(d-1)$-tensor $\widehat{\mathbf{D}}_{f,i}$ describes the diffusion and the dispersion in the plane of the fracture. In (5.75), the following approximation is implicitly used

$$\langle F_{f,i}(c_f)\rangle(t,x,y) \approx F_{f,i}\left(\bar{c}_f(t,x,y)\right), \tag{5.78}$$

where $\bar{c}_f$ denotes the set of $\bar{c}_{f,i}$ for all i.

For the numerical solution, system (5.74) - (5.77) must be closed by the specification of the boundary and initial conditions. On the edges of the fractures, the no-flux boundary conditions are imposed, as for the case of density-driven flow (see section 5.2.6).

5.4 Finite-volume discretization and numerical solvers

This section is devoted to the numerical methods for the solution of the models presented in sections 5.2 and 5.3. As both the models have similar properties from the point of view of the extension to the fractured medium whereas discretizations of the model of the density-driven flow require some additional constructions, the description of the numerical methods is confined on the model from section 5.2. A large part of changes extending d^{3}f and r^{3}t to the fractured media has been made in the UG library, so that both the programs refer to the same modules. This concerns the management of the grid, the placement of degrees of freedom and the basics of the finite-volume discretization. Application-specific parts of r^{3}t are implemented similarly to those in d^{3}f.

The main feature of the models from sections 5.2 and 5.3 is the presence of the partial differential equations on the domains of different dimensionalities. For example, in the model of the density-driven flow, the coupled systems (5.39) - (5.40) and (5.41) - (5.42) have dimensionalities $d-1$ and d, as well as separate unknown functions defined in the same domain. This feature is important for the discretization. Below a vertex-centered finite volume discretization of (5.39) - (5.40) and (5.41) - (5.42) is presented.

The finite-volume method, also known as the control volume /KAR 95/ or finite volume element methods /CAI 90/, is very popular in the numerical solution of PDEs. Application of this method to (5.1) - (5.2) in domains without fractures was presented in /FRO 98a/. A similar method is used for both (5.39) - (5.40) and (5.41) - (5.42). The essential difference of the presented method from the one described in /FRO 98a/ is that the $(d-1)$-dimensional fractures embedded in the d-dimensional domain have their own degrees of freedom. Furthermore, the proposed method allows to resolve the jumps of the d-dimensional part of the solution at the fractures.

5.5 Discretization grids and degrees of freedom

The following notation is used. The time interval is covered by a grid $\{t^n : n \geq 0\}$ with $0 = t^0 < \cdots < t^n < \cdots;\ \tau^n := t^n - t^{n-1}$. It is assumed that Ω is polygonal, and the $(d-1)$ -dimensional network of fractures $\mathcal{S} \subset \Omega$ is piecewise planar. The domain Ω is covered by a conformal triangulation $\mathbf{T}_\Omega$ that consists of triangles and quadrilaterals if $d = 2$ and tetrahedra, prisms and hexahedra if $d = 3$. It is supposed that for every element $e \in \mathbf{T}_\Omega$, $e \cap \mathcal{S}$ is either empty or consists only of corners, *whole* sides and *whole* edges of e. Thus, $\mathcal{S}$ is covered by the $(d-1)$-dimensional triangulation $\mathbf{T}_\mathcal{S} := \{e \cap \mathcal{S} : e \in \mathbf{T}_\Omega, e \cap \mathcal{S} \text{ is a side of } e\}$. To simplify the notation, it is assumed that every $e \in \mathbf{T}_\Omega$ has at most one side on $\mathcal{S}$ and at most one on $\partial\Omega$. Similarly, $e \in \mathbf{T}_\mathcal{S}$ may not have more than one side on $\widehat{\mathcal{B}}$. The generalization is straightforward.

Denote by Ω_h the set of all grid points, i. e. corners of the elements of $\mathbf{T}_\Omega$. Let $\mathcal{S}_h := \Omega_h \cap \mathcal{S}$. For the approximation of the discontinuities on $\mathcal{S}$, grid functions are considered that may have several values at every $x \in \mathcal{S}_h$. To define them properly, a special enumeration of the grid points is introduced so that several indices correspond to the same x. Degrees of freedom are uniquely assigned to these indices and not directly to geometric positions.

To this end, for every $x \in \Omega_h$, consider a ball $B(x) = \{y : |y - x|_2 < 1/2\ \mathrm{dist}(x, \mathbf{T}_\Omega)\}$, where $\mathrm{dist}(x, \mathbf{T}_\Omega)$ is the minimum distance between x and those sides and edges of elements $e \in \mathbf{T}_\Omega$ which do not contain x. Fractures $\mathcal{S}$ cut these balls into disjoint open subsets (cf. Fig. 5.2). Denote these subsets of $B(x)$ for all $x \in \Omega_h$ by $B_1, \ldots, B_N$, where $N \geq |\Omega_h|$ is the total number of them. The closure of every B_i contains only one $x \in \Omega_h$, and this point is denoted by x_i. Under this enumeration, there may be $x_i = x_j \in \mathcal{S}_h$ for $i \neq j$. For simple straight fractures, points $x \in \mathcal{S} \setminus \widehat{\mathcal{B}}$ have two different indices, and the intersection points of the fractures have even more ones. For $x \in \Omega_h \setminus \mathcal{S}_h$ or $x \in \mathcal{S}_h \cap \widehat{\mathcal{B}}$, $\mathcal{S}$ does not split $B(x)$, that means $\mathcal{S}$ cuts $B(x)$ into one part. For $e \in \mathbf{T}_\Omega$, denote $\Lambda^e := \{i : 1 \leq i \leq N, x_i \in e, B_i \cap e \neq \emptyset\}$. These are indices of corners of e regarding the orientation of e with respect to the fractures.

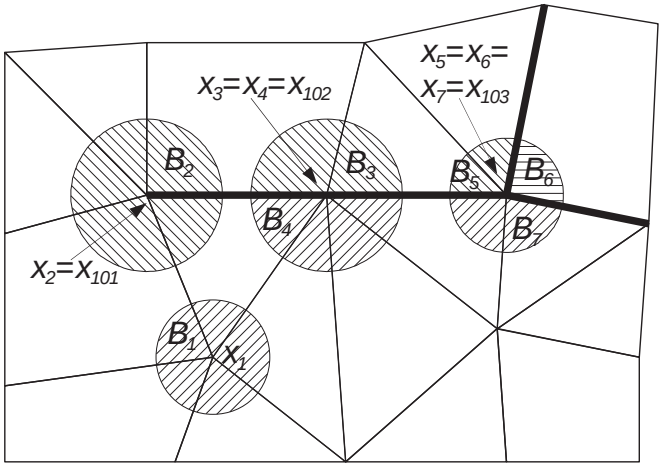


Fig. 5.2 Enumeration of the grid nodes in a piece of a grid with *N*=100. The thick lines denote the fractures

To represent the numerical solution of (5.41) - (5.42), for time t^n and index $1 \le i \le N$ the degrees of freedom c_{mi}^n and p_{mi}^n are introduced. For each i, $\hat{c}_m(t,x)$ and $\hat{p}_m(t,x)$ in the solution of (5.41) - (5.42) are continuous in B_i. In the numerical solution, c_{mi}^n and p_{mi}^n approximate the limits

$$c_{mi}^n \approx \lim_{x \to x_i, x \in B_i} \hat{c}_m(t^n, x), \quad p_{mi}^n \approx \lim_{x \to x_i, x \in B_i} \hat{p}_m(t^n, x). \tag{5.79}$$

Piecewise linear functions $c_{mh}^n, p_{mh}^n : \Omega \to \mathbb{R}$ are defined by the linear interpolation of the values c_{mi}^n and p_{mi}^n for $i \in \Lambda^e$ in every $e \in \mathbf{T}_\Omega$. These functions may be discontinuous only at $\mathcal{S}$.

In the fractures, grid functions c_{fh}^n and p_{fh}^n are introduced approximating c_f and p_f at time t^n independently on c_{mh}^n and p_{mh}^n. Let $N_f := |\mathcal{S}_h|$. Additionally to the enumeration above, indices $N+1, \dots, N+N_f$ are assigned to all the points from $\mathcal{S}_h$, so that $\mathcal{S}_h = \{x_{N+1}, \dots, x_{N+N_f}\}$. Then $c_{fh}^n, p_{fh}^n : \mathcal{S} \to \mathbb{R}$ are continuous piecewise linear functions with nodal values c_{fi}^n and p_{fi}^n at $x_i \in \mathcal{S}_h$. For $e \in \mathbf{T}_\mathcal{S}$, let $\Lambda^e := \{i : N < i \le N + N_f, x_i \in e\}$.

Furthermore, let $\Lambda_i^e := \Lambda^e \setminus \{i\}$, if $i \in \Lambda^e$, and $\Lambda_i^e := \emptyset$ otherwise. Then $\Lambda_i := \bigcup_{e \in \mathbf{T}_\Omega} \Lambda_i^e$ is the set of all the indices of neighbouring grid points of x_i. For $1 \le i \le N$, $\Lambda_i \subset \{1, \dots, N\}$, whereas for $i > N$, $\Lambda_i \subset \{N+1, \dots, N+N_f\}$. Besides this, for $i \le N$, let $\hat{\Lambda}_i := \{j : N < j \le N + N_f, x_i = x_j\}$ and, for $i > N$, let $\hat{\Lambda}_i := \{j : 1 \le j < N, x_i = x_j\}$. Sets $\hat{\Lambda}_i$ represent the relations between the "fracture DOF (degree of freedom) indices" and the "bulk medium DOF indices" of points $x_i \in \mathcal{S}_h$. Note that for $i \le N$, $|\hat{\Lambda}_i| \le 1$.

5.5.1 The finite-volume discretization

With each i, $1 \le i \le N$, a computation cell ("control volume") is associated by constructing a conformal dual mesh of finite volumes $V_i \subset \mathbb{R}^d$. The choice are the so-called barycenter based control volumes. V_i is defined as a union of V_i^e for all $e \in \mathbf{T}_\Omega$ such that $i \in \Lambda^e$. To get V_i^e, the element e is cut by the segments of the straight lines, connecting the barycenter of this element with the centers of its sides (for $d = 2$), or by the segments of the plains spanning the barycenter of the element, the centers of the edges and the barycenters of the sides (for $d = 3$). Then V_i^e is the part of e containing x_i. The segments are denoted by γ_{ij}^e, i. e. $\gamma_{ij}^e := e \cap \partial V_i \cap \partial V_j$ for $j \le N$. Besides, $\gamma_{ij}^e := e \cap \partial V_i \cap \mathcal{S}$ is defined with $j \in \hat{\Lambda}_i$ and $\gamma_{i0}^e := e \cap \partial V_i \cap \partial\Omega$. Then

$$\partial V_i = \bigcup_{e: i \in \Lambda^e} \bigcup_{j \in \Lambda_i^e \cup \hat{\Lambda}_i \cup \{0\}} \gamma_{ij}^e. \tag{5.80}$$

By $\mathbf{n}_{ij}^e$ the unit normal vector to γ_{ij}^e is denoted pointing out of V_i, i. e. $\mathbf{n}_{ji}^e = -\mathbf{n}_{ij}^e$. For $j \in \hat{\Lambda}_i$ (i. e. when γ_{ij}^e lies on a fracture), normals $\mathbf{n}^{(1)}$ and $\mathbf{n}^{(2)}$ are constant on γ_{ij}^e. One of them is $\mathbf{n}_{ij}^e$, the other one $\left(-\mathbf{n}_{ij}^e\right)$.

Control volumes V_i are used in the discretization of (5.41) (5.42). For every i, $1 \le i \le N$, if x_i does not lie on the Dirichlet boundary, (5.41) (5.42) are integrated over V_i. After the application of the divergence theorem, one gets:

$$\sum_{e,j} \int_{\gamma_{ij}^e} \left(\rho^{pW} \widehat{\mathbf{q}}_m - \rho' \hat{\mathbf{j}}_m\right) \cdot \mathbf{n}_{ij}^e ds = \rho^{pW} \int_{V_i} S_m dx, \tag{5.81}$$

$$\phi_m \int_{V_i} \partial_t c_{mh} + \sum_{e,j} \int_{\gamma_{ij}^e} \left(c_{mh}^n \widehat{\mathbf{q}}_m + \hat{\mathbf{j}}_m\right) \cdot \mathbf{n}_{ij}^e ds = \int_{V_i} \hat{c}_{m,S} S_m dx, \tag{5.82}$$

where the summation runs over all $e \in \mathbf{T}_\Omega$, such that $i \in \Lambda^e$, and $j \in \Lambda_i^e \cup \hat{\Lambda}_i \cup \{0\}$. In this work, for the approximation of the time derivative the backward Euler scheme is used:

$$\int_{V_i} \partial_t c_{mh} \approx |V_i| \frac{c_{mi}^n - c_{mi}^{n-1}}{\tau^n}. \tag{5.83}$$

The summands in (5.81) – (5.82) corresponding to the indices $i \in \Lambda_i^e \cup \{0\}$ depend only on c_{mh}^n and p_{mh}^n. For them, the approximation from /FRO 98a/ is used. In particular, the so-called consistent velocity is used for $\mathbf{q}_m(c_{mh}^n, p_{mh}^n)$, cf. /FRO 98/, /FRO 96a/, and an upwind method is applied for the discretization of the convection term $c_{mh}^n \mathbf{q}_m$. Consider the summand with $j \in \hat{\Lambda}_i$. It requires a special treatment. Let e be such an element that

$i \in \Lambda^e$ and $\gamma_{ij}^e \neq \emptyset$. Let $k \in \{1,2\}$ be such that $\mathbf{n}_{ij}^e = -\mathbf{n}^{(k)}$. (5.70) is used and the following approximations:

$$\int_{\gamma_{ij}^e} \left(\rho^{pW} \widehat{\mathbf{q}}_m - \rho' \hat{\mathbf{J}}_m\right) \cdot \mathbf{n}_{ij}^e ds = \int_{\gamma_{ij}^e} \hat{Q}_{fn}^{(k)} ds \approx \left|\gamma_{ij}^e\right| \hat{Q}_{fn}^{(k)}(p_{mh}^n, \bar{p}_{fh}^n; c_{mh}^n, \bar{c}_{fh}^n), \tag{5.84}$$

$$\int_{\gamma_{ij}^e} \left(c_{mh}^n \widehat{\mathbf{q}}_m + \hat{\mathbf{J}}_m\right) \cdot \mathbf{n}_{ij}^e ds = \int_{\gamma_{ij}^e} \hat{P}_{fn}^{(k)} ds \approx \left|\gamma_{ij}^e\right| \hat{P}_{fn}^{(k)}(p_{mh}^n, \bar{p}_{fh}^n; c_{mh}^n, \bar{c}_{fh}^n). \tag{5.85}$$

These approximations are algebraic functions of c_{mh}^n, p_{mh}^n, $\bar{c}_{fh}^n$ and $\bar{p}_{fh}^n$.

For the discretization of (5.39) - (5.40), $(d-1)$-dimensional computation cells are constructed in $\mathcal{S}$: With every $i > N$ the $(d-1)$-dimensional control volume is associated

$$S_i := \bigcup_{j \in \hat{\Lambda}_i} \bigcup_{e \in \mathbf{T}_\Omega : j \in \Lambda^e} \gamma_{ij}^e. \tag{5.86}$$

At the intersections of the fractures, S_i may lie in several intersecting planes. For $e \in \mathbf{T}_\mathcal{S}$, boundary segments σ_{ij}^e of S_i are introduced analogously to γ_{ij}^e: For $i, j > N$, $\sigma_{ij}^e := e \cap \partial S_i \cap \partial S_j$. Besides this, σ_{i0}^e is the intersection of $e \cap \partial S_i$ with the edge of the fracture. For every σ_{ij}^e, $\mathbf{n}_{ij}^e$ is the unit normal vector that lies in the plane of e and points out of S_i. Note that S_i are barycenter based control volumes, too. Integration of (5.39) - (5.40) over S_i yields:

$$\begin{aligned} \epsilon \sum_{e,j} \int_{\sigma_{ij}^e} \left(\rho^{pW} \langle \mathbf{q}_f^\sigma \rangle - \rho' \langle \mathbf{J}_f^\sigma \rangle\right) \cdot \mathbf{n}_{ij}^e dl + \int_{S_i} \left(\hat{Q}_{fn}^{(2)} + \hat{Q}_{fn}^{(1)}\right) ds \\ = \epsilon \rho^{pW} \int_{S_i} \bar{S}_f ds, \end{aligned} \tag{5.87}$$

$$\begin{aligned} \phi_f \epsilon \int_{S_i} \partial_t \bar{c}_{fh} ds + \epsilon \sum_{e,i} \int_{\sigma_{ij}^e} \left(\bar{c}_{fh} \langle \mathbf{q}_f^\sigma \rangle + \langle \mathbf{J}_f^\sigma \rangle\right) \cdot \mathbf{n}_{ij}^e dl + \int_{S_i} \left(\hat{P}_{fn}^{(2)} + \hat{P}_{fn}^{(1)}\right) ds \\ = \epsilon \int_{S_i} \bar{c}_{fh} \bar{S}_f ds, \end{aligned} \tag{5.88}$$

where the summation runs over all $e \in \mathbf{T}_\mathcal{S}$, such that $i \in \Lambda_i^e$, and $j \in \Lambda_i^e \cup \hat{\Lambda}_i \cup \{0\}$. For the approximation of the time derivative in (5.88), the backward Euler scheme is used, too:

$$\int_{S_i} \partial_t \bar{c}_{fh} ds \approx |S_i| \frac{c_{fi}^n - c_{fi}^{n-1}}{\tau^n}. \tag{5.89}$$

As σ_{ij}^e are segments of straight lines (for $d = 3$) or points (for $d = 2$), the integrals are discretized over them in (5.87) - (5.88) using the method from /FRO 98a/ formulated in $d-1$ dimensions for $\bar{c}_{fh}^n$ and $\bar{p}_{fh}^n$. For $\langle \mathbf{q}_f^\sigma \rangle$, the consistent velocity from /FRO 96a/ is

used. For the convection term in (5.88), a $(d-1)$-dimensional version of the upwind method for the discretization of (5.82) is applied.

To approximate the integrals of $\hat{Q}_{fn}^{(k)}$ and $\hat{P}_{fn}^{(k)}$ (cf. (5.68) - (5.69)) in (5.87) - (5.88), the fact is used that $S_i \coloneqq \bigcup_{j\in\hat{\Lambda}_i} \bigcup_{e:j\in\Lambda^e,\mathbf{n}_{ij}^e=-\mathbf{n}^{(k)}} \gamma_{ij}^e$ (cf. (5.86)) where γ_{ij}^e are disjoint planar sets:

$$\int_{S_i} \hat{Q}_{fn}^{(k)} ds \approx \sum_{j\in\hat{\Lambda}_i} \sum_{e:j\in\Lambda^e,\mathbf{n}_{ij}^e=-\mathbf{n}^{(k)}} \left|\gamma_{ji}^e\right| \hat{Q}_{fn}^{(k)}(p_{mj}^n, \bar{p}_{fi}^n; c_{mj}^n, \bar{c}_{fi}), \tag{5.90}$$

$$\int_{S_i} \hat{P}_{fn}^{(k)} ds \approx \sum_{j\in\hat{\Lambda}_i} \sum_{e:j\in\Lambda^e,\mathbf{n}_{ij}^e=-\mathbf{n}^{(k)}} \left|\gamma_{ji}^e\right| \hat{P}_{fn}^{(k)}(p_{mj}^n, \bar{p}_{fi}^n; c_{mj}^n, \bar{c}_{fi}). \tag{5.91}$$

The contribution of (5.90) - (5.91) to (5.87) - (5.88) is exactly the same as the contribution of the terms (5.84) - (5.85) to (5.81) - (5.82) so that the entire discretization is conservative w.r.t. the mass of the total fluid phase and mass of the salt.

Conditions (5.71) - (5.72) at $\widehat{\mathcal{B}}$ are natural boundary conditions for the finite-volume discretization. They introduce no additional terms in (5.81) - (5.82) and (5.87) - (5.88). Further boundary conditions should be used for $\bar{c}_f$ and $\bar{p}_f$ at $\mathcal{S} \cap \partial\Omega$.

Using the introduced approximations of the integrals in (5.81) - (5.82) for all V_i and in (5.87) - (5.88) for all S_i, one obtains a sparse system of $N + N_f$ nonlinear algebraic equations. The solution of this system approximate the analytical solution of the model derived in section 5.2.

Technically, the assembling of this nonlinear system can be implemented as a cycle over only the elements of $\mathbf{T}_\Omega$. When assembling the contribution of, say, triangle (x_1, x_2, x_4) in Fig. 5.2, not only the local matrices are computed for (5.81) - (5.82), but also a part of the local matrix for (5.87) - (5.88) for the segment $[x_{101}, x_{102}]$. This part consists of (a) the integrals of $\hat{Q}_{fn}^{(k)}$ and $\hat{P}_{fn}^{(k)}$ for only one k such that $\mathbf{n}^{(k)}$ is the inner normal for the triangle and (b) the terms with the time derivative and the integrals over σ_{ij}^e multiplied by $1/2$. As soon as the contribution of the element on the opposite side of the fracture is assembled, the terms (a) with the second k as well as the same value of terms (b) are added to the local matrix of $[x_{101}, x_{102}]$ (so that the factors $1/2$ sum up to unity). Thus distinguishing between the sides of the fractures can be avoided.

5.5.2 Solution of the discretized system

In programs d^3f and r^3t, implicit time discretizations are used. This means, the discretization of the model in space and time leads to a large sparse system of algebraic equations in every time step. Computation of the stationary flow with d^3f requires the solution of a large sparse non-linear system of algebraic equations, too. In r^3t, further methods like operator splitting, can be used for the reaction terms.

In the simulations, these non-linear systems are solved by the Newton method. To make use of the sparsity of the linear systems in the iterations of the nonlinear solver, they are solved by the biconjugate gradient stabilized method (BiCGStab) with the geometric multigrid preconditioning (cf. /BAR 93/). In the multigrid cycle, the ILU_β-smoothers and the Gaussian elimination are used as the coarse grid solver. This multigrid preconditioner proved to be very efficient in the case of not too complicated geometries such that the coarse grid is not extremely detailed. The matrices in the grid hierarchy are not computed by the Galerkin formula but assembled as the Jacobians of the discretized nonlinear systems for each grid.

The quality of the smoothing by the ILU_β-decompositions in the geometric multigrid method strongly depends on the ordering of the grid nodes. The algorithm implemented in UG for the lexicographical ordering of the nodes is based on the geometric properties of the grid. Applied directly to the grids described in section 5.5, this algorithm produces a random ordering of the nodes situated at the same geometric positions on the fractures. For this ordering, the BiCGStab iteration demonstrates typically a very pour convergence.

To avoid this situation, the geometric positions of the vertices $v \in \mathcal{V}^*$ are changed to make the fracture “thick”, i. e. these vertices are moved towards the surrounding medium away from $\mathcal{S}$. The positions of the vertices from $\mathcal{V}_\mathcal{S}$ are not changed. The distance at which the vertices are moved is very small in comparison with the diameters of the grid elements, so that this transformation does not change the geometry of the grid essentially. For this new grid, the standard algorithms from UG produce a proper ordering. After the ordering phase, the original geometric positions of the vertices are restored whereas the ordering is kept unchanged. For the ordering constructed in this way, the BiCGStab iteration with the geometric multigrid preconditioner achieves very good convergence rate.

Nevertheless, in the case of a large number of fractures, the coarse grid becomes very detailed and the efficiency of the geometric multigrid preconditioner is essentially influenced by the coarse grid solver. In these situations, the Gaussian elimination can be replaced with the filtering algebraic multigrid method.

A further difficulty arises in the simulations of the density-driven and stationary flows with d^{3}f due to the floating-point representation of the nodal values of the pressure. In the model derived in section 5.2, up to the Dirichlet boundary conditions, the pressure is used only in the form of the gradients or the finite differences at the fractures. The discretized model depends only on the finite differences of the pressures at neighbouring grid nodes, i. e. only on the local variations of the pressure. But these variations are typically very small in comparison with the values of the pressure in the whole domain, especially for the fine grids. This leads to the loss of the absolute precision in the numerical approximation of the gradients due to the cancellation phenomena in the floating-point arithmetic. Then the numerical solvers cannot achieve the prescribed absolute precision and stop to converge. This situation can be avoided by setting a greater threshold for the absolute precision, but this reduces the accuracy of the solution, and, furthermore, this threshold depends on the (usually a priori unknown) maximum pressure in the domain.

As a remedy, the computation of the discrete solution in every time step has been split into two phases. In the first phase, the system is solved with a relatively large threshold of the absolute precision. It should be denoted the result of this computation by $(c_{\mathrm{ref}}, p_{\mathrm{ref}})$. Then p_{ref} is used as a reference pressure in the second phase of the computation. In that phase, the deviation $\delta p \coloneqq p - p_{\mathrm{ref}}$ is used as the unknown function instead of p. The gradient in the model is represented as $\nabla p = \nabla \delta p + \nabla p_{\mathrm{ref}}$ (and analogously with the gradients in the fractures), where ∇p_{ref} is known so that the terms depending on it can be moved into the right-hand side. The local variations of δp are usually comparable with the values of δp, and the cancellation phenomena do not reduce the precision significantly. The result of the second phase of the computation is the concentration c and the correction δp, both computed with a practically acceptible precision.

Basing on p_{ref} and δp, the correct Darcy velocity can be computed, for example, for the visualization. Note, however, that a specially implemented procedure which uses the couple $(p_{\mathrm{ref}}, \delta p)$ should be used for this. Merely storing the sum $p = p_{\mathrm{ref}} + \delta p$ and the computation of the velocity from it makes no sense as the cancellation phenomena

take place during the summation. Furthermore, it should be noted that the precision of the computation of p (in contrast to that of c) in one time step does not play any role in the computation of the next time step as the model does not include $\partial_t p$ explicitly.

5.6 Numerical experiments

To validate the derived model and to verify the functionality of the software, several numerical experiments are carried out. The results from the simulations are compared with the low-dimensional representation of the fractures with the results from the experiments with the full-dimensional fractures. Some of these tests with the program d^{3}f are present here. Further tests can be found in /GRI 10a/, /GRI 11/ and /STI 11/.

5.6.1 Tests in 2d

The problem of the density-driven flow in the whole region of observation is formulated by using (5.39) - (5.40) with (5.29), (5.36) in the fracture, and (5.41) - (5.44) in the embedding medium. The interface conditions are expressed by (5.67) - (5.71). In order to validate these results, simulations with the d-dimensional fracture (cf. /JOH 02/, /JOH 06a/) are performed. In this case, (5.1) - (5.5) is solved, with the appropriate coefficients, both in the fracture and in the embedding medium, and the interface conditions (5.45) - (5.49) are used. Also in this case, the results are shown in terms of velocity profile and isolines of the mass fraction.

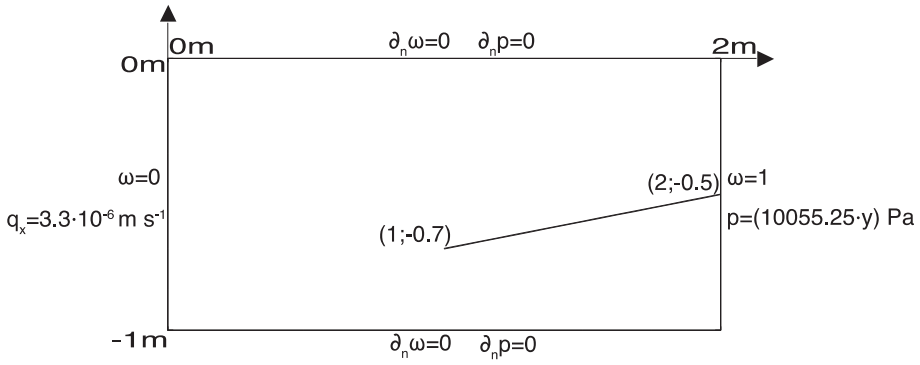


Fig. 5.3 Geometry and boundary conditions for the modified Henry problem featuring a fracture

The following examples are computed for both the $\boldsymbol{d}$- and the $(d-1)$-dimensional approach: (i) oblique fracture, (ii) junction of two oblique fractures. A modified version of the classic seawater intrusion problem by Henry /HEN 64a/ is considered where the domain, a rectangle $2 \times 1\ \mathrm{m}^2$, features a fracture (cf. Fig. 5.3). At the top ($z = 0$) and the bottom ($z = -1\ \mathrm{m}$), zero-flux boundary conditions are imposed for both the flow and the transport equations. At the inland side (left, $x = 0$), $c_m = 0\ \mathrm{kg} \cdot \mathrm{m}^{-3}$ is set ($\omega = 0$, fresh water) and a constant flux ($q = -3.3 \cdot 10^{-5}\ \mathrm{m} \cdot \mathrm{s}^{-1}$, cf. /SIM 04/) is prescribed. At the sea side (right, $= 2\ \mathrm{m}$), $c_m = \bar{c}_f = 1.025\ \mathrm{kg} \cdot \mathrm{m}^{-3}$ ($\omega = 1$) and hydrostatic pressure are imposed. The parameters used for the computations are listed in

Tab. 5.1 Parameters used for the computations

Symbol	Quantity	Value
D_m	Diffusion coefficient in the medium	$6.6 \cdot 10^{-6}\ \mathrm{m}^2 \cdot \mathrm{s}^{-1}$
D_f	Diffusion coefficient in the fracture	$13.2 \cdot 10^{-6}\ \mathrm{m}^2 \cdot \mathrm{s}^{-1}$
$\mathbf{g}$	Gravity	$9.81\ \mathrm{m} \cdot \mathrm{s}^{-1}$
K_m	Permeability of the medium	$1.019368 \cdot 10^{-9}\ \mathrm{m}^2$
K_m	Permeability of the fracture	$1.019368 \cdot 10^{-6}\ \mathrm{m}^2$
ϕ_m	Porosity of the medium	0.35
ϕ_f	Porosity of the fracture	0.7
μ	Viscosity	$10^{-3}\ \mathrm{kg} \cdot \mathrm{m}^{-1} \cdot \mathrm{s}^{-1}$
ρ^{pW}	Density of water	$10^{-3}\ \mathrm{kg} \cdot \mathrm{m}^{-3}$
ρ^{pB}	Density of brine	$1.025 \cdot 10^{3}\ \mathrm{kg} \cdot \mathrm{m}^{-3}$
a_α^t, a_α^ℓ	Dispersivity lengths	0

In the first experiment the thin fracture ($\epsilon = 1.96116 \cdot 10^{-3}$ m) has endpoints at $(x, z) = (1\text{ m}, -0.7\text{ m})$ and $(x, z) = (2\text{ m}, -0.5\text{ m})$ (cf. Fig. 5.3 and Fig. 5.4). The fundamental result is that the velocity in the fracture produces a deflection of the isolines of the mass fraction, which is maximal at the right end of the fracture (sea side). The magnitude and the direction of the velocity in the fracture (from left to right) hinders the spreading of the brine in the fracture. This happens because of the strong permeability contrast between the fracture and the medium. Slight variations of the slope of the fracture do not change the essence of the described phenomenology.

Simulations were performed on a grid with about $6 \cdot 10^4$ grid nodes and the timestep was chosen $\tau^n = 15$ sec. In the full-dimensional representation, fracture width was resolved by 8 layers of elements inside the fracture.

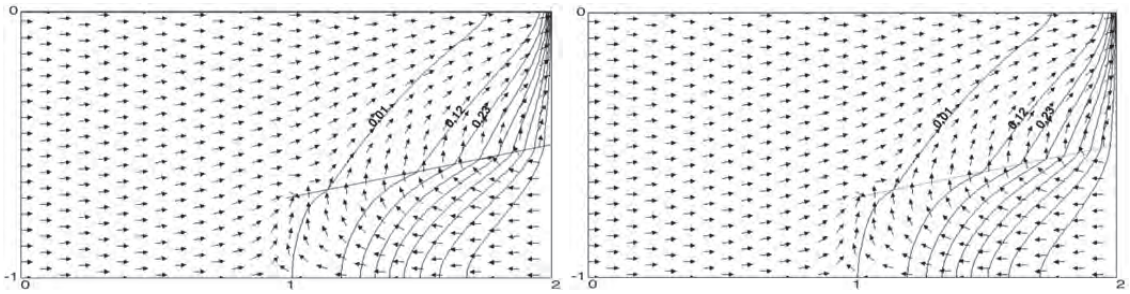


Fig. 5.4 Isolines of the mass fraction (corresponding to $\boldsymbol{\omega_i = 0.01 + 0.11i}$, $\boldsymbol{i = 0, \ldots, 9}$) and velocity directions of the (left) $\boldsymbol{d}$- and (right) $(\boldsymbol{d-1})$-dimensional simulation of Henry's problem with a fracture of width $\boldsymbol{\epsilon = 1.96116\text{ m}}$ at time $\boldsymbol{t = 25}$ min

To compare the results of the simulations with the low- and the full-dimensional representations of the fracture, the point with $x = 1.5$ m on the middle line of the fracture is chosen. Then the concentration $\bar{c}_f$ from the simulation with the low-dimensional fracture representation is compared with the concentration obtained by averaging the concentration from the simulation with the full-dimensional representation over the normal cross-section. As the full-dimensional simulation computes the mass fraction ω, it is transformed into the concentration by formula $c(\omega) = \frac{\rho^{pW}\rho^{pB}\omega}{\rho^{pW}+(\rho^{pW}-\rho^{pB})\omega}$. For convenience, the non-dimensional concentrations (volume fractions) $\varphi_f \coloneqq c_f/\rho^{pB}$ are compared whose maximum value is 1. The absolute error between the non-dimensional concentrations of the two simulations is $E(\varphi_f) \coloneqq \left|\varphi_f^{\text{full}} - \bar{\varphi}_f^{\text{low}}\right|$. These results are presented in Fig. 5.5.

As a further test, the intersection of two fractures with $= 1.7888 \cdot 10^{-3}$ m is presented. The first fracture has the endpoints at $(1\,\mathrm{m}, -0.25\,\mathrm{m})$ and $(2\,\mathrm{m}, -0.75\,\mathrm{m})$, and the second one $(1\,\mathrm{m}, -0.75\,\mathrm{m})$ and $(2\,\mathrm{m}, -0.25\,\mathrm{m})$. All the other model parameters are the same as in the previous test.

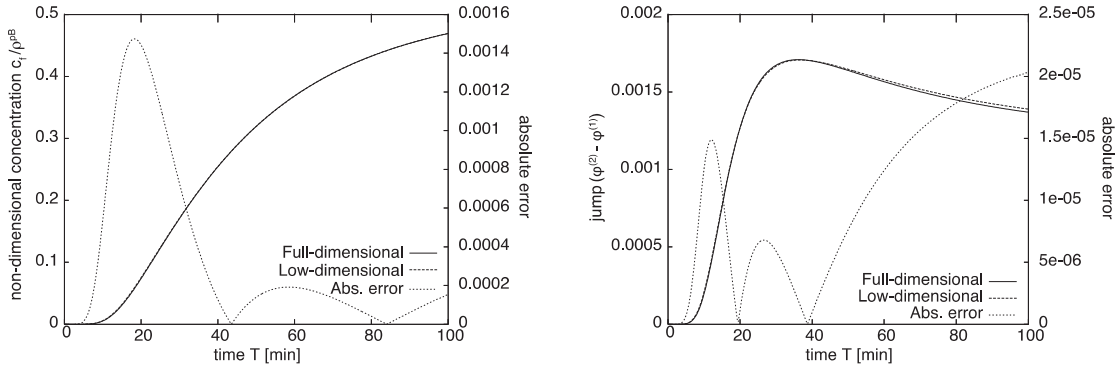


Fig. 5.5 Comparisons of full- and low-dimensional simulations at *x*=1.5 m non-dimensional concentrations (left) in the fracture and its absolute error, jump of the non-dimensional concentrations (right) between the sides of the fracture and its absolute error

Simulations using both the low- and full-dimensional representations of the fractures produce similar pictures of the flow field and the distribution of the mass fraction, cf. Fig. 5.6. The deflection of the isoline cutting the lower part of the second fracture (on the right of the intersection point) points towards the inland side. In all other cases, the isolines of the mass fraction are deflected towards the sea side (right). Thus, along the second fracture the deflection of the isolines undergoes a transition, which takes place at the intersection point. This is also the center of the vortices.

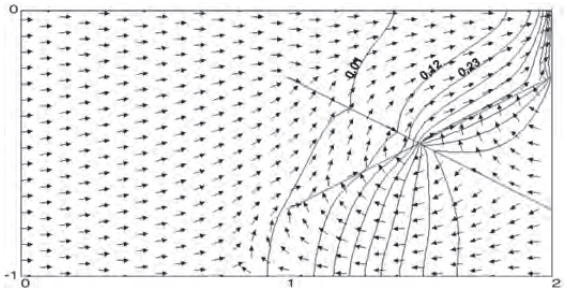


Fig. 5.6 Isolines of the mass fraction (corresponding to $\boldsymbol{\omega_i = 0.01 + 0.11i}$, $\mathbf{i} = \mathbf{0}, \ldots, \mathbf{9}$) and velocity directions from the simulation with an intersection of fractures at time $\mathbf{t = 25}$ min

Thus, for sufficiently thin fractures, the simulations with the low-dimensional representation of the fracture show qualitatively and quantitatively good agreement with the results obtained in the simulations with the full-dimensional representation of the fracture. Some restrictions of the model are discussed in /GRI 10a/, /GRI 11/and /STI 11/.

5.6.2 Test in 3d

The usage of the programs in 3d does not principally differ from that in 2d. As an example, a three-dimensional variant of the problem from section 5.6.1 is used. The domain is an extrusion of the domain from section 5.6.1 in the y-direction, so that it is a parallelepiped of size $2 \times 1 \times 1\ \mathrm{m}^3$. It contains two crossing fractures, a slightly oblique one and a vertical one. Fig. 5.7 presents the location of the fractures and a part of the boundary grid. The same model parameters (in particular, in the fractures) and boundary conditions as in section 5.6.1 are used. The horizontal fracture has the aperture $2 \cdot 10^{-3}\ \mathrm{m}$, and the vertical one $2.5 \cdot 10^{-3}\ \mathrm{m}$.

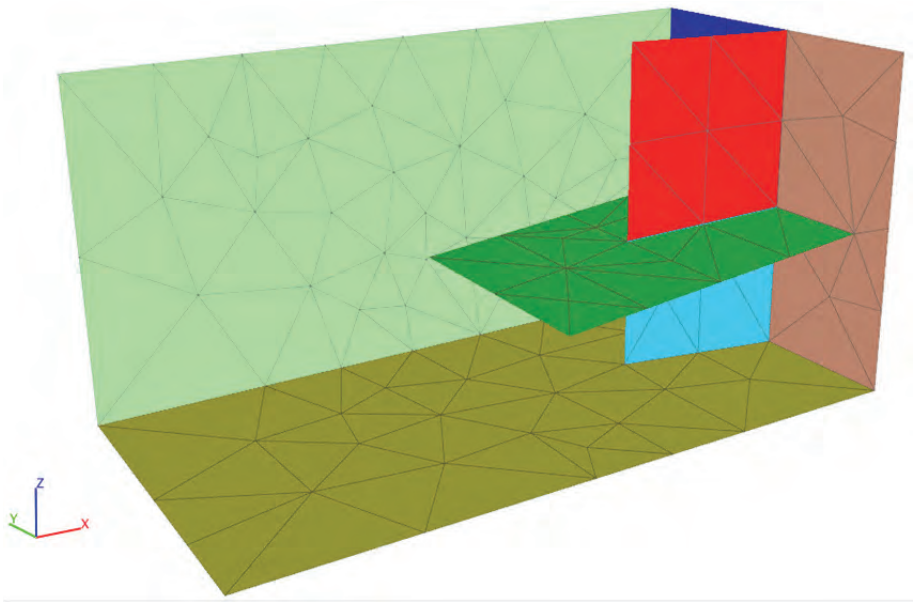


Fig. 5.7 Location of the fractures, boundary grid and decomposition into the boundary surfaces in the 3d test

Fig. 5.8 shows the isolines for $c = 0.1$ (yellow, transparent) and $c = 0.3$ (blue, solid) at time 100 h. The vertical fracture allows the saltwater to penetrate into the upper part of the domain. The effect of the horizontal fracture can be mainly seen near the vertical one and is there the same as in the 2d tests in section 5.6.1.

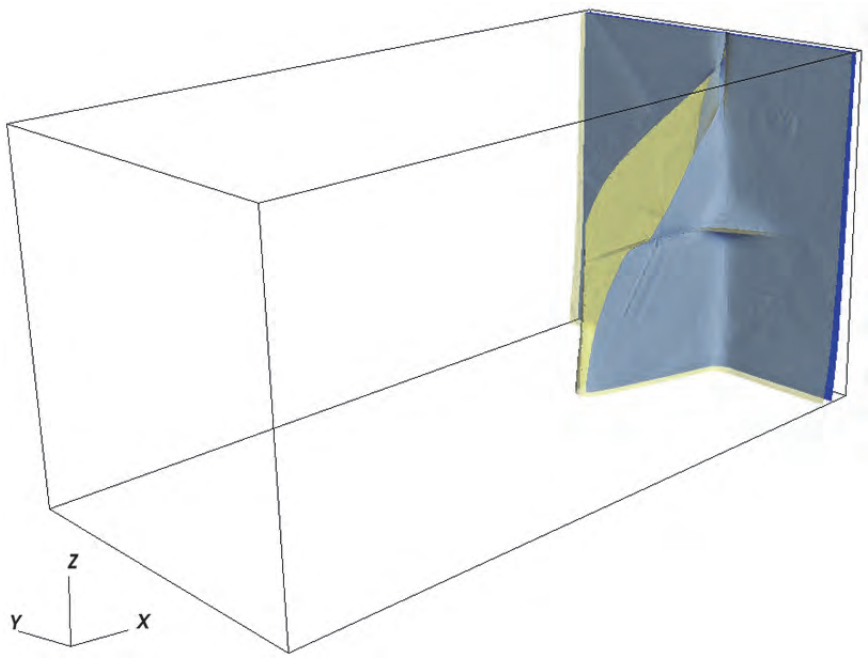


Fig. 5.8 Isolines for c=0.1 (yellow, transparent) and c=0.3 (blue, solid) at time 100 min. in the 3d test

In Fig. 5.7, every boundary surfaces is painted with its own color. Note that the fractures intersect the boundary. In particular, the intersection of the horizontal fracture with the boundary is completely embedded in the “rear wall” and does not cut it completely. For this, the “rear wall” has been split into 2 “bulk” surfaces. The intersections of the fractures with the “rear wall” are subdivided into 4 further (degenerated) surfaces. Thus, the “rear wall” consists of 6 surfaces, and the Dirichlet boundary conditions for c and p must be specified on all them.

6 Modelling of free groundwater table

In many applications of the groundwater flow the position of water table, i. e., the top surface that separates the fully and partially saturated zone in porous media, is unknown in advance and has to be determined by the solution of corresponding mathematical model.

In this project, to enable such applications with d^{3}f, several novel algorithms have been developed and implemented. The basic idea behind the algorithms is to use an implicit representation of the groundwater table by a so called level set function. In such a way one can use a fixed (enlarged) computational domain that encloses the groundwater flow with dynamic top surface and a fixed computational grid with no complex construction of moving meshes having the grid nodes fitting the dynamic position of groundwater table. Furthermore, the movement of groundwater surface can be found by the solution of advection equation for which well-established numerical methods exist.

The advantages of level set methods are accompanied by some additional requirements during operation that are not necessarily trivial. The implementation in d^{3}f involves extrapolation algorithms for the computation of missing groundwater flow velocities above the top surface and the computations of signed distance function that is required to obtain the normal directions with respect to the top surface of groundwater flow. A few parameters have to be chosen to control the behaviour of the new numerical methods.

6.1 Level set function

In this section the level set formulation of an implicit description for the position of groundwater table will be given.

6.1.1 Mathematical description

As mentioned before, the position of the groundwater table (i. e., the top surface of water in porous media) will be described in an implicit way as the zero level set of some function, the level set function.

The groundwater table is considered only as some fixed domain D and in the time interval $(0,T)$. If one denotes by $\Gamma(t)$ the time dependant position of the considered groundwater table, the set of all points lying on $\Gamma(t)$ will be defined as the zero set of **level set function** $\phi(x,t)$, i. e.,

$$\Gamma(t) = \{\phi(x,t) = 0\,,\ x \in D\,,\ t \in (0,T)\}\,. \tag{6.1}$$

It is supposed that $\Gamma(t)$ is a single non-intersecting curve (or surface in 3D) that begins and ends at the boundary ∂D of the domain D. In such a way, it divides the computational domain D into two parts, the part below and the part above $\Gamma(t)$. Of course, the groundwater flow is considered only in the part below $\Gamma(t)$ and no flow is computed in the part above $\Gamma(t)$. The time dependent part of D where the flow is considered will be denoted by $\Omega(t)$ and the rest by $\Omega^{out}(t) := D \setminus \Omega(t)$.

Of course, $\Gamma(t) \subset \partial\Omega(t)$. It is supposed that the rest of boundary of Ω, i. e. $\partial\Omega \setminus \Gamma(t)$, is fixed and can be treated in a standard way, e. g., some Dirichlet and/or Neumann boundary condition can be prescribed. See Fig. 6.1 for illustration of the introduced notations.

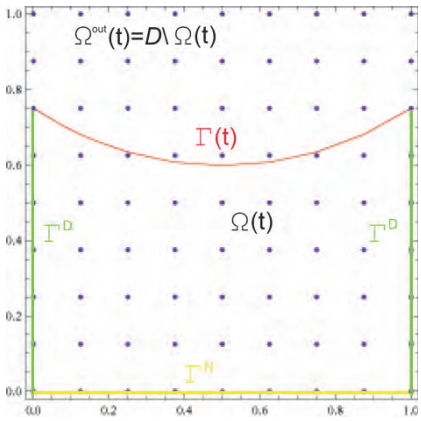


Fig. 6.1 Illustration of the notation. The dynamic groundwater table is denoted by $\Gamma(t)$, the groundwater flow is considered only in $\Omega(t)$

To distinguish easily for each $x \in D$ to which part it belongs, it will be required that $\phi(x,t) < 0$ if $x \in \Omega(t)$ and, analogously, $\phi(x,t) > 0$ if $x \in \Omega^{out}(t)$. In the following this is referred to as that the level set function fulfils a **sign property**.

The initial position $\Gamma(0)$ must be defined by the user providing some appropriate choice of corresponding **initial level set function** $\phi^0(x) = \phi(x,0)$. Such initial position of the

groundwater table is either given by the desired engineering application that has to be simulated numerically, or if some unknown static position of groundwater table is searched, it represents some appropriate initial guess that has to be suggested during the model set-up.

If a trivial shape of initial position is considered (e. g., a horizontal surface), the choice of the level set function can be defined using some prepared functions in d^{3}f. If the offer of prepared initial level set functions does not fit the considered application, a new function must be implemented for which three rules must be fulfilled:

1. The zero set (i. e., the set of all points where the value of level set function equals zero) must be identical with the initial position of groundwater table.
2. The level set function must be non-positive in the part of domain where the groundwater flow takes place and positive in the part where no flow is considered.
3. The gradient of level set function shall be nonzero with the most preferable choice $|\nabla\phi^0| \approx 1|$.

Of course, for any initial position $\Gamma(0)$ there exist infinitely many level set functions $\phi^0(x)$ for which $\Gamma(0) = \{x \in D, \phi^0(x) = 0\}$ together with the required sign property of $\phi^0(x)$ for values $x \notin \Gamma(0)$.

For several reasons to be discussed later one unique choice is preferable, the so called **signed distance function** that will be denoted by $\Phi^0(x)$. In fact, the signed distance function $\Phi(x,t)$ will be computed in this approach also for $t > 0$, therefore $\Phi^0(x) = \Phi(x,0)$.

The function Φ^0 describes the (minimal) distance of any point $x \in D$ to the curve $\Gamma(0)$, i. e.

$$\Phi^0(x) = \min_{y \in \Gamma(0)} |x - y| \,. \tag{6.2}$$

The notation $|x|$ denotes the Euclidian norm of the vector x, i. e.

$$|x| = \left(\sum_{i=1}^{d} x_i^2 \right)^{\frac{1}{2}} \,. \tag{6.3}$$

The signed distance function Φ^0 is always a continuous function, but not necessary a smooth one, because the gradient $\nabla\Phi^0(x)$ needs not to be defined for some $x \in D$. Nevertheless, this function can be found by solving the so called eikonal equation

$$|\nabla\Phi^0(x)| = 1\,,\ x \in D\,, \quad \Phi^0(x) = 0\,,\ x \in \Gamma(0)\,, \tag{6.4}$$

whereas its solution must be searched in a weak form /SET 99/, see some description later in this report.

Any level set function $\phi^0(x)$ associated with $\Gamma(0)$ that fulfills the sign property (and for which the (non-zero) gradient $\nabla\phi^0(x)$ is available) describes uniquely the normal vectors $\vec{N}^0(x)$ with respect to $\Gamma(0)$ at $x \in \Gamma(0)$ by

$$\vec{N}^0(x) = \frac{\nabla\phi^0(x)}{|\nabla\phi^0(x)|}, \tag{6.5}$$

that points outwards with respect to $\Omega(t)$.

If $\vec{N}^0(x)$ in (6.5) is evaluated for some $\hat{x} \notin \Gamma(0)$, it is a normal vector for some curve of which all points $x \in D$ are given implicitly by $\phi^0(x) = \phi^0(\hat{x})$. Again, the normal vectors will play an important role in the time dependent computations, therefore

$$\vec{N} = \vec{N}(x,t) = \frac{\nabla\Phi(x,t)}{|\nabla\Phi(x,t)|}\,. \tag{6.6}$$

6.1.2 Numerical implementation

The level set function is represented in d^{3}f in the standard way of a grid function like the pressure or concentration that will be described now in more details.

Let $x_i \in D$ for $i = 1, 2, \ldots, I$ be grid nodes of some triangulation for the domain D. Let the discretization of time interval for which the numerical simulation will be realized have the form $t^0 < t^1 < \ldots < t^n < \ldots t^N$. The elements of the triangulation will be denoted by $T^e \subset D$ for $e = 1, 2, \ldots, E$.

The **finite element interpolation (shape) functions**, as used by finite element methods, will be denoted by $\Psi_i = \Psi_i(x)$ and the following standard properties will be exploited: Ψ_i are continuous functions,

$$\Psi_i(x_i) = 1\,,\ \Psi_i(x_j) = 0\,,\ j \neq i\,, \tag{6.7}$$

and the gradient $\nabla\Psi_i(x)$, $x \in T^e$ is well defined for each element of the triangulation.

If the initial level set function ϕ^0 is given analytically, either using predefined functions in d³f or the one implemented by user, this function is represented by its numerical approximation $\hat{\phi}^0(x)$ is obtained using the nodal values

$$\phi_i^0 :\approx \phi^0(x_i) \tag{6.8}$$

and the interpolation

$$\hat{\phi}^0(x) = \sum_{i=1}^{I} \phi_i^0 \Psi_i(x)\,. \tag{6.9}$$

Consequently, the groundwater table $\Gamma(0)$ is approximated by $\hat{\Gamma}(0)$ that is given as the zero set of $\hat{\phi}^0$. Therefore, the shape of such approximation is depending on the type of finite element interpolation functions. If triangles T^e (or tetrahedra in 3D) are used, the finite element interpolation functions Ψ_i are piecewise linear, and, consequently, the groundwater table is approximated by a polygonal curve (in 2D case). Analogously, the gradient $\nabla\phi^0(x)$ is approximated by $\nabla\hat{\phi}^0(x)$, but this approximation is not uniquely defined on boundaries of elements T^e, i. e., on edges, respectively sides of T^e.

As examples, two level set functions that are available in d³f are presented here.

The first example is two- or three-dimensional linear level set function of the form

$$\phi^0(x_1, x_2, x_3) = c_0 + c_1 x_1 + c_2 x_2 + c_3 x_3\,. \tag{6.10}$$

It is strongly recommended that for any application the linear level set function is chosen such that the norm of its (constant) gradient is equal one, that means $\sqrt{c_1^2 + c_2^2 + c_3^2} = 1$. In such a case it defines the signed distance function. Note that the

level set function in (6.10) can be represented exactly by the finite element interpolation (6.8) and (6.9) if at least piecewise linear shape functions are used.

Once the function ϕ^0 is defined and initialized, it can be plotted using standard UG commands, see Figure below for an illustration.

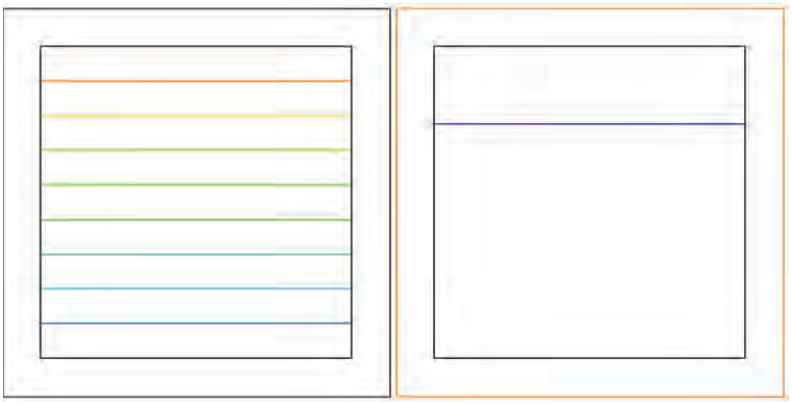


Fig. 6.2 Several contour lines (level sets) of the initial level set function (left) and zero contour line that defines the initial position of horizontal groundwater table (right)

The second example is two- or three-dimensional level set function describing a part of circular interface:

$$\phi^0(x_1, x_2, x_3) = \sqrt{(x_1 - c_1)^2 + (x_1 - c_2)^2 + (x_3 - c_3)^2} - r, \tag{6.11}$$

where r denotes the radius and $c_i\,, i = 1, 2, 3$ the coordinates of the center for the circle (or sphere in 3D case). Note that (6.11) defines always the signed distance function. Concerning the finite element approximation of this function, it cannot be represented exactly and the quality of the approximation $\hat{\Gamma}(0)$ is depending on the mesh.

6.2 Computation of groundwater flow for a fixed groundwater table

Firstly the extension of d^{3}f is for groundwater flow of which the density is constant, e. g., the potential flow, will be described. In that case the equation for the unknown pressure is given by /BEA 88/, /FEI 99/

$$\vec{q} = -\frac{K}{\mu}\left(\nabla p - \rho\vec{g}\right) \tag{6.12}$$

and

$$\nabla \cdot \vec{q} = 0 \quad \Rightarrow \quad -\nabla \cdot K \nabla p = 0\,, \tag{6.13}$$

where ρ and μ are constant properties of the groundwater (the density and viscosity, respectively), and K is the permeability that can be in general space dependant and anisotropic (i. e. represented by a matrix). Some standard boundary conditions must accompany the partial differential equation (6.13).

The problem (6.13) is stationary, the only time dependency can be introduced by some time dependent computational domain $\Omega(t)$ as it was discussed in section 6.1. This is motivated by arguments that the groundwater flow is immediately equilibrated with the actual position of the groundwater table.

The movement of the groundwater table in time will be described in section 6.3. Here the case of a fixed time t, e. g. $t = 0$, is considered, for which the position of groundwater table $\Gamma(t)$ is known. So the task is to determine the groundwater flow described by (6.12) and (6.13) for a domain given by the part of D below $\Gamma(t)$, i. e. $\Omega(t)$.

For a fixed time one has to treat a standard model of groundwater flow. Some extension is necessary only for used numerical methods as described later. The boundary conditions on $\Gamma(t)$ are always the prescribed zero value for the pressure, i. e. the zero Dirichlet boundary condition.

Without going into details, the general model of density driven flow in d^{3}f can be formally used with moving groundwater table, but at the current stage of research and development, the density shall be constant in the neighbourhood of the whole top groundwater table $\Gamma(t)$.

6.2.1 Numerical implementation

The position of the approximated top surface $\hat{\Gamma}(t)$, that means the groundwater table, is described by some level set function $\hat{\phi}^n$, e. g. by the initial level set for $n = 0$. The idea of level set formulation is to use the standard computations of groundwater flow as usual in d^{3}f with no modification of the triangulation of the domain D.

For a fixed position of groundwater table, the computational domain is considered only in the part $\Omega(t)$ of D below the fixed groundwater table. Consequently, the part of computational domain $\Omega^{out}(t)$ above the top surface is not included to simulations, it means no pressure (and velocity) is computed for grid nodes where the (fixed) level set function is positive. Therefore, the standard partial differential equation (6.13) for the groundwater flow is computed at the fixed time t^n only in the grid nodes x_i for which $\phi_i^n \leq 0$.

The boundary condition $p = 0$ on $\Gamma(t)$ at any fixed time t is set directly by numerical discretization and cannot be changed.

The main numerical tools how to extend the finite volume discretization of d^{3}f for the case of implicitly given computational domain is the so called immersed interface formulation. It is heavily based on linear interpolation and extrapolation what will be explained next.

6.2.2 Immersed interface formulation

The standard finite volume discretization as used in d^{3}f and described in /FRO 98b/ can be realized only with elements T^e that lies completely below $\hat{\Gamma}(t)$. Nevertheless, the discretization method must be applied also for elements T^e for which at least one corner, say x_i, lies below $\hat{\Gamma}(t)$, i. e. $\phi_i^n < 0$, and, at the same time at least one corner, say x_j, lies above it. This is not possible without doing some additional work before, because the value of pressure in x_j, say p_j, is not available as it is not included to the unknown values of the pressure to be determined.

To derive a meaningful value of p_j that can be used for the derivation of *i*-th discrete equation, the following procedure is used for arbitrary type of element T^e. This procedure is inspired by the so called immersed interface methods /LIZ 06/.

As mentioned in the previous paragraph, the above situation is characterized by the property $\phi_i^n < 0$ and $\phi_j^n > 0$. Supposing a linear interpolation between these two values, a unique point $\hat{x}_{ij}$ exists for which this interpolation evaluates to zero. The coordinates of such a point can be easily defined by

$$\hat{x}_{ij} = (1 - \alpha_{ij}^n)x_i + \alpha_{ij}^n x_j, \tag{6.14}$$

where

$$\alpha_{ij}^n = \frac{\phi_i^n}{\phi_i^n - \phi_j^n} \,. \tag{6.15}$$

Clearly, $x_{ij}^n \in \hat{\Gamma}(t^n)$, therefore, the numerical pressure can be supposed to be zero at this point. Consequently, the value p_j can be extrapolated along the line starting at x_i, crossing x_{ij}^n and ending at x_j, namely,

$$p_j = \frac{\alpha_{ij}^n - 1}{\alpha_{ij}^n} p_i \,. \tag{6.16}$$

Although $\alpha_{ij} > 0$ is valid in theory, this parameter can reach very small values and the computations of p_j using (6.16) can become unstable. Therefore, some constant parameter ϵ is defined and if $\phi_i^n < \epsilon$, the point x_i is considered to lie on $\hat{\Gamma}(t)$, and the value of numerical pressure is set $p_i = 0$ using the Dirichlet boundary condition.

The results of a simple example are presented below for a fixed horizontal groundwater table. Note that some outflow boundary conditions are defined at the bottom part of the square domain D to obtain some nontrivial groundwater flow regime in this example.

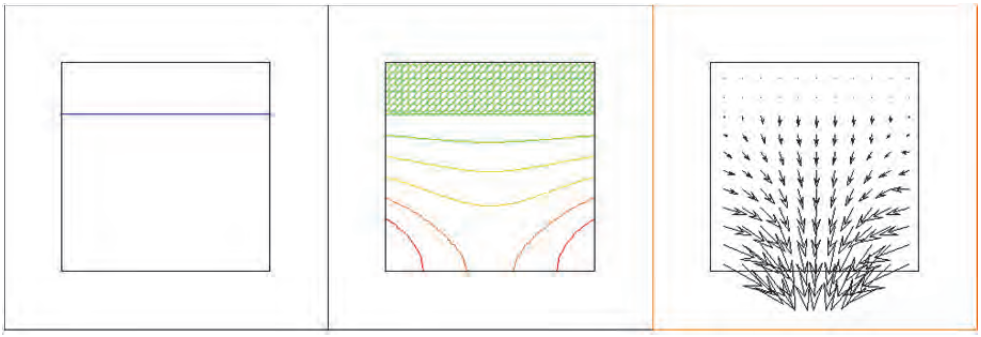


Fig. 6.3 Groundwater table (left), pressure (middle), and groundwater flow (right)

Next the computations of pressure, velocity and concentration in d^{3}f in the case of an implicitly given computational domain are commented. Here the immersed interface formulation has to be used.

6.2.3 The pressure

The pressure is computed only below the groundwater table, and at the exact position of the groundwater table it is equal 0, e. g. Dirichlet boundary conditions are used for the pressure on $\Gamma(t)$. This condition cannot be changed by users. The definition of other boundary conditions on the boundary ∂D of the fixed domain D is done in standard way. Note that for the part of ∂D above $\Gamma(t)$, the definition of boundary conditions is ignored.

If the standard UG graphics is used to plot the pressure (as in Fig. 6.3), the values of the pressure above the groundwater table are formally set to 0.

The UG graphics is in general imprecise for plotting the pressure near the groundwater table, because it can plot only functions that are linear in each element (the triangle in this example). For elements intersected by groundwater table, the pressure is piecewise linear inside of one element, i. e. it has a "knick" in such element.

6.2.4 The groundwater velocity field

The numerical groundwater velocity field can be computed only for grid elements, e. g. triangles, of which at least one corner lies below the groundwater table. That means, the level set function is negative in at least one corner of the element. If all corners lie below $\Gamma(t)$, the standard computation of velocity is used for this element. If at least one corner of the element lies above $\Gamma(t)$, at least one value of the pressure in a corner of the element is missing to compute the gradient of pressure numerically in (6.12) and to compute the velocity in the standard way. To provide the missing pressure value, an extrapolation is used from two values - the pressure below $\Gamma(t)$ and the zero pressure at $\Gamma(t)$.

A new plot procedure was implemented and can be used with UG graphics, as done in Fig. 6.3. This plot procedure sets the zero velocity for all elements of which all corners lie above $\Gamma(t)$, for all other elements the procedure is used as described in the previous paragraph.

The previously described extrapolation procedure of pressure is not unique, therefore one has to decide which particular form has to be used when computing the velocity.

This non-uniqueness can be illustrated for a triangle T^e for which one corner lies above $\Gamma(t)$ and the two remaining ones below it. To extrapolate the missing pressure value for the corner above, one has two choices because of two corners below $\Gamma(t)$ to be used with the extrapolation.

In the plot procedure, the shape functions for each corner below $\Gamma(t)$ are evaluated in the point the velocity is to be computed for. The corner for which the shape function has the largest value is then used to extrapolate the missing pressure value for the corner above $\Gamma(t)$ as described in section 6.2.3. Finally, the standard procedure for the velocity computation can be used.

6.2.5 The concentration

The concentration, as usual in the standard version of d^{3}f, has to be defined by initial conditions for some initial time. Concerning the extension of d^{3}f for the presence of a free groundwater table, the concentration is kept above and at $\Gamma(t)$ at the values given by initial conditions that can be viewed, formally, as Dirichlet boundary conditions for the concentration.

The concentration is constant everywhere in the example of Fig. 6.3, that means, the groundwater density is constant. It is strongly recommended to consider only examples for which the salt concentration at and near the top surface is everywhere (practically) constant, because at the current stage of research the mathematical modelling of groundwater table for variable density flows (especially the unstable situations of more dense flow above less dense one) is not yet finished.

6.3 Computation of the dynamic groundwater table

Before explaining each part of the computation for moving groundwater table in details, it is briefly illustrated, but entirely, for one time step.

The basic idea behind the mathematical modelling of any moving interface is that only the speed of the interface in normal direction is necessary to consider. Any tangential movement along the interface has no influence on the position of the interface itself. The outward-pointing unit normal vector with respect to the groundwater table $\Gamma(t)$ will be denoted by $\vec{N} = \vec{N}(\gamma, t)\,,\ \gamma \in \Gamma(t)$, i. e., this vector is pointing outward of $\Omega(t)$.

Following /BEA 88/, one can prescribe the speed of the groundwater table in normal direction, say S, by

$$S = \frac{1}{n_e}\left(\vec{q} - \vec{A}\right) \cdot \vec{N}, \tag{6.17}$$

where n_e is the so called effective porosity with respect to flow /BEA 88, p. 255/, and $\vec{A} = \vec{A}(\gamma)$ is the so called accretion, i. e., the rate at which the groundwater is added to the aquifer at the groundwater table.

Following /BEA 88/, the effective porosity n_e is very difficult to measure and, in general, can vary in time and space between some initial and final water content. In the case of stationary (equilibrated in time) groundwater table, i. e. when $S = 0$, this parameter is equal to the porosity of porous media. A constant value of n_e can be set in the script file when computing any example with d^{3}f.

Although from a physical point of view, the speed S and the normal vectors $\vec{N}$ are only available at the groundwater table, but the level set formulation requires their definition at the whole domain D. It can be shown /ASL 03/, /FRO 10b/ that an appropriate tool for such extension of S and $\vec{N}$ is the computation of signed distance function Φ and a constant extrapolation of S along characteristics generated by the normalized gradient of Φ, i. e.

$$\vec{N}(x,t) = \frac{\nabla\Phi(x,t)}{|\nabla\Phi(x,t)|}\,. \tag{6.18}$$

These two special and nontrivial procedures were required to be implemented in the extension of d^{3}f.

Once the normal vectors $\vec{N}$ and the speed S are available in the whole domain D at each time $t \geq 0$, one can compute the advection step of the level set function

$$\partial_t\phi(x,t) + S(x,t)\vec{N}(x,t) \cdot \nabla\phi(x,t) = 0\,, \quad \phi(x,0) = \phi^0(x)\,. \tag{6.19}$$

The advection equation (6.19) is basic principle of level set methods. This equation can be solved numerically with popular discretization methods for fixed grids and the

movement of the zero level set of the function ϕ, the solution of (6.19), represents implicitly the movement of the groundwater table.

The three important steps of the level set methods to solve (6.19) are:

- computations of signed distance function and its normalized gradient $\vec{N}(x,t)$,
- computations of the extrapolated speed $S(x,t)$,
- computations of the advection step for level set function $\phi(x,t)$.

Before explaining these steps in detail, typical features of all three steps are illustrated in the following section.

6.3.1 Example of computations

The previous considerations are briefly described by dealing with the first time step of computation as it can be realized with the example mentioned above.

Firstly, the signed distance function, its normalized gradient and the extrapolated speed in the direction of the normalized gradient have to be defined.

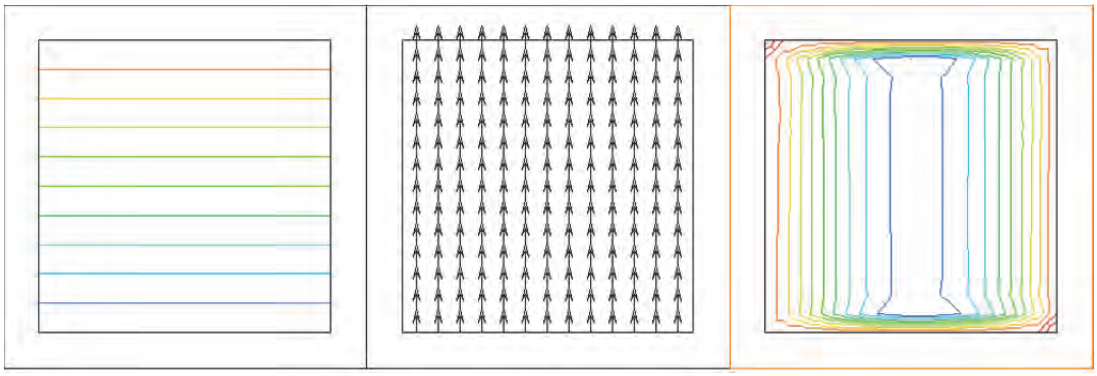


Fig. 6.4 Signed distance function (left), its gradient (middle) and the extrapolated speed (right)

The signed distance function Φ^0 in Fig. 6.4 is identical to the initial level set function ϕ^0 in Fig. 6.2. The gradient of the function is constant in this case.

Clearly, if the initial position of the groundwater table is a straight line, such position can be represented exactly by a linear level set function Φ^0 for which the norm of its gradient is equal one. Such function don't need not to be computed numerically. Of course, the gradient of any linear function is perpendicular to all contour lines of the function.

Therefore, the normal vectors $\vec{N}(x,t)$ at $t=0$ in (6.19) are simply unit vectors (0,1) and these vectors determine the direction in which the level set function $\phi(x,t)$ will be moved by solving the advection equation (6.19). It remains to determine the magnitude (the speed) of this movement.

As described in the previous section, the speed S describing the movement of the groundwater table is defined only locally at the surface, see (6.17). To illustrate this for example $\vec{A}=\vec{0}$ and $n_e=1$ are taken for simplicity.

From Fig. 6.2 one can see that the groundwater flow $\vec{q}$ points inwards to the domain and has its maximal value at the middle of the groundwater table. Therefore, from (6.17) S has its largest negative value at the middle of the groundwater table $\Gamma(0)$, it remains negative at $\Gamma(0)$ and approaches 0 near the boundary ∂D. $S=0$ is set at two points where the groundwater table meets the boundary of D, to avoid the movement of these two end points. In fact $S=0$ is set on the whole boundary ∂D for simplicity, i. e., all level sets (contour lines) of $\phi(x,t)$ will not move at the boundary.

The simplest way how to extend the values of S available only at the horizontal initial groundwater table is to consider a constant value of S along perpendicular lines to the groundwater table. There each constant is determined by the available value of S at the cross-section of the perpendicular line and the horizontal groundwater table. Such extension can be seen on the right picture of Fig. 6.4 where the contour lines of $S(x,t)$ at $t=0$ take a form of vertical lines inside of D. Near the boundary ∂D the function S changes linearly (depending on the underlying grid) to value $S=0$.

In this simple initial situation can easily be defined $\vec{N}(x,t)$ and $S(x,t)$ in (6.19) for $t=0$ without solving any equation. In general, the signed distance function $\Phi(x,t)$ (that gives $\vec{N}(x,t)$ by using (6.18)) is obtained by solving the so called eikonal equation. Similarly, the extrapolated speed $S(x,t)$ can be obtained for the general case by solving some linear advection equation.

Having $\vec{N}(x,0)$ and $S(x,0)$, one can solve (6.19) for $t=0$. This can be done now only numerically as it will be described in section 6.4.3. Nevertheless, from the form of $\vec{N}(x,0)$ and $S(x,0)$ in Fig. 6.4 one can expect that all level sets of $\phi(x,t)$ for $t>0$ (i. e., including the zero one, the groundwater table) will move downwards with the largest movement in the middle part of each level set and with no movement at their end points. Such behavior can be clearly observed in the left picture of Figure bellow, where the new position of groundwater table, the zero level set of $\phi(x,t)$, is plotted after numerical solution of (6.19) for one time step.

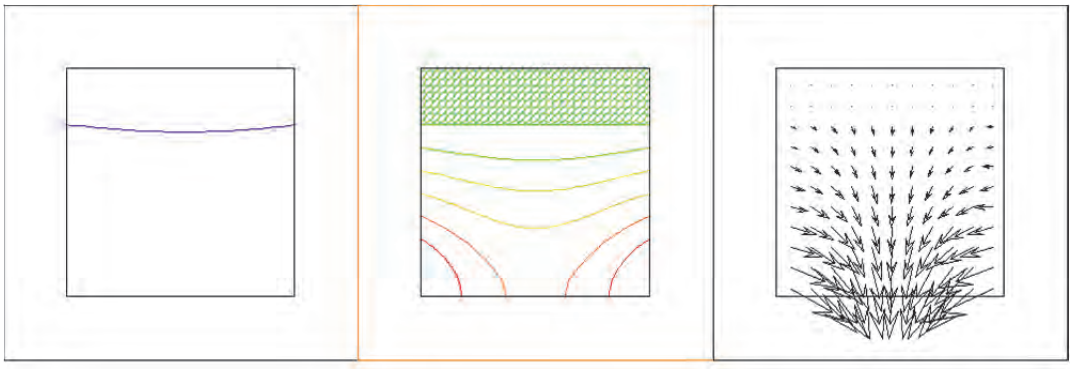


Fig. 6.5 Position of groundwater table after one advection step (left), pressure (middle) and groundwater velocity field for the new position (right)

Having the new position of groundwater table, one can compute for the changed domain $\Omega(t)$ the new groundwater flow situation analogously to the approach as it has been realized for the initial time. The resulted pressure and groundwater flow can be seen in the middle and right picture of Fig. 6.5, respectively.

The next sections will describe how the signed distance function and the extrapolated speed can be updated again with respect to new position of the groundwater table and groundwater flow, see Fig. 6.6.

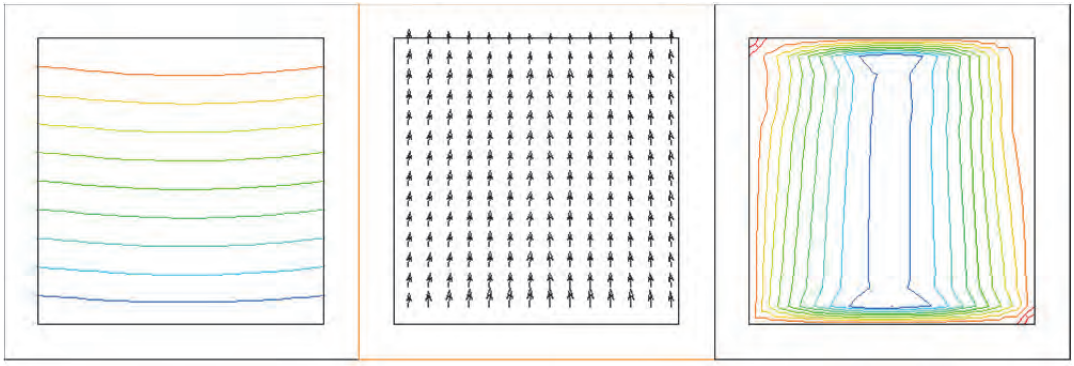


Fig. 6.6 Signed distance function (left), normal directions (middle) and extrapolated speed (right) for the new position.

6.4 Level set method for the dynamic groundwater table

In next sections each of the three procedures mentioned will be described in more detail. As it will be shown, all three procedures are based on a numerical solution of the following general advection equation

$$\partial_t u(x,t) + \vec{V}(x,t) \cdot \nabla u(x,t) = g(x)\,, \tag{6.20}$$

that must be accompanied by some initial and boundary conditions. In (6.20), $u = u(x,t)$ is the unknown function to be determined. The right hand side $g(x)$ is known and given, and the vector field $\vec{V}$ is either given (then the equation represents a linear advection equation) or dependent on the gradient of u when (6.20) takes the form of nonlinear partial differential equation. Clearly, (6.19) is a special case of (6.20) when $u = \phi$, $g = 0$, and $\vec{V} = S\vec{N}$.

Concerning numerical methods for the solution of (6.20), the so called flux-based level set method /FRO 07a/ is implemented in d^{3}f using its high-resolution variant /FRO 07b/, /FRO 09/, and extended for the immersed interface formulation /FRO 10b/. The first order accurate variant of these methods applied to modeling of dynamic groundwater table was published in /FRO 10a/.

6.4.1 Computation of the signed distance function

The purpose of the signed distance function Φ is to give the direction $\vec{N}$ for the movement of the groundwater table, i. e., the advection of ϕ. This direction is computed from the normalized gradient of Φ.

The computation of the signed distance function is definitely the most complicated part of the level set method for the modelling of moving surfaces. Here, only the most important aspects of this topic will be given, for more details one can refer to, e. g., /SET 99/.

Supposed that the time t is fixed, in the following it is skipped in the notation. Let the interface Γ (the groundwater table) be given implicitly as a zero level set of the function $\phi = \phi(x)$, it means $\Gamma = \{x \in D,\ \phi(x) = 0\}$. The signed distance function can be found then as a weak solution /SET 99/ of the following nonlinear stationary partial differential equation with Dirichlet boundary condition

$$|\nabla\Phi(x)| = 1\,, \quad \Phi(x) = 0\,,\ x \in \Gamma\,. \tag{6.21}$$

The equation is called in literature the “eikonal equation” /SET 99/, and it expresses the fact that the norm of the gradient of the signed distance function is equal one.

The notion of weak solution plays an important role for the eikonal equation. For instance, one can see that if some function Φ fulfills (6.21) then the same is true for the function -Φ. Furthermore, one can show that the weak solution of (6.21) is a continuous function, but the gradient $\nabla\Phi$ needs not to be defined in a classical sense for each $x \in D$.

(6.21) can be reformulated in a more convenient form for these purposes. Firstly,

$$|\nabla\Phi| = \frac{\nabla\Phi}{|\nabla\Phi|} \cdot \nabla\Phi\,. \tag{6.22}$$

Next one can introduce a pseudo-time variable s and consider $\Phi = \Phi(x, s)$. The purpose of this extension is to reformulate the solving procedure for the stationary problem (6.21) as a stationary solution of some time dependent advection equation that takes

formally the form (6.20). This can easily be done by considering two following advection equations:

$$\partial_s \Phi(x,s) + \frac{\nabla\Phi(x,s)}{|\nabla\Phi(x,s)|} \cdot \nabla\Phi(x,s) = 1\,, \tag{6.23}$$

and

$$\partial_s \Phi(x,s) - \frac{\nabla\Phi(x,s)}{|\nabla\Phi(x,s)|} \cdot \nabla\Phi(x,s) = -1\,. \tag{6.24}$$

As the initial condition for (6.23) and (6.24) one can use $\Phi(x,0) = \phi(x)$. Moreover, the two equations have the analogous Dirichlet boundary conditions to (6.21) described by $\Phi(x,s) = 0$ for $x \in \Gamma$ and $s \geq 0$.

Clearly, these equations take the form of the general advection equation (6.20) with the vector field depending nonlinearly on $\nabla\Phi$. It can be shown that for some time *s*, say $s = T$, the solution of (6.23), respectively (6.24), becomes stationary and, consequently, the function $\Phi(x,T)$ fulfills (in a weak sense) the eikonal equation (6.21), see also /SET 99/.

The two equations differ in what part of D they are solved. The first one (6.23) is solved for $x \in \Omega^{out} = D \setminus \Omega$, and the second one (6.24) for $x \in \Omega$. Such splitting of the domain D follows from the form of the velocity vector field in (6.23) and (6.24), and the characteristic curves generated by it.

ϕ fulfills the sign property, i. e. $\phi(x) > 0$ for $x \in D \setminus \Omega$ and $\phi(x) < 0$ for $x \in \Omega$. Therefore, in the neighbourhood of Γ for $s = 0$ and $x \in D \setminus \Omega$ the vector field $\vec{V} = \nabla\Phi/|\nabla\Phi|$ is pointing from Γ to the inside of $D \setminus \Omega$. $\vec{V}$ has almost normal direction to the interface. Analogously, $\vec{V} = -\nabla\Phi/|\nabla\Phi|$ for $x \in \Omega$ in the neighbourhood of Γ is pointing from Γ to the inside of Ω. Therefore, for both equations the characteristic curves starting at Γ are following a path to the inside of the corresponding computational domain. So Γ takes a form of an inflow boundary.

Fig. 6.7 illustrates the solution Φ of the eikonal equation and the related vector fields in the case of a circular interface.

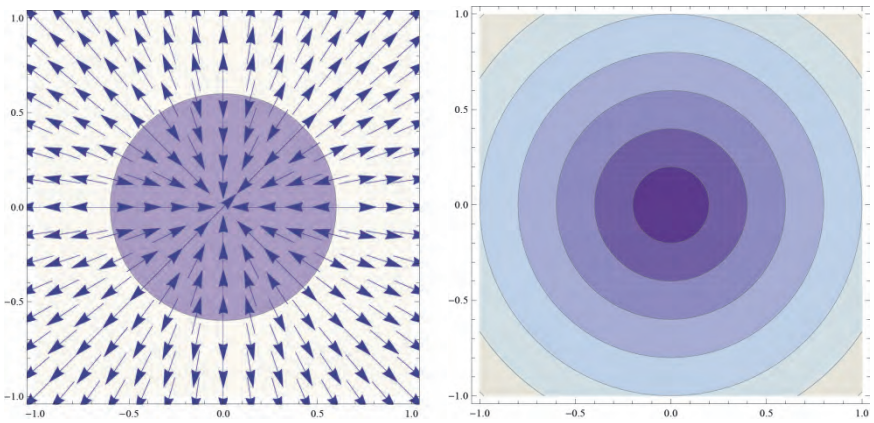


Fig. 6.7 Domain Ω (blue circle) inside of D (square) and two vector fields, the first one $\nabla\Phi/|\nabla\Phi|$ in $D \setminus \Omega$, and the second one $-\nabla\Phi/|\nabla\Phi|$ in Ω, (left) and the corresponding signed distance function Φ (right)

(It is important to mention one important feature of the described procedure to compute the signed distance function that has a significant influence on its numerical computations. Namely, the position of the interface Γ is known only implicitly, so the computational domains Ω and $D \setminus \Omega$ might not be reconstructed explicitly. This situation is numerically treated using again an immersed interface formulation /FRO 10b/.

It is good to discuss here briefly two distinguish roles of functions, ϕ and Φ, that both describe implicitly the position of Γ as its zero level set. In the numerical computations described later only the level set function ϕ will be used to identify the position of Γ, although the signed distance function could be also used for this purpose. The reason f is that in numerical computations the zero contour line of the approximation of Φ and of the approximation of ϕ can differ. This difference is visible only for coarse meshes, nevertheless, to introduce no additional errors to the approximative position of the groundwater table, one shall use the numerical signed distance function Φ strictly only for the approximation of the normal vectors $\vec{N}$.

In the described procedure of this section, the function Φ is obtained as a stationary (time independent) solution of the time dependent equations (6.23) and (6.24). In practice, it is sufficient to compute "almost" stationary solutions, as the equilibrated values of Φ are obtained sooner in the neighborhood of Γ where the appropriate values of Φ are the most important.

The time dependent eikonal equation (6.23) and (6.24) are nonlinear advection equations. The corresponding numerical scheme uses some (pseudo-) timestep that can be

computed automatically depending on the given minimal and maximal Courant number. This number gives some natural restriction on the choice of time steps, and so this choice needs not to be done by users.

Finally, to illustrate the computations of the signed distance function for more complex cases, two additional figures are presented that show results computed with UG procedures used in d^3f.

In Fig. 6.8 the interface takes the form of a square. Note the behaviour of the signed distance function in the neighbourhood of the four corners of the square. A similar behaviour might be observed also in computations of the groundwater table if it takes a piecewise smooth form due to, e. g., a non-smooth permeability field.

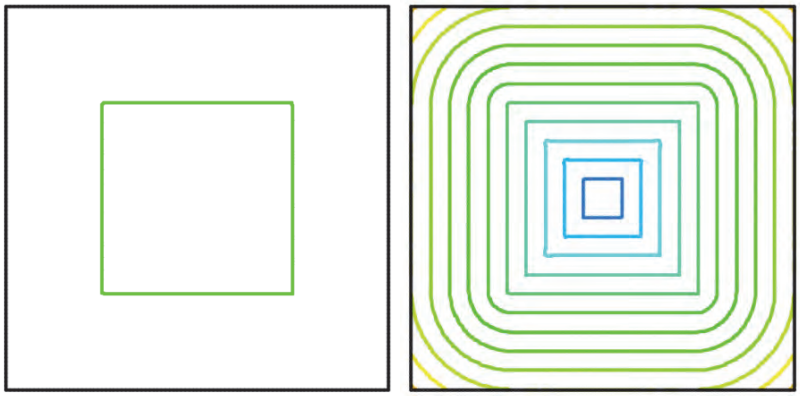


Fig. 6.8 Signed distance function (right picture) for the implicitly given interface in the shape of a square (left picture)

In Fig. 6.9 some smooth, but complex shaped interface is shown. The computed signed distance function is smooth in the neighborhood of the interface, but due to the complicated shape of Γ some corners are developed faraway from Γ, and some contour lines (the level sets) are divided into several pieces.

Fig. 6.9 Resulting signed distance function (right picture) for the implicitly given interface of some complex shape (left picture)

6.4.2 Computation of extrapolated speed

In the following, the time variable t is fixed and skipped from the notation of some functions. Let the signed distance function $\Phi = \Phi(x)$ be given (i. e. computed as described in section 6.4.1) that gives the direction $\vec{N} = \vec{N}(x)$ for $x \in D$ of the movement of the groundwater table according to (6.19). The next task is to determine the speed S in (6.19) using its definition (6.17). Because the groundwater velocity $\vec{q}$ is not known in the part of the domain above the groundwater table, to use (6.17) the speed S will be computed by the extrapolation of known speed at the groundwater table in normal direction $\vec{N}$.

The idea of extrapolation in this case is simple. One has to solve numerically the following particular form of general advection equation (6.20) for $x \in \Omega^{out} = D \setminus \Omega$

$$\partial_s S(x,s) + \vec{N}(x) \cdot \nabla S(x,s) = 0\,, \quad S(x,s) \text{ given at } \Gamma\,. \tag{6.25}$$

Analogously to previous section, the variable s represents a pseudo-time and the equation is solved to stationary case, say at $s = T$, when

$$\vec{N}(x) \cdot \nabla S(x,T) = 0\,, \quad x \in D\,. \tag{6.26}$$

This procedure can be seen as the constant extrapolation of S from Γ along the characteristics generated by the vector field $\vec{N}$ /FRO 10b/. From (6.26) one obtains that the

contour lines (the level sets) of the stationary solution $S(x,T)$ are perpendicular to Γ, see the right pictures in Fig. 6.4 and Fig. 6.6.

Again, the advection equation (6.25) takes a form of the general equation (6.20) with the Dirichlet boundary conditions on the implicitly given boundary Γ. It is solved using high-resolution flux-based level set method /FRO 07a/, /FRO 09/ by exploiting the immersed interface formulation /FRO 10b/.

In the following the speed S will be extrapolated from Γ also to the inside of Ω, although for this part of D the definition (6.17) is applicable. As described in /ADA 99/, such choice of extrapolation for S and the consecutive usage of S in (6.19) has a preferable property for the solution ϕ of (6.19).

Particularly, to extrapolate S from Γ to $x \in \Omega$, one solves

$$\partial_s S(x,s) - \vec{N}(x) \cdot \nabla S(x,s) = 0\,, \quad S(x,s) \text{ given at } \Gamma\,. \tag{6.27}$$

Again, the extrapolated speed S is obtained as a stationary (time independent) solution of the time dependent linear advection equation (6.27), respectively (6.25). In practice, it is often enough to compute "almost" stationary solutions.

In having obtained the stable numerical solution and an optimum grid refinement, physical instabilities were induced by varying the various physical parameters and their effects studied by making the relevant changes in the stability criterion. The results are presented in the following sections. The grid and time stepping in Fig. 6.5 in the case of the circular interface Γ is computed with procedures used in d^{3}f for $\vec{q} = (1,0)$, $n_e = 1$ and $\vec{A} = \vec{0}$. The contour lines of extrapolated speed S takes clearly the form of straight lines that are perpendicular to Γ.

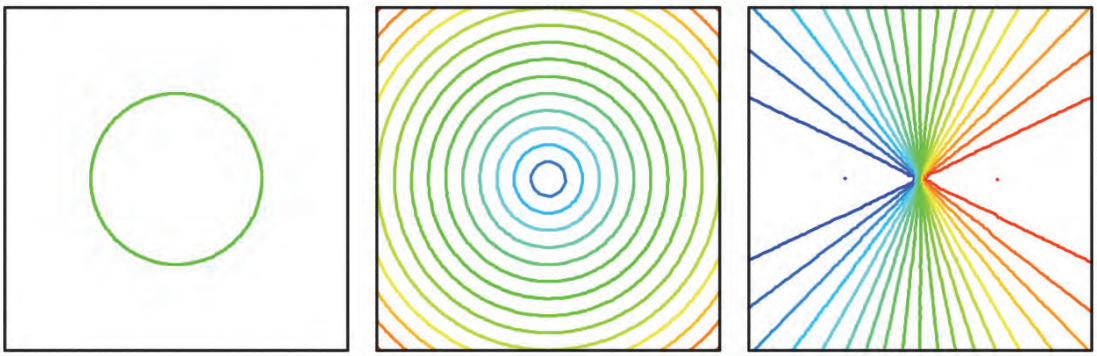


Fig. 6.10 left: circular interface Γ
middle: related signed distance function Φ
right: constant extrapolation along characteristics generated by $\nabla\Phi$

6.4.3 Computation of the advection equation for the level set function

Once the direction and the magnitude (velocity) of the movement of the groundwater table is given and extended to the whole domain, the level set function ϕ can be advected by computing one (physical) timestep of linear advection equation (6.19). In such a way, all contour lines of ϕ are moved with the speed $S\vec{N}$, including its zero contour line (i. e., the groundwater table).

Once the position of the groundwater table is changed, the overall algorithm can be repeated, i. e., the pressure and the groundwater flow can be computed for the new position of the groundwater table, the signed distance function and the extrapolated speed can be adjusted and another advection step for ϕ can be realized.

6.5 Computation of transport in r³t with free groundwater table

Once the groundwater flow is modelled with d³f, often a contaminant transport has to be computed where the advection part of the transport is prescribed by the vector field $\vec{q}$ giving the computed flow regime. This important possibility has been implemented in r³t also for the case of free surface groundwater flow.

To make this possible the data that describe the computed flow have to be transferred between two separate software tools d³f and r³t. For that purpose the numerical velocity field of groundwater flow has to be saved within d³f and afterwards to read in into r³t to characterize the advection part of contaminant transport. To introduce no additional

approximation errors, it is preferable to use the same computational grid as it was used in d^{3}f.

The most typical and relevant computations of flow and transport in porous media are for the case when the groundwater flow is stationary or almost equilibrated in time. The essential assumption of r^{3}t is that the contaminants in the groundwater have negligible or no influence on the flow regime, therefore the transport can be computed independently. From a practical point of view it means that the flow is saved in d^{3}f at some chosen time of interest, and it is supposed to stay constant in time for consecutive computations with r^{3}t.

If the computations with d^{3}f have been realized with moving groundwater table using the level set formulation, the analogous situation shall be considered also in r^{3}t, namely that the transport of contaminant shall occur not in the whole computational domain D, but only in its subset $\Omega(t)$. For that purpose not only the velocity field is saved in d^{3}f, but also the numerical level set function ϕ that describes the position of the top surface for the groundwater implicitly at the chosen time of interest.

In such a way, if approximations of the velocity field and the level set function from d^{3}f are available in r^{3}t, analogously to previously described immersed interface methods, the discretization of r^{3}t is adapted to this case and the transport of contaminant is considered not in the whole domain D, but only in its part $\Omega \subset D$ where the groundwater flow is available.

7 Computation of potential flow

The origin and most important purpose of d^{3}f is to compute density driven flow, that means groundwater flow with variable density. Such physical situation can occur, e. g., in the neighbourhood of a salt dome where a mixture of fresh and saline waters can occur. In most of other practical situations, the density of groundwater can be considered to be constant. Consequently, it is very often desirable to compute the so called potential flow where the groundwater flow is proportional to the pressure gradient or piezometric head. Moreover, for the case of extremely long time simulations of radioactive contaminant transport – the main application of r^{3}t – it is feasible to characterize the groundwater flow in some stationary regime.

In the framework of d^{3}f such situation can be the characterized most conveniently by defining some constant initial salt concentration in D and “no boundary condition” when the default choice of homogeneous Neumann boundary condition, the so called closed boundary, is used in r^{3}t. Consequently, no transport of brine can occur and the groundwater density remains constant at the value determined by the initial brine concentration.

If no time dependence is introduced for the input data characterized the boundary conditions for the pressure and/or the source terms for the flow equation, the resulting equations are in fact stationary with no time dependence.

To compute effectively such situation, the discretization and the corresponding script files have been modified to enable d^{3}f users to compute the stationary case, i. e., the potential flow, directly with no time dependent computations. Even if some time dependence is still considered for the flow with constant density, that means that the transport equation in d^{3}f is trivial to solve and the only partial differential equation to be solved numerically is the flow equation. The discretization as implemented in d^{3}f is effective enough to compute efficiently such special cases.

8 Numerical advances

8.1 Higher order finite volume schemes

Density driven flow in d³f is described by two conservation equations: the flow equation, expressing the conservation of the fluid-phase as a whole, and the equation for the transport of salt, formulating the balance of mass of the brine. These two equations are given by /FEI 99/:

$$\frac{\partial}{\partial t}\left(\phi\rho_f\right) + \nabla\left(\boldsymbol{q}\,\rho_f\right) = s_f, \tag{8.1}$$

$$\frac{\partial}{\partial t}\left(\phi\rho_f\chi_s\right) + \nabla\left(\boldsymbol{q}\,\rho_f\chi_s - \boldsymbol{D}_s\rho_f\nabla\chi_s\right) = s_s, \tag{8.2}$$

where

$$\boldsymbol{q} = -\frac{K}{\mu}\left(\nabla p - \rho_f\boldsymbol{g}\right) \tag{8.3}$$

is the Darcy velocity. An important aspect of these equations is the flux balance that becomes more obvious when the equations are written in integral form:

$$\frac{\partial}{\partial t}\int_B \phi\rho_f\, dV + \int_{\partial B}\boldsymbol{q}\,\rho_f\, dS = \int_B s_f\, dV, \tag{8.4}$$

$$\frac{\partial}{\partial t}\int_B \phi\rho_f\chi_s\, dV + \int_{\partial B}\boldsymbol{q}\,\rho_f\chi_s - \boldsymbol{D}_s\rho_f\nabla\chi_s\, dS = \int_B s_s\, dV. \tag{8.5}$$

Here, the integration volume B is an arbitrary subset of the considered domain Ω and is usually called control volume and each integral equation expresses the fact that changes of quantities in the control volume are due to fluxes over the boundary or sinks/sources within the control volume.

Since the conservation property of these equations arise from the underlying basic physical properties, a discretization scheme should ideally reflect these properties in the numerical scheme. In order to guarantee the discrete conservation property, a finite volume scheme is used in d^{3}f /FRO 96/. A generalization of this type of discretization scheme has been implemented for the equations of d^{3}f.

8.1.1 Definition of a Finite Volume Scheme of Higher Order

The generalization of the vertex-centered finite volume scheme has been proposed in /VOG 10/. It can be applied to conservation equations that are of the form:

$$\frac{\partial}{\partial t}\int_B c\, dV + \int_{\partial B} \boldsymbol{F}(c)\, dS = \int_B f\, dV, \tag{8.6}$$

where c is some unknown solution, **F** is a flux function and f is some source term.

In order to solve such type of equation numerically by a vertex-centered finite volume technique, two choices have to be made: The Ansatz space for the unknown solution has to be chosen, and the set of discrete control volume B, where the conservation law is fulfilled numerically, must be specified.

The usual choice for the Ansatz functions in d^3f is a set of linear trial function, i. e. the unknown solution is represented by linear functions on each element of the mesh discretizing the domain. This approach requires only degrees of freedoms that are locate in the vertices of the mesh. See Fig. 8.1 (left) for an example.

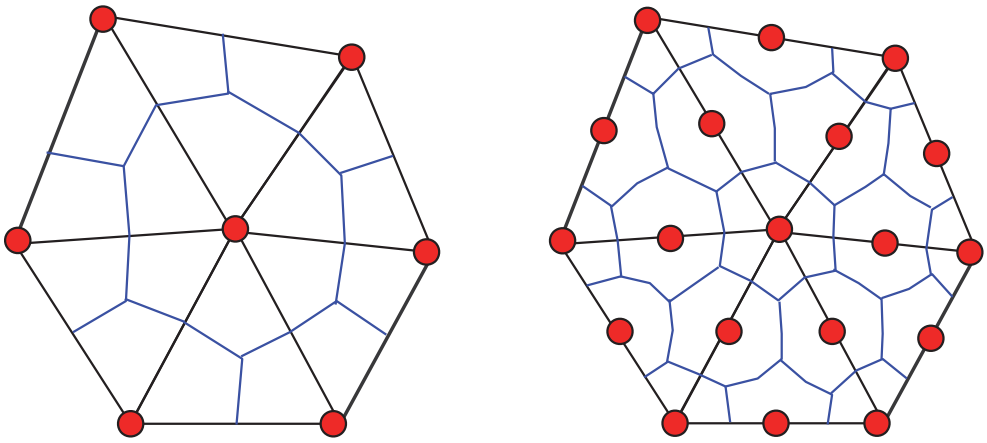


Fig. 8.1 A finite-element mesh (black lines) with the degrees of freedoms (red dots) for linear Ansatz functions (left) and quadratic Ansatz functions (right). The blue lines show the discrete control volumes constructed by the barycentric method

The generalization to higher orders can be achieved by using a polynomial representation for the unknown solution on each element that uses functions of order greater than one. Lagrange finite elements can be used for this purpose and result in degrees of

freedom on vertices, edges, faces and volumes in order to ensure the continuity of the solution. A visualization of the degrees of freedoms for quadratic Ansatz functions is given in Fig. 8.1 (right).

For the set of control volumes several possibilities exist. In the software d^{3}f the barycentric control volumes are used /FRO 96/. These are constructed by taking the convex hull of the following points: a vertex of the mesh, all barycenters of adjacent edges and faces of the vertex and all barycenters of the adjacent elements. An example of such a control volume is shown in Fig. 8.1 (left). The generalization of this construction is related to the chosen Ansatz space. The idea is to construct one control volume for each degree of freedom. This ensures that the resulting linear system remains quadratic. The procedure to construct such control volumes is as follows: For a given element of the mesh, subdivide the element virtually into smaller elements of the same type, such that the finer partition corresponds to a distribution of degrees of freedom equivalent to the linear case, i. e. all virtual subelements carry exclusively degrees of freedom in their vertices. Now, the same barycentric construction procedure as used for the linear case can be applied to the virtually refined elements to produce control volumes related to the subelements. Fig. 8.2 visualizes this procedure for a triangle with quadratic (left) and cubic (rigth) Ansatz spaces. The result of such a refinement is demonstrated in Fig. 8.2 (right).

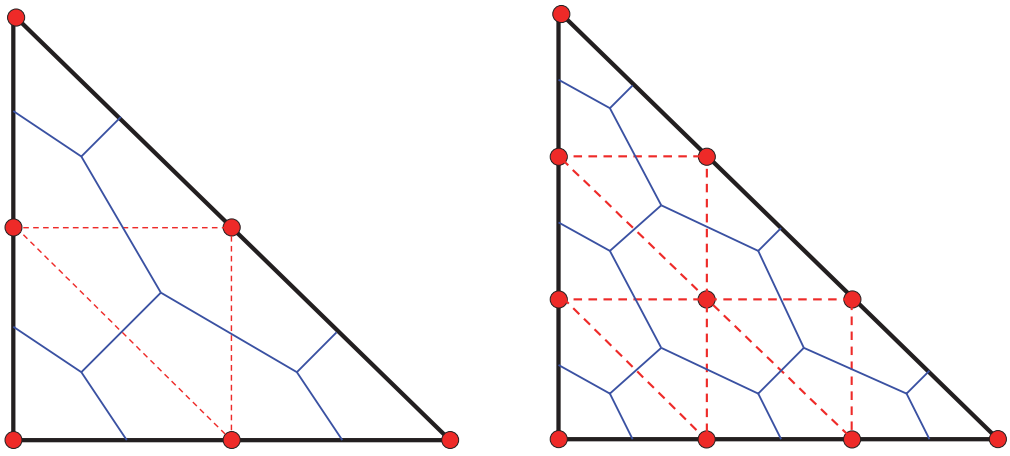


Fig. 8.2 Subdivision of triangles
A single triangle (black lines) is virtually subdivided into smaller triangles (red lines) such that the subtriangles have the same structure as linear elements. Applying the barycentric construction for linear control volumes results in control volumes as indicated by the blue lines

8.1.2 Application to the equation for density driven flow

The finite volume scheme with Ansatz functions of higher order can be used to discretize the equation modelling the density driven flow.

Let $\{\Phi_i(\boldsymbol{x})\}_{i=1,\dots,N}$ be the set of Ansatz functions that are necessary to form a basis of the piecewise polynomial spaces described above, where N is the total number of degrees of freedom. For both unknown solutions χ_s, p the same Ansatz space, formed by these functions, is used. Thus, numerical solutions are searched, that represent the unknown brine mass fraction by

$$\chi_s(\boldsymbol{x}, t) = \sum_{i=1}^{N} \chi_{s_i}(t)\, \Phi_i(\boldsymbol{x}) \tag{8.7}$$

and the unknown pressure by

$$p(\boldsymbol{x}, t) = \sum_{i=1}^{N} p_i(t) \Phi_i(\boldsymbol{x}). \tag{8.8}$$

Now, let $\{B_i\}_{i=1,\dots,N}$ be the set of control volumes that are constructed based on the generalized barycentric control volume approach described above. The discretized form of the equations (8.4), (8.5) is given by:

$$\frac{\partial}{\partial t}\int_{B_i} \phi\rho_f \, dV + \int_{\partial B_i} \boldsymbol{q}\, \rho_f \, dS = \int_{B_i} s_f \, dV\,, \quad \text{for all } i = 1, \dots, N, \tag{8.9}$$

$$\frac{\partial}{\partial t}\int_{B_i} \phi\rho_f\chi_s \, dV + \int_{\partial B_i} \boldsymbol{q}\, \rho_f\chi_s - \boldsymbol{D}_s\rho_f\nabla\chi_s \, dS = \int_{B_i} s_s \, dV, \quad \text{for all } i = 1, \dots, N. \tag{8.10}$$

Here, by χ_s, p the finite dimensional representations from (8.7), (8.8) are used.

8.1.3 Numerical test example

In order to test the implementation using higher order Ansatz functions the scheme is applied to the Henry problem /HEN 64/.

The quality of the solutions, computed using different orders of Ansatz spaces, is compared using a reference solution. This is a numerical solution of the problem at time step *t* = 120s, that is computed on a very fine grid and using very small time steps such that this solution is expected to be close to the true solution. The difference between

the reference solution χ_s^{ref} and some coarser approximation χ_s is computed in H^1-norm, i. e. by

$$\delta := \left(\int_\Omega |\chi_s - \chi_s^{ref}|^2 + |\nabla\chi_s - \nabla\chi_s^{ref}|^2 \, dV \right)^{1/2}. \tag{8.11}$$

The results for the linear, quadratic and cubic Ansatz spaces are shown in Tab. 8.1.

Tab. 8.1 Measurements of the approximation rate for different orders of Ansatz spaces used to discretize the density driven flow equations
FVp (p = 1,2,3) denotes the order of Ansatz function, N the number of degrees of freedom, $\boldsymbol{\delta}$ the difference between the computed solution and the reference solution.

FV1		
N	δ	rate
15	3,23e0	---
45	3,08e0	0,07
153	2,62e0	0,23
561	2,05e0	0,35
2145	1,27e0	0,69
8385	6,85e-1	0,88
33153	3,29e-1	1,06
131841	1,56e-1	1,07

FV2		
N	δ	rate
45	2,58e0	---
153	2,15e0	0,27
561	1,52e0	0,50
2145	7,93e-1	0,94
8385	3,18e-1	1,31
33153	1,01e-1	1,65
131841	2,67e-2	1,92

FV3		
N	δ	rate
91	1,93e0	---
325	1,43e0	0,43
1225	7,79e-1	0,88
4753	2,96e-1	1,40
18721	7,64e-2	1,96
74305	1,30e-2	2,56

A visualization of the decrease of the approximation difference is shown in Fig. 8.3.

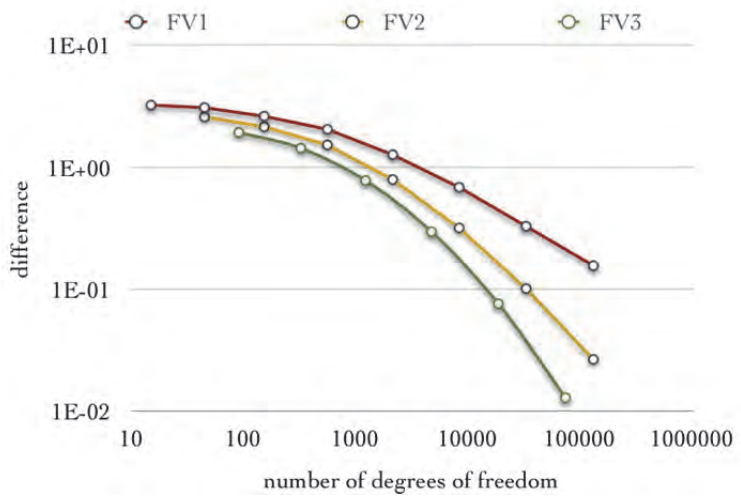


Fig. 8.3 Difference to a reference solution of the Henry problem for the first three orders of Ansatz spaces measures in H^1-norm

8.2 Algebraic multigrid solvers for density driven flow

8.2.1 Introduction

In saline solutions density effects play an important role, which gives rise to the problem of density driven flow. This chapter introduces a novel algebraic multigrid approach for density driven flow. The work is organised as follows: In section 8.2.2 the problem of interest is introduced and the context of the development of the solvers is described. In section 8.2.3 an algebraic multigrid method is introduced. The convergence of this method is proved in section 8.2.4. section 8.2.5 is devoted to the design of different preconditioning strategies. section 8.2.6 concludes this work by providing numerical experiments underlining the capabilities of the method.

8.2.2 Density driven flow

8.2.2.1 Governing equations

The full equations for density driven flow state conservations of the fluid and the salt mass respectively, and are typically formulated in terms of *p*ressure p and *s*alt mass fraction ω, e. g., /BEA 91/, /HOL 98/:

$$\partial_t(\Phi\rho) + \nabla \cdot [\rho\mathbf{q}] = \rho q_V \tag{8.12}$$

$$\partial_t(\Phi\rho\omega) + \nabla \cdot [\rho\omega\mathbf{q} - \rho\mathbf{D}\nabla\omega] = \rho\omega q_V \tag{8.13}$$

with the Darcy velocity

$$\mathbf{q} = -\frac{K}{\mu}(\nabla p - \rho\mathbf{g}) \tag{8.14}$$

and Scheidegger-type tensor

$$\mathbf{D} = \mathbf{D}_{mol} + \mathbf{D}_{mec}(\mathbf{q}) \tag{8.15}$$

Here, the density $\rho = \rho(\omega)$ and viscosity $\mu = \mu(\omega)$ are given by non-linear material laws. The Porosity Φ, permeability K, gravity $\mathbf{g}$, sources q_V are constant or depend on space only.

A simplified and frequently used version of (8.12) and (8.13) is the Boussinesq approximation, cf. /NAE 08/, /BOU 03/. In this case the dependence of $\rho = \rho(\omega)$ on ω is only considered for the velocity $\mathbf{q}$:

$$\nabla \cdot [\mathbf{q}] = q_V \tag{8.16}$$

$$\partial_t(\Phi\omega) + \nabla \cdot [\omega\mathbf{q} - \mathbf{D}\nabla\omega] = \omega q_V \tag{8.17}$$

Although this formulation is known to be less accurate for certain problems, /JOH 03/, it is considered in subsection 8.2.2.3 from a theoretical point of view.

8.2.2.2 Solution strategies

The fully non-linear system stated in (8.12) and (8.13) is solved using the software package d^{3}f /FEI 99/. Time is discretized implicitly. The non-linear system resulting in each time step is linearized using a Gauß-Newton method. A line search strategy is employed for a globalisation of the method. In the innermost loop, it is required to solve a sequence of linear system of equations, denoted by

$$J_h \mathbf{u}_h = \mathbf{f}_h. \tag{8.18}$$

Here, $\mathbf{f}_h$ is the nonlinear defect of the current iteration, and J_h is the Jacobian. The vector $\mathbf{u}_h$ is the resulting correction and search direction respectively.

8.2.2.3 Model problem

In order to study this in greater detail, the following model problem is stated: When (8.16) and (8.17) is linearized in some point $(p_0, \boldsymbol{\omega}_0)^T$ one obtains

$$\nabla \cdot \left[-\tfrac{K}{\mu}\nabla\overline{p} + \mathbf{q}_{0'}\overline{\omega}\right] = d_p \tag{8.19}$$

$$\partial_t(\Phi\overline{\omega}) + \nabla \cdot \left[-\omega_0 \tfrac{K}{\mu}\nabla\overline{p} + (\omega_0\mathbf{q}_{0'} + \mathbf{q}_0)\overline{\omega} - \mathbf{D}_0\nabla\overline{\omega}\right] = d_\omega \tag{8.20}$$

Here $(\overline{p}, \overline{\omega})^T$ is the search direction and

$$\mathbf{q}_0 := -\tfrac{K}{\mu}(\nabla p_0 - \rho_0\mathbf{g}), \quad \mathbf{q}_{0'} := \tfrac{K}{\mu}\rho'_0\mathbf{g}$$

are the Darcy velocity and its derivative w.r.t ω in the linearization point respectively. For the sake of simplicity, derivatives of the dispersion tensor $\mathbf{D}_0$ and the viscosity μ have been neglected. Rewriting (8.18) component-wise one obtains

$$J^{pp}\mathbf{u}^p + J^{p\omega}\mathbf{u}^\omega = \mathbf{f}^p \tag{8.21}$$

$$J^{\omega p}\mathbf{u}^p + J^{\omega\omega}\mathbf{u}^\omega = \mathbf{f}^\omega \tag{8.22}$$

From (8.19) and (8.20) the following facts are deduced for this system: Firstly, the problem is elliptic w. r. t. p and the parabolic w. r. t. ω. Secondly, if $\omega_0 = \text{const}$, the variables decouple, as the $J^{\omega p}$ block from (8.20) may be eliminated by means of (8.19). In this case, one can first solve for ω and then, in a next step for p. Although this assumption is unrealistic, it may be satisfied in parts of the computational domain. In subsection 8.2.5.2 will be come back to this observation.

8.2.3 Algebraic Multigrid

The time for the solution process is to a great extent often governed by the computational time required for the solution of the linearized system (8.18). As multigrid methods feature an optimal computational complexity for a variety of problem, they provide an attractive solver and have frequently been used in this context. These classical geometric approaches may however be limited: In some cases geometric methods are not robust (or may even fail) due to the choice of parameters and/or geometric anisotropies. In other cases, the mesh provided initially as a coarse grid may have a substantial size already. This is the case, e. g., when fractures or known singularities are resolved on the coarsest mesh.

In these situations the class of algebraic multigrid (AMG) methods provides an attractive alternative. This does not rely on a grid hierarchy, which is provided initially. The key idea is instead to extract all information from the linear system to solve, e. g., from the matrix graph and the matrix entries.

One of the most popular, not to say classic, version of this class has been described in /RUG 87/, /STU 01/. Other approaches include smoothed aggregation, /VAN 96/, /MAN 99/, /VAN 01/, or AMG based on element interpolation, /BRE 01/. This work focuses on **Filtering algebraic multigrid method** (FAMG), previously described in

/WAG 00/ and /NAE 08/. The essential results are however likely to be applicable to different AMG approaches as well.

After fixing notation in section 8.2.3.1 and 8.2.3.2, the key idea of the Filtering AMG method is described in section 8.2.3.3. In subsection 8.2.3.4 there will be described the process of coarse grid selection. The extension of the method to systems of PDEs is finally described in subsection 8.2.3.5.

8.2.3.1 Preliminaries

For reasons of flexibility, the system to solve is denoted by

$$A_h \mathbf{u}_h = \mathbf{f}_h. \tag{8.23}$$

This can either be the original system (8.18), or a transformed system which will be introduced later in Section 8.2.3.5. Alternatively, one can also consider a sub-block, e. g.,

$$A_h = J_h^{pp}.$$

Instead of solving a single discrete problem of type (8.23) multigrid methods employ a hierarchy of linear systems of equations

$$A_k \mathbf{u}^k = \mathbf{f}^k, \quad k = 1 \ldots L. \tag{8.24}$$

Degrees of freedom or variables are represented by index sets V^k, where k indicates the grid level. The number of variables is assumed to be decreasing, $|V^k| \geq |V^{k+1}|$ for all $k \geq 1$. The vectors $\mathbf{u}^k, \mathbf{f}^k$ correspond to grid functions and should be identified with $\mathbb{R}^{V^k}$, $k \geq 1$.

The transfer of grid functions between different grids is defined by the following transfer operator:

While interpolation

$$P_{k+1}^k : \mathbb{R}^{V^{k+1}} \to \mathbb{R}^{V^k}, \quad \mathbf{u}^{k+1} \mapsto \mathbf{u}^k, \tag{8.25}$$

maps a grid function from a coarse grid to a fine grid, the restriction

$$R_k^{k+1}: \mathbb{R}^{V^k} \to \mathbb{R}^{V^{k+1}}, \quad \mathbf{f}^k \mapsto \mathbf{f}^{k+1}. \tag{8.26}$$

operates in the opposite direction. Throughout this work, it is assumed that all transfer have full rank.

8.2.3.2 Setup phase

Algebraic multigrid methods construct the grid hierarchy (9) recursively in a separate setup phase, which precedes the actual solution phase. This is done recursively, starting on the finest grid. It is often convenient to consider a two-grid method only. In this case, $h \equiv k$ and $H \equiv k+1$ are written. Given a matrix A_h and an associated index set V^h three tasks must be fulfilled in order to construct a coarse grid operator A^H and an index set V^H:

1. Selecting the coarse grid variables V^H.
This is usually obtained from C/F-splitting

$$V^h = C \dot{\cup} F \quad \Rightarrow \quad V^H := C,$$

which splits the variables V^h into a fine set F and a coarse set C. The later one is then used to define the grid on the next level.

2. Constructing interpolation P_H^h, R_h^H.
Interpolation is defined as a matrix dependent, local operator. Thus the sparsity pattern of P_H^h should be fixed. A fine grid variable $i \in F$ is in general interpolated from a set of parent nodes $E_i \subset C$:

$$\left(P_H^h \mathbf{u}_H\right)_i = \sum_{j \in \mathrm{E_i}} w_{ij} u_j \ , \ \forall \mathbf{u}_H = \left(u_j\right) \in \mathbb{R}^{V^H}. \tag{8.27}$$

An exact interpolation of coarse grid variables $i \in C$ can be used to guarantee that these operators have full rank. After reordering the variables, $u_h = \left(u_{F,h}^T, u_{C,h}^T\right)^T$, the matrix can be represented as $P_H^h = (W_{FC}^T, I)^T$ and there can be written

$$u_h = \begin{pmatrix} u_{F,h} \\ u_{C,h} \end{pmatrix} = \begin{pmatrix} W_{FC} u_H \\ u_H \end{pmatrix} = P_H^h u_H \ .$$

3. Constructing a coarse grid operator A_H.

This is either by a Petrov-Galerkin ansatz $A_H := R_h^H A_h P_H^h$, or, for $R_h^H = {P_H^h}^T$, by the Galerkin ansatz $A_H := {P_H^h}^T A_h P_H^h$.

The whole process is iterated and stopped, if $|V^H|$ is sufficiently small or coarsening is stopped as $|V^H| \approx |V^h|$. The resulting coarse grid problems are typically solved by a direct smoother.

As will be described in Subsections 8.2.3.4 and 8.2.3.5, the first two tasks are solved at once by an iterative selection of a fine grid: A node $i \in V^h$ has the potential to become a fine grid node, if it can be interpolated 'well enough' from (surrounding) coarse grid nodes. For the time being, the symmetric version of FAMG is described and $R_k^{k+1} := (P_{k+1}^k)^T$ chosen. Although more general constructions are possible, they have not proven to be successful. The goal is to preserve the elliptic character for the pressure.

8.2.3.3 Filtering Algebraic Multigrid

The key idea is to minimise the norm of the coarse grid operator approximately, while an additional constraint, the so called filter condition, guarantees that certain vectors are interpolated exactly. An interpolation operator is to be constructed as the solution of the following minimisation problem:

$$\min_P \left|\left|\left|X_p (I - PR') {S'}_p Y_p^{-1}\right|\right|\right|_2^2, \tag{8.28}$$

$$s.t. (I - PR')\tilde{\mathbf{t}}_p = 0. \tag{8.29}$$

For the sake of simplicity, all grid related subscripts have been dropped. The triple bar norm $\left|\left|\left|\cdot\right|\right|\right|_2$ refers to the Frobenius norm of a matrix, $\left|\left|\left|X\right|\right|\right|_2^2 := \sum_{i,j} x_{ij}^2$ for all $X = (x_{ij})_{i=1,j=1}^{m,n} \in \mathbb{R}^{m \times n}$. Moreover ${S'}_p$ is a smoother, or an approximation to a smoother, $P = (W_{FC}^T, I)^T$ is the interpolation and $R' = (0, I)$ is the injection to the coarse grid. The vector $\tilde{\mathbf{t}}_p$ is a representation of the near null space of the operator. The matrices X_p and Y_p serve as scaling matrices.

Let the vector $\mathbf{q}_i^p$ denote the i-th row of $(I - PR')$, i. e. ,

$$(\mathbf{q}_i^p)^T := \varepsilon_i^T (I - PR'),$$

where ε_i is the canonical unit vector for $i \in V$. According to (11) the following representation is found:

$$\langle \mathbf{q}_i^p, \varepsilon_k \rangle = \begin{cases} 1, & i = k, \\ -w_{ik}^p, & i \in E_i, \\ 0, & k \notin E_i, \end{cases} \tag{8.30}$$

for $i \in F$ and $q_i^p = 0$ for $i \in C$ respectively.

Given a (possible) fine grid node i, which is interpolated from its parent E_i using weights defined by q_i^p from (8.30), the following minimisation problems must be solved:

$$\min_{q_i^p} \left(x_{ii}^p \parallel (S'_p Y_p^{-1})^T q_i^p \parallel_2 \right)^2, \tag{8.31}$$

$$s.t. \langle q_i^p, \tilde{\mathbf{t}}_p \rangle = 0. \tag{8.32}$$

for all $i \in F$, which is equivalent to (8.28) and (8.29).

8.2.3.4 Suitable parents and C/F splitting

So far it was assumed that parent nodes E_i required to interpolate $i \in F$ had been fixed. In order to judge the quality of different sets of parents, a measure for the quality of interpolation is introduced first:

Definition 3.1 (Quality measure)
Let $\mathrm{E_i}$ be a set of parent nodes, and $\mathrm{q_i^p}$ be the solution of the local minimisation problems (8.31) and (8.32). The value

$$\mu(\mathrm{i}, \mathrm{E_i}) = \left| \mathrm{x_{ii}^p} \right|^2 \parallel \left(\mathrm{S'_p Y_p^{-1}} \right)^{\mathrm{T}} \mathrm{q_i^p} \parallel_2^2$$

then defines a measure for the quality of interpolation.

Only sets of parent nodes are used which are suitable in the following sense:

Definition 3.2 (Suitable parents)

For $\mathrm{i} \in \mathrm{V}$ let $\mathcal{C}_\mathrm{i} \subset \mathcal{P}(\mathrm{N}_\mathrm{i})$ be a class of possible parents. A set $\mathrm{E} \in \mathcal{C}_\mathrm{i}$ is a suitable set of parents for i w.r.t. the class $\mathcal{C}_\mathrm{i}$, if

$$\mu(\mathrm{i}, \mathrm{E}) \leq \delta \tag{15.a}$$

and

$$\theta\mu(\mathrm{i}, \mathrm{E}) \leq \inf_{\mathrm{C} \in \mathcal{C}_\mathrm{i}} \mu(\mathrm{i}, \mathrm{C}). \tag{8.33}$$

Here $\delta > 0$ is an absolute threshold judging the quality of interpolation. The parameter $0 < \theta < 1$ allow to select parents which are close to the minimum.

Typical parameters are $\delta = 1.0$ and $\theta = 0.9$.

By using this notion of suitable set of parents, a C/F splitting of the variables can be obtained by the following labeling algorithm /WAG 00/. This algorithm iteratively selects fine grid nodes for the set F, which are interpolated from pairs of neighbors $E \in \mathcal{E}^{(2)}$ which are assigned to the coarse set $\mathcal{C}$. This selection process involve a weight $\lambda(\mathrm{i}, \mathrm{E}_\mathrm{i}^{(\mathrm{j})})$ which is defined as a weighted balance between the number of newly created coarse grid nodes and the number of new connections in the coarse grid operator.

Labelling algorithm

Require: Variables V, suitable parents $(i, E^j) \in \mathcal{E}^{(2)}$

Ensure: Splitting $V = C \dot{\cup} F$

1:	$F = C = \{\}$ /* fine and coarse set are empty */
2:	$\mathrm{U} = \mathrm{V}$ /* all nodes are undecided */
3:	Compute weights λ_i^(j) := $\lambda(\mathrm{i}, \mathrm{E}_\mathrm{i}^{(\mathrm{j})})$ for all $\mathrm{i} \in \mathrm{V}$ and $\mathrm{E}_\mathrm{i}^{(\mathrm{j})} \in \mathcal{E}_\mathrm{i}^{(2)}$
4:	**while** $\mathcal{C}^{(2)} \neq \{\}$ **do**
5:	Select $\left(\mathrm{i}, \mathrm{E}_\mathrm{i}^{(\mathrm{j})}\right) \in V \times \mathcal{E}^{(2)}$ with minimal $\lambda_\mathrm{i}^\mathrm{j}$

6:	$\mathrm{F} \leftarrow \mathrm{F} \cup \{\mathrm{i}\}$ /* node i becomes fine*/
7:	$\mathrm{C} \leftarrow \mathrm{C} \cup \mathrm{E}_{\mathrm{i}}^{(\mathrm{j})}$
8:	$\mathrm{U} \leftarrow \mathrm{U} \setminus \left(\{\mathrm{i}\} \cup \mathrm{E}_{\mathrm{i}}^{(\mathrm{j})} \cap \mathrm{U}\right)$
9:	$\mathcal{E}^{(2)} \leftarrow \mathcal{E}^{(2)} \setminus \left(\bigcup_{\mathrm{k}} E_i^{(k)} \cup \{\mathrm{E} \in \mathcal{E}^{(2)} : \mathrm{i} \in \mathrm{E}\}\right)$
10:	Compute modified weights $\lambda_{\mathrm{i}}^{(\mathrm{j})}$
11:	**end while**
12:	$\mathrm{C} \leftarrow \mathrm{C} \cup \mathrm{U}$ /* Undecided nodes become coarse */

8.2.3.5 Systems of equations

For systems of equations, a diagonal interpolation is considered, i. e.,

$$(P_H^h \mathbf{u}_H)_i^\alpha = \sum_{j \in P_i} w_{ij}^{(\alpha\alpha)} u_j^\alpha \tag{8.34}$$

for fine grid nodes $i \in F$. Now the entries of the block diagonal interpolation $Q_i = (q_i^\alpha)_\alpha$ are given as the solution of

$$\min_{Q_i} \sum_\alpha x_{ii}^{\alpha\alpha} \parallel S'^T q_i^\alpha \parallel_{X^{-1}}^2, \ \forall i \in F \tag{8.35}$$

$$\text{s.t.} \langle q_i^\alpha, \tilde{\mathbf{t}}^{\boldsymbol{\alpha}} \rangle = 0, \ \forall \alpha \tag{8.36}$$

where Q_i has a prescribed sparsity pattern (8.34). The matrix X represents a diagonal scaling define d by $X = \mathrm{Diag}(x_{ii}^{\alpha\alpha}), x_{ii}^{\alpha\alpha} := |a_{ii}^{\alpha\alpha}|, \alpha \in \{p, \omega\}, i \in V$, and the diagonal entries have been redefined such that $X > 0$.

8.2.4 Convergence

In this section, a convergence theory for the method presented in section 8.2.3.3 is presented. For the remainder this section is restricted to scalar symmetric positive defi-

nite systems. For $A := \mathrm{A_h} > 0$ there is $D \;:= \mathrm{D_h} := \mathrm{Diag}(A_h) > 0$, and the following Jacobi scales are used:

$$(\cdot,\cdot)_{\alpha,h} := \langle D_h((D_h)^{-1}A_h)^{\alpha}\,\cdot,\cdot\rangle, \quad \|\cdot\|_{\alpha,h} := \sqrt{(\cdot,\cdot)_{\alpha,h}}$$

to define inner products and corresponding norms for $0 \le \alpha \le 2$.

8.2.4.1 Global results

A smoother with the iteration operator S is said to posses a **smoothing property**, /RUG 87/, if there is a constant $\sigma > 0$, such that

$$\| S \;\mathbf{e}\|_{1,h}^2 \le \|\mathbf{e}\|_{1,h}^2 - \sigma \|\mathbf{e}\|_{2,h}^2 \tag{8.37}$$

holds for all $\mathbf{e}$. The following proposition guarantees convergence and is a slight extension of the well-known convergence result in 0, /STU 01/.

Proposition 4.1 (Two-grid convergence)
Let the pre-smoother S' be non-divergent, $\|S'\|_{1,\mathrm{h}} \le 1$. The post-smoother S should satisfy the smoothing property (8.37). Moreover, assume there is a $\tau > 0$, such that the following interpolation property

$$\|(\mathrm{I} - \mathrm{PR}')\mathrm{S}'\mathbf{e}\|_{0,\mathrm{h}}^2 \le \tau \|\mathbf{e}\|_{1,\mathrm{h}}^2 \tag{8.38}$$

holds for all $\mathbf{e}$. Then the two-grid method converges and the energy norm of its operator is bounded by $\|\mathrm{STS}'\|_{1,\mathrm{h}} \le \kappa < 1$, where the contraction number is given by

$$\kappa^2 = \begin{cases} \tau/\sigma, & \tau < \sigma \\ \tau/4\sigma, & \sigma \le \tau \le 2\sigma \\ 1 - \sigma/\tau, & \tau > 2\sigma \end{cases}$$

Proof.
(i) Let $\mathbf{e} \neq 0$. As K is a projector w.r.t. $(\cdot,\cdot)_{1,h}$,

$$\|KS'\mathbf{e}\|_{1,h}^2 = \langle AKS'\mathbf{e}, KS'\mathbf{e} - P\mathbf{e}^H\rangle$$

holds for arbitrary $\mathbf{e}^H$. In particular the identity $KS'\mathbf{e} - P\mathbf{e}^H = (I - PR')S'\mathbf{e}$ holds for $\mathbf{e}^H = (R' - A_H^{-1}P^TA)S'\mathbf{e}$. Applying the Schwarz inequality yields

$$\| KS'\mathbf{e} \|_{1,h}^2 \leq \| KS'\mathbf{e} \|_{2,h} \| (I - PR')S'\mathbf{e} \|_{0,h}$$

and thus

$$\frac{\|KS'\mathbf{e}\|_{1,h}^2}{\sqrt{\tau}\|\mathbf{e}\|_{1,h}} \leq \| KS'\mathbf{e} \|_{2,h}.$$

Applying the smoothing property (18) to the vector $KS'\mathbf{e}$ yields

$$\| SKS'\mathbf{e} \|_{1,h}^2 \leq \| KS'\mathbf{e} \|_{1,h}^2 - \sigma \| KS'\mathbf{e} \|_{2,h}^2$$

$$\leq \| KS'\mathbf{e} \|_{1,h}^2 \left(1 - \frac{\sigma}{\tau}\frac{\|KS'\mathbf{e}\|_{1,h}^2}{\|\mathbf{e}\|_{1,h}^2}\right)$$

which is equivalent to

$$\frac{\|SKS'\mathbf{e}\|_{1,h}^2}{\|\mathbf{e}\|_{1,h}^2} \leq \frac{\|KS'\mathbf{e}\|_{1,h}^2}{\|\mathbf{e}\|_{1,h}^2}\left(1 - \frac{\sigma}{\tau}\frac{\|KS'\mathbf{e}\|_{1,h}^2}{\|\mathbf{e}\|_{1,h}^2}\right). \tag{8.39}$$

after dividing by $\| \mathbf{e} \|_{1,h}^2 > 0$.

(ii) As the left hand side is non-negative,

$$\frac{\|KS'\mathbf{e}\|_{1,h}^2}{\|\mathbf{e}\|_{1,h}^2} \leq \frac{\tau}{\sigma}$$

holds, and one obtains the convergence estimate $\tau < \sigma$.

(iii) Let $\sigma \leq \tau$. As K is a projection and S' is non-divergent

$$x_e := \frac{\|KS'\mathbf{e}\|_{1,h}^2}{\|\mathbf{e}\|_{1,h}^2} \in [0,1].$$

holds. The right hand side of (8.39) is a polynomial of degree 2, $p(x_e) := x_e(1 - \frac{\sigma}{\tau}x_e)$. The maximum is obtained for $x_e^* = \tau/2\sigma$ with $p(x_e^*) = \tau/4\sigma$. If $x_e^* > 1$ and thus $\tau > 2\sigma$, one gets at least $p(x_e) \leq 1 - \sigma/\tau$.

Note that the version 0, /STU 01/ only considered post-smoothing. In property (8.38) pre-smoothing S' and interpolation P are considered as one unit influencing the constant τ. The post-smoother S only influences the constant σ.

8.2.4.2 Local results

Under appropriate assumptions, the interpolation property (8.38) can be expressed locally. It is imposed that a set of the following local criteria L1-L4:

Assumption L1.
Let $\mathrm{A} > 0$. For a $\mathrm{C/F}$-splitting of the variables let there be matrices $\mathrm{A_i} \geq 0$ and a constant $\overline{\mathrm{C}} > 0$, such that

$$\sum_{i \in F} A_i \leq \overline{C} A. \tag{8.40}$$

Assumption L2.
For $\mathrm{i} \in \mathrm{F}$ there are $\mathrm{A_i}$-orthogonal subspaces $\mathrm{G_i}$ and $\mathrm{B_i}$ which allow for a decomposition

$$\mathbb{R}^V = G_i \oplus B_i. \tag{8.41}$$

The matrices A_i induce semi-norms

$$|\cdot|_{1,i}^2 := (\cdot,\cdot)_{1,i} := \langle A_i \cdot,\cdot \rangle$$

which are full norms on the A_i-orthogonal complement of $\mathrm{Ker}(A_i)$. Eq. (8.40) implies

$$\sum_{i \in F} |\mathbf{e}|_{1,i}^2 \leq \overline{C} \parallel \mathbf{e} \parallel_{1,h}^2, \quad \forall \mathbf{e} \in^n.$$

where according to (21.b) the identity $|\mathbf{e}|_{1,i}^2 = |\mathbf{e}_g^{(i)}|_{1,i}^2 + |\mathbf{e}_b^{(i)}|_{1,i}^2$ holds for $\mathbf{e} = \mathbf{e}_g^{(i)} + \mathbf{e}_b^{(i)}$.

Moreover imposed are the following two assumptions for the subspaces G_i and B_i:

Assumption L3.
There is a constant $\gamma > 0$, such that

$$\gamma \parallel \mathbf{e}_{\mathrm{g}}^{(\mathrm{i})} \parallel_{0,\mathrm{h}}^2 \leq |\mathbf{e}_{\mathrm{g}}^{(\mathrm{i})}|_{1,\mathrm{i}}^2 \tag{8.42}$$

for all $\mathrm{i} \in \mathrm{F}$ and $\mathbf{e}_{\mathrm{g}}^{(\mathrm{i})} \in \mathrm{G_i}$.

Assumption L4.

There is a constant $\beta > 0$, such that

$$a_{ii}\langle q_i, S'\mathbf{e}_b^{(i)}\rangle^2 \leq \beta |\mathbf{e}_b^{(i)}|_{1,i}^2, \tag{8.43}$$

for all $i \in F$ and all $\mathbf{e}_b^{(i)} \in B_i$

These additional prerequisites characterise the splitting given in Assumption **L2** (8.41) in greater detail: While (8.42) guarantees the spectral equivalence of the operators D and A_i on the subspace G_i, (8.43) is an interpolation property on the subspace B_i. The spaces G_i and B_i thus represent good and bad subspaces w.r.t. the smoother: While oscillatory components in G_i can be treated by smoothing, smooth components in B_i must be treated by interpolation.

Remark 4.1

If the objective (8.31) is constructed using the scalings $X = Y = \sqrt{\mathrm{Diag}(A)}$, one obtains

$$\delta_i := \mu(i, E_i) = \max_{\|\mathbf{e}\|_{0,h}^2} a_{ii}\langle q_i, S'\mathbf{e}\rangle^2 \leq \delta, \tag{8.44}$$

if each variable $i \in F$ is interpolated from suitable sets of parents E_i only.

Remark 4.2

From (8.42) and (8.43) is concluded that $\mathrm{Ker}(A_i) \subset B_i$ and that the corresponding vectors must be interpolated exactly, i. e. ,

$$\langle q_i, S'\mathbf{e}\rangle = 0, \quad \forall \mathbf{e} \in \mathrm{Ker}(A_i).$$

For FAMG this must be guaranteed by appropriate constraints in the filter condition (8.32).

Using these assumptions, the following estimate can be derived:

Lemma 4.1

If (8.40) to (8.44) are satisfied, then

$$\sqrt{a_{ii}}\langle q_i, S'\mathbf{e}\rangle \leq \left(\sqrt{\beta} + \sqrt{\frac{\delta_i}{\gamma}}\right)|\mathbf{e}|_{1,i}, \quad \forall \mathbf{e}. \tag{8.45}$$

Proof.

For $i \in F$ and $\mathbf{e} \in \mathbb{R}^n$ consider the decomposition $\mathbf{e} = \mathbf{e}_b^{(i)} + \mathbf{e}_g^{(i)}$ from (8.41). On the one hand (8.42) implies

$$\sqrt{a_{ii}}\langle q_i, S'\mathbf{e}_g^{(i)}\rangle \leq \sqrt{\delta_i} \parallel \mathbf{e}_g^{(i)} \parallel_{0,h} \leq \sqrt{\frac{\delta_i}{\gamma}} |\mathbf{e}_g^{(i)}|_{1,i}.$$

On the other hand (8.43) implies,

$$\sqrt{a_{ii}}\langle q_i, S'\mathbf{e}_b^{(i)}\rangle \leq \sqrt{\beta} |\mathbf{e}_b^{(i)}|_{1,i}.$$

As $|\mathbf{e}_g^{(i)}|_{1,i} \leq |\mathbf{e}|_{1,i}$ and $|\mathbf{e}_b^{(i)}|_{1,i} \leq |\mathbf{e}|_{1,i}$ the claim follows from a summation of the former inequalities.

Inequality (8.45) is finally sufficient for the convergence of the two-grid method in the sense of Proposition 4.1:

Proposition 4.2 (Local-to-global estimate)

Let the C/F-splitting satisfy **L1**. For every $\mathrm{i} \in \mathrm{F}$ let there be a constant $\tau_\mathrm{i} > 0$, such that the local estimate

$$\mathrm{a_{ii}}\langle \mathrm{q_i}, S'\mathbf{e}\rangle^2 \leq \tau_\mathrm{i} \ |\mathbf{e}|_{1,\mathrm{i}}^2, \quad \forall \mathbf{e}. \tag{8.46}$$

Then also

$$\parallel (I - PR')S'\mathbf{e} \parallel_{0,h}^2 \leq \tau \ \parallel \mathbf{e} \parallel_{1,h}^2, \quad \forall \mathbf{e},$$

where $\tau = \max_{i \in F} \tau_i \overline{C}$ where $\overline{C}$ is from (8.40).

Proof.

For $\mathbf{e}$ one gets

$$\parallel (I - PR')S'\mathbf{e} \parallel_{0,h}^2 = \sum_{i \in F} a_{ii}\langle q_i, S'\mathbf{e}\rangle^2 \leq \sum_{i \in F} \tau_i \ |\mathbf{e}|_{1,i}^2 \leq \max_{i \in F} \tau_i \overline{C} \ \parallel \mathbf{e} \parallel_{1,h}^2.$$

8.2.4.3 Example: Choice of local operators

The measure introduced in (8.33) are similar to those used for AMGe /BRE 01/. In this section, FAMG is formulated as an element-free variant of this method /HEN 01/. In contrast to the former mentioned works, the smoother S' is included into the construction of interpolation. Consider a finite element context with P1/Q1 ansatz functions. For some triangulation $\mathcal{T}$ the stiffness matrix A is a result of local element stiffness matrices,

$$A = \sum_{t\in\mathcal{T}} A_t, \quad \text{with} \quad A_t = A_t^T \geq 0, \quad t \in \mathcal{T}.$$

Local element-based matrices for vertex i can then be defined by

$$A_i^{(E,1)} := \sum_{t:\, i\in N(t)} A_t, \quad A_i^{(E,2)} := \sum_{t:\, N_i\cap N(t)\neq\emptyset} A_t$$

where $N(t)$ denote the corner vertexes of element t, and N_i denotes the neighbourhood of vertex i. For this decomposition, the following estimates hold:

$$\sum_{i\in F} A_i^{(E,d)} \leq \overline{C}_{E,d}\ A, \quad d = 1{,}2,$$

where the super script index $d = 1{,}2$ indicates the depth of elements considered, and

$$\overline{C}_{E,1} := \max_{t\in\mathcal{T}} |N(t)\cap F|, \quad \overline{C}_{E,2} := \max_{t\in\mathcal{T}} \left|\bigcup_{s\in\mathcal{T}:N(s)\cap N(t)\neq\emptyset} N(s)\cap F\right|.$$

Remark 4.3

Approaches with objectives similar to (8.33) are AMG based on computational molecules /KRA 06/, /KRA 07/ and AMG based on least-squares interpolation /BRA 11/, /KAH 09/.

For the inequality (8.45) in Lemma 4.1 the generalised eigenvalue problem is considered:

$$\lambda D\mathbf{e} = A_i\mathbf{e}. \tag{8.47}$$

For all eigenvectors $\mathbf{e}_k$ with $\lambda_k \geq 0$ and $\gamma > 0$ define spaces

$$B_i := B_\gamma(D, A_i) = \langle \mathbf{e}_k : \lambda_k < \gamma \rangle, \quad G_i := G_\gamma(D, A_i) = \langle \mathbf{e}_k : \lambda_k \geq \gamma \rangle, \tag{8.48}$$

such that (8.42) holds for the subspace G_i. As a proof of (8.43) is difficult in a general setting, the idea is illustrated by the following model problem: Considered is a discretization of Poisson's equation on the rectangle $(M, N \cdot a)$ with $M, N \in \mathbb{N}$ an aspect ratio $a > 0$. The discrete grid is defined by $\Omega_h = \{(m, n \cdot a): 0 \leq m \leq M, 0 \leq n \leq N\}$. For bilinear elements, the element stiffness matrix for the element $t = (1,0) \times (0, a)$ reads

$$A_t = \frac{1}{6a}\begin{pmatrix} 2(1+a^2) & 1-2a^2 & -1-a^2 & -2+a^2 \\ 1-2a^2 & 2(1+a^2) & -2+a^2 & -1-a^2 \\ -1-a^2 & -2+a^2 & 2(1+a^2) & 1-2a^2 \\ -2+a^2 & -1-a^2 & 1-2a^2 & 2(1+a^2) \end{pmatrix}. \tag{8.49}$$

Now, the local matrix $A_i := A_i^{(E,2)}$ is considered and its local support by $L_i := \langle \boldsymbol{\varepsilon}_k : A_i \boldsymbol{\varepsilon}_k \neq 0 \rangle$ is defined. For a Jacobi smoother followed by an i-smoothing step S', $\langle q_i, S'\mathbf{v} \rangle = 0$ holds all $\mathbf{v} \perp L_i$. For proving (8.43) it is thus sufficient to restrict to L_i. To further simplify, a vertex i in the interior of the domain is considered. Schematically, this is shown in Fig. 8.4.

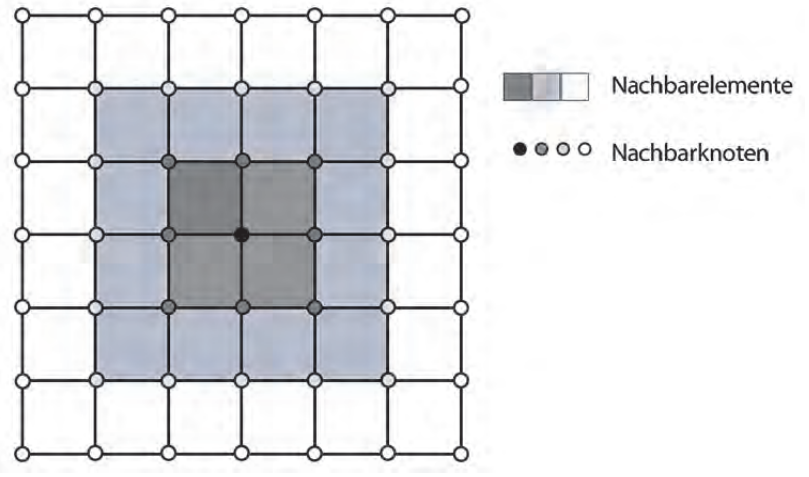


Fig. 8.4 Visualisation of the operator $\boldsymbol{A_i}$ and its local support $\boldsymbol{L_i}$ for the model problem.
Elements in the neighbourhood are shown in dark ($\boldsymbol{d} = \mathbf{1}$) and light ($\boldsymbol{d} = \mathbf{2}$) grey. Adjacent grid nodes belonging to $\boldsymbol{L_i}$ are also shown in grey. White elements and nodes are not considered in the local construction

The space L_i has a basis of 25 eigenvectors $\mathbf{e}_k$ with eigenvalues λ_k. The constant, $\mathbf{e}_0 = \mathbf{1}$, is the only eigenvector for the eigenvalue 0. Given an appropriate filter condition, there is obtained

$$\langle q_i, S'\boldsymbol{e}_0\rangle = \langle q_i, \mathbf{1}\rangle = 0.$$

It is assumed that all following vectors $(\mathbf{e}_k)_{k\geq 1}$ are normalised w.r.t. $|\cdot|_{1,i}$. These then provide an orthonormal basis of $L_i \backslash \langle \mathbf{e}_0 \rangle$. Defining the constants

$$b_k := \sqrt{a_{ii}\langle q_i, S'\mathbf{e}_k\rangle^2}, \quad \beta_k := \sum_{j=1}^{k} b_j, \quad k \geq 1,$$

there is obtained

$$\sqrt{a_{ii}}|\langle q_i, S'\mathbf{e}\rangle| \leq \sum_{k\geq 1} \sqrt{b_k}\left|(\mathbf{e}, \mathbf{e}_k)_{1,i}\right| \leq \sqrt{\sum_{k\geq 1} b_k}\sqrt{\sum_k (\mathbf{e}, \mathbf{e}_k)_{1,i}^2} =: \sqrt{\beta_{24}}|\mathbf{e}|_{1,i}, \ \forall \mathbf{e},$$

which is equivalent to (8.43) and (8.46) respectively. Fig. 8.5 provides an analysis of these measures results for the isotropic (top, a=1) and the anisotropic (bottom, a=10) cases respectively. Vertex i is interpolated by its top and bottom neighbor, the interpolation weight is 0.5. The smoother S' is damped by a factor $\omega = 1/2$. The figure show the distribution of the eigenvalues λ_k and of the corresponding interpolation measures b_k and β_k for $k = 1, \ldots, 24$. As β_k provides the interpolation estimate for the subspace $\langle \mathbf{e}_j | j = 0 \ldots k\rangle$, it provides an estimate for the subspace B_i defined in (8.43).

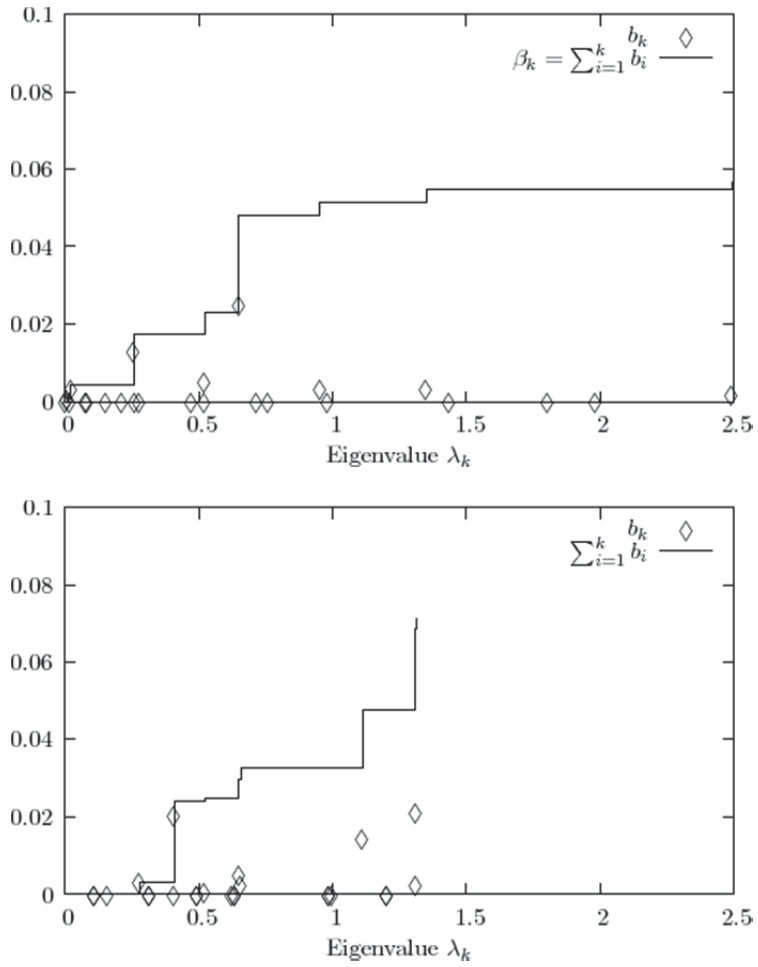


Fig. 8.5 Distribution of the eigenvalues $\boldsymbol{\lambda_k}$ of the generalized eigenvalue problem (8.47) for model problem (8.49).

The measures $\boldsymbol{b_k}$ and $\boldsymbol{\beta_k}$ are indicators for the interpolation quality for certain eigenvectors. Results are shown for the isotropic model ($\boldsymbol{a} = \mathbf{1}$, top), and the anisotropic model ($\boldsymbol{a} = \mathbf{10}$, bottom)

8.2.5 Effective preconditioners

In this section the different ingredients for AMG solvers for density driven flow are presented. This begins with a study of the algebraically smooth error in Section 8.2.5.1. In Section 8.2.5.2 a transformation is introduced which decouples pressure and concentration locally, which also aims to eliminate negative diagonal entries. Subsections 8.2.5.3 8.2.5.5 finally introduce various preconditioners applicable to problems of density driven flow.

8.2.5.1 Algebraically smooth error

The proper choice of AMG components depends on the character of **the algebraically smooth error,** i. e., those error components which are not treated efficiently by the smoother. In order to study this further there is focussed on Elder's problem, cf. /VOS 87/, a test case from a practical application which is discretized using the fully non-linear equations. Fig. 8.6 shows the solution at one particular point in time, which serves as the linearization point $(p_0, \omega_0)^T$ for the next Newton iteration. Instead of plotting the pressure p, which is only defined up to a constant in this case, only the salt mass fraction ω and the velocity field $\mathbf{q}$ are shown.

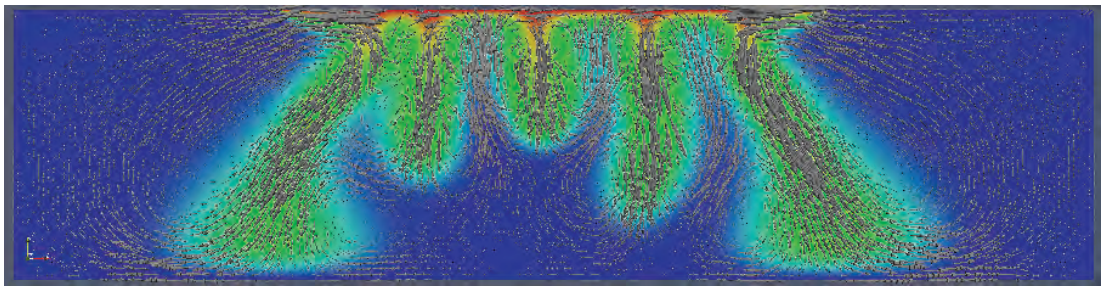


Fig. 8.6 Solution at one particular point in time.
In this point, the problem is linearized resulting in the Jacobian investigated in the further course of this study

In this configuration, the algebraically smooth error for both components can then be studied independently. Fig. 8.7 illustrates the smoothed error $S(\mathbf{r}_p^T, 0)^T$: In this case the initial error before relaxation is concentrated in the p-component. In the p-part, one observes a typical behaviour for elliptic problems. Due to the coupling, the error is also distributed into the salt mass fraction. Here, the error tends to follow the velocity profile for $\mathbf{q}_0$. Complementarily Fig. 8.8 illustrates the smoothed error $S(0, \mathbf{r}_\omega^T)^T$: Here, the error is first concentrated in the ω-component, but then distributed into the p-component by relaxation.

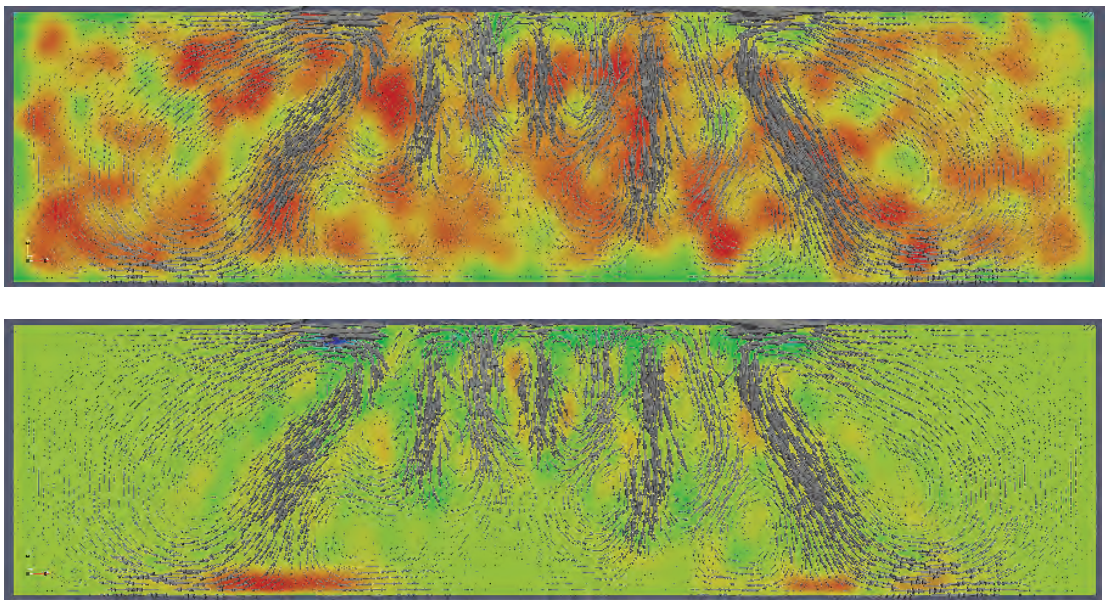


Fig. 8.7 Smooth Error after 3 SGS relaxation sweeps after an initialisation with a random vector $(\mathbf{r}_{\boldsymbol{p}}, \mathbf{0})^T$;
results for pressure $\boldsymbol{p}$ (top), and salt mass fraction $\boldsymbol{\omega}$ (bottom).

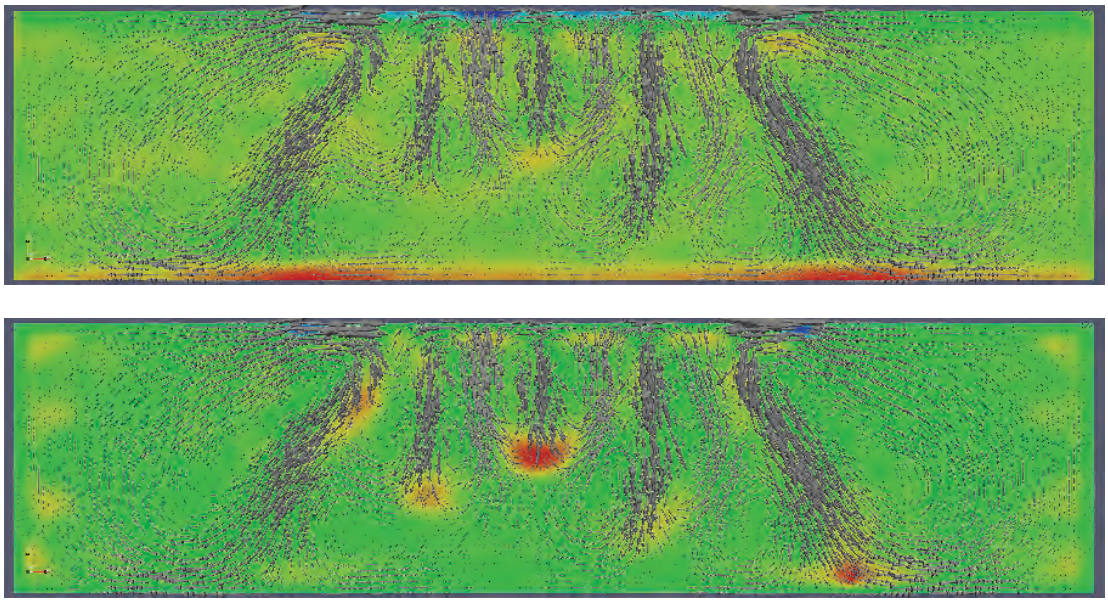


Fig. 8.8 Smooth Error after 3 SGS relaxation sweeps after an initialization with a random vector $(\mathbf{0}, \mathbf{r}_{\boldsymbol{\omega}})^T$; arrangement of $\boldsymbol{p}$ and $\boldsymbol{\omega}$ as in Fig. 8.7.

8.2.5.2 Left Transformation

One problem for FAMG results from potentially large negative diagonal entries in the $J^{\omega\omega}$ block. This is due the derivative $\mathbf{q}_0$, in the equations for the salt mass fraction (8.24). As a remedy, the following strategy is used: Instead solving $J\mathbf{u} = \mathbf{f}$ one can alternatively consider a (left-)transformed system

$$LJ\boldsymbol{u} = L\boldsymbol{f}. \tag{8.50}$$

As motivated by analysis for the model problem (8.2.2.3) it is desirable to imitate a decoupling of p and ω locally. This can be achieved by a multiplication with a block-diagonal matrix $L = \mathrm{Diag}(L_{ii})$ which transforms the diagonal entries

$$J_{ii} = \begin{pmatrix} j^{(p,p)} & j^{(p,\omega)} \\ j^{(\omega,p)} & j^{(\omega,\omega)} \end{pmatrix} \tag{8.51}$$

into

$$(LJ)_{ii} = \begin{pmatrix} j^{(p,p)} & j^{(p,\omega)} \\ 0 & j^{(\omega,\omega)} - \frac{j^{(\omega,p)}}{j^{(p,p)}} j^{(p,\omega)} \end{pmatrix}. \tag{8.52}$$

Similar techniques have been used for multiphase flow, e. g., /BEH 82/, /WAL 85/, /SCH 03/, /LAC 03/, /CLE 07/, but focus to eliminate $j^{(p,\omega)}$. Note that transformation (8.50) does not modify the properties of standard smoothers. It does not preserve the norm of the defect however.

8.2.5.3 Linear Gauss-Seidel block type preconditioners

A general concept to solve the linear system of equation is to employ block Gauß-Seidel type method (BGS). In this case both equations are solved iteratively in the order of their occurrence. For a (p, ω)-ordering one obtains:

$$\hat{A}^{pp}(\mathbf{u}_{k+1}^{p} - \mathbf{u}_{k}^{p}) = \mathbf{f}^{p} - A^{pp}\mathbf{u}_{k}^{p} - A^{p\omega}\mathbf{u}_{k}^{\omega}, \tag{8.53}$$

$$\hat{A}^{\omega\omega}(\mathbf{u}_{k+1}^{\omega} - \mathbf{u}_{k}^{\omega}) = \mathbf{f}^{\omega} - A^{\omega p}\mathbf{u}_{k+1}^{p} - A^{\omega\omega}\mathbf{u}_{k}^{\omega}. \tag{8.54}$$

Here, $\hat{A}^{pp}$ and $\hat{A}^{\omega\omega}$ are preconditioners for A^{pp} and $A^{\omega\omega}$ respectively. The iteration matrix is given by

$$S_{BGS}^{(p,\omega)} = I - (\hat{A}_{\mathrm{BGS}}^{(p,\omega)})^{-1} A$$

$$= \begin{pmatrix} I & 0 \\ 0 & I \end{pmatrix} - \begin{pmatrix} \hat{A}^{pp} & 0 \\ A^{\omega p} & \hat{A}^{\omega\omega} \end{pmatrix}^{-1} \begin{pmatrix} A^{pp} & A^{p\omega} \\ A^{\omega p} & A^{\omega\omega} \end{pmatrix}$$

$$= \begin{pmatrix} I - (\hat{A}^{pp})^{-1} A^{pp} & -(\hat{A}^{pp})^{-1} A^{pw} \\ -(\hat{A}^{\omega\omega})^{-1} A^{\omega p} \left(I - (\hat{A}^{pp})^{-1} A^{pp}\right) & I - (\hat{A}^{\omega\omega})^{-1} \hat{S}^{\omega\omega} \end{pmatrix}, \tag{8.55}$$

where $\hat{S}^{\omega\omega} := A^{\omega\omega} - A^{\omega p}(\hat{A}^{pp})^{-1} A^{pw}$ is a Schur complement w.r.t. $\hat{A}^{pp}$. Note that choosing $\hat{A}^{\omega\omega}$ as an inexact solver for $\hat{S}^{\omega\omega}$ results in an Uzawa-type method. If the iteration is performed in the reversed (ω, p)-ordering, one obtains

$$S_{BGS}^{(\omega,p)} = I - (\hat{A}_{\mathrm{BGS}}^{(\omega,p)})^{-1} A$$

$$= \begin{pmatrix} I & 0 \\ 0 & I \end{pmatrix} - \begin{pmatrix} \hat{A}^{pp} & A^{p\omega} \\ 0 & \hat{A}^{\omega\omega} \end{pmatrix}^{-1} \begin{pmatrix} A^{pp} & A^{p\omega} \\ A^{\omega p} & A^{\omega\omega} \end{pmatrix}$$

$$= \begin{pmatrix} I - (\hat{A}^{pp})^{-1} \hat{S}^{pp} & -(\hat{A}^{pp})^{-1} A^{pw} \left(I - (\hat{A}^{\omega\omega})^{-1} A^{\omega\omega}\right) \\ -(\hat{A}^{\omega\omega})^{-1} A^{\omega p} & I - (\hat{A}^{\omega\omega})^{-1} A^{\omega\omega} \end{pmatrix}, \tag{8.56}$$

where now $\hat{S}^{pp} := A^{pp} - A^{p\omega}(\hat{A}^{\omega\omega})^{-1} A^{\omega p}$. Both formulations indicate conditions for a convergent method:

1. The preconditioners $(\widehat{\mathrm{A}}^{\mathrm{pp}})^{-1}$ and $(\widehat{\mathrm{A}}^{\omega\omega})^{-1}$ should be chosen such that the iterations are fast to converge. For the system treated in (p, ω)-order the iterations $\mathrm{I} - (\widehat{\mathrm{A}}^{\mathrm{pp}})^{-1}\mathrm{A}^{\mathrm{pp}}$ and $\mathrm{I} - (\widehat{\mathrm{A}}^{\omega\omega})^{-1}\hat{\mathrm{S}}^{\omega\omega}$ should converge quickly. For the system treated in (ω, p)-order the iterations $\mathrm{I} - (\widehat{\mathrm{A}}^{\omega\omega})^{-1}\mathrm{A}^{\omega\omega}$ and $\mathrm{I} - (\widehat{\mathrm{A}}^{\mathrm{pp}})^{-1}\hat{\mathrm{S}}^{\mathrm{pp}}$ should converge quickly.
2. In any case, both off-diagonal blocks $(\widehat{\mathrm{A}}^{\mathrm{pp}})^{-1}\mathrm{A}^{\mathrm{pw}}$ and $(\widehat{\mathrm{A}}^{\omega\omega})^{-1}\mathrm{A}^{\omega\mathrm{p}}$ should be sufficiently small. In defiance of the tentative assumption that $\mathrm{A}^{\omega\mathrm{p}} \approx 0$, especially the later fact cannot be guaranteed. For (8.56), this can however be compensated by a good approximation $(\widehat{\mathrm{A}}^{\omega\omega})^{-1}$.

8.2.5.4 Two-stage preconditioners

The formulation (8.52) was previously used as an indicator that the system in (8.50) is essentially decoupled. In this case one is tempted to solve for ω first and then for p by means of (8.56). Numerical experiments show however that this is not sufficient in many cases. Interestingly enough, this may produce similar results as iteration (8.55). Inspired by this observation, the class of so called **two-state preconditioners**, defined by the iteration matrix

$$S_{2-\text{stage}} = S_{\text{block}} S_{\text{pres}}$$

may be an attractive alternative alternative. Here S_{pres} results from (8.55) by formally setting $(\hat{A}^{\omega\omega})^{-1} = 0$ and S_{block} is a simple block smoothing scheme. The idea of this approach is to apply a global correction for the pressure block A^{pp} first. This is then followed by second a sweep with a standard point-wise smoother. This additional relaxation aims to reduce the error in ω locally and to recouple the equations.

8.2.5.5 Monolithic AMG preconditioner

The last alternative is to applty a monolithic AMG solver as presented in in subsection 8.2.3.5. This is not applicable, if the matrix $A^{\omega\omega}$ contains large negative diagonal entries. In this case it is advisable – if not mandatory – to use the transformed system (8.50). As the method has already been described at length, the essential features are summarized at this point only.

The operator S' for problem (8.28)/(8.29) is provided by a point-block-Jacobi smoother. The smoother respects the local coupling of the unknowns. The filter vectors $\tilde{\mathbf{t}}^{\boldsymbol{\alpha}}$ are provided by the (smoothed) constants $t_0^p = (\mathbf{1}, 0)^{\mathrm{T}}$ and $t_0^{\omega} = (0, \mathbf{1})^{\mathrm{T}}$. The filter vectors on the next level are obtained by restricting $t_{k+1}^{\alpha} = R' \tilde{t}_k^{\alpha}$, and one smoothing step, $\tilde{t}_k^{\alpha} = S^{\alpha} t_k^{\alpha}$. In order to preserve linear independence, this smoothing step involves an uncoupled smoother S^{α} which is derived from the principal part $A^{\alpha\alpha}$. Additionally, the sub-matrix for the pressure A^{pp} is used for the definition of the strength of connections. Finally, fine grid nodes are interpolated only, if $-0.1 \leq (S')_{ii}^{\alpha\alpha} \leq 0.5$. This helps to detect nodes, where the smoother diverges and enforces that this belong to the coarse grid.

8.2.5.6 Nonlinear block solvers

In this work there was focused on a solution of the fully coupled system (8.12) and (8.13) by a Newton method. An alternative to the fully coupled method may be given by the **iterative coupling** algorithm /ORT 66/, /HAG 75/. This is the non-linear analogue of the preconditioners presented in subsection 8.2.5.3, cf. /STU 07/, /LUB 09/. Within each of the sub steps, a non-linear system is solved. Tests with density driven flow are however still in infancy.

8.2.6 Numerical Experiments

In the following section the performance of the method is investigated for two different test cases. Three different solvers are compared: The first is a standard geometric multigrid solver (GMG). The second is a (ω, p)-Block-Gauss-Seidel scheme (BGS). Blocks are formed by unknowns according to (8.52): It is solved first for ω, then for p. This provides an exact solver for problems with $A^{\omega p} = 0$. Finally, a monolithic AMG (FAMG) solver is considered. In this case both unknowns are solved simultaneously. All methods serve as a preconditioner in a BiCGSTAB method. A V(1,1)-cycle is used, the smoother is a symmetric Gauß-Seidel scheme with a particular stabilisation strategy (/JOH 04/). Note that the presented comparisons must be understood as a proof of concept: As the AMG method requires an additional setup, they are usually less efficient than their geometric counter parts.

8.2.6.1 Test 1: Saltpool problem

The first model problem is derived from the saltpool benchmark, cf. /JOH 06/. A two-dimensional domain $(0, \sqrt{2}a) \times (0,1)$ is considered which is resolved by a structured quadrilateral grid (2×66049 dof). In order to investigate the robustness with respect to geometric isotropies, the parameter a characterises a stretching in x-direction. Initially, the box is filled a fluid with a 10 % salt mass fraction in the lower 30 % of the box. During the course of the experiment, fresh water is injected in the upper left corner, while it is removed in the upper right corner by appropriate dirichlet conditions.The walls are treated as impermeable boundaries. Note that for AMG the choice of the coarse grid solver is crucial, as coarse matrices may become singular since the coarse grids do not contain Dirichlet nodes any more. Therefore an iterative scheme is applied as a coarse grid solver.

Numerical Results are provided in Tab. 8.2. As expected geometric multigrid deteriorates slightly with an increasing anisotropy a. The performance of BGS and FAMG is robus w.r.t. a. The convergence of BGS however is an indicator that essentially a nonlinear decoupling is observed: Provided that there is a large difference in the density, salt and fresh water phases essentially decouple. In the mixing region, the flow actually follows the boundary layer, and mixing is primarily due to diffusion.

Tab. 8.2 Convergence results for the salt pool problem. Given are the number of time steps, the total number of Newton iterations (#nl), the total number of linear iterations (#lin) as well as the maximum and average number of linear iterations per Newton iteration(#lin/#nl)

Anisotropy a	**Method**	**Steps**	**#nl**	**#lin**	**#lin/#nl (max)**	**#lin/#nl (ave)**
1	GMG	49	132	1183	53	8.96
	BGS	44	117	2603	100	22.25
	FAMG	43	113	839	13	7.42
2	GMG	46	113	1356	65	12.00
	BGS	43	109	2295	63	21.06
	FAMG	43	109	826	14	7.58
4	GMG	43	99	1529	43	12.72
	BGS	43	101	2213	74	21.91
	FAMG	43	101	660	11	6.53
8	GMG	43	113	2239	51	19.8
	BGS	43	111	2231	70	20.01
	FAMG	43	111	705	13	6.35

8.2.6.2 Test 2: Layered Aquifer

The second problem serves as a test how well the method treats problems with variable and discontinuous permeabilities. Fig. 8.9 shows the computational domain of an aquifer over a salt dome. The triangular, unstructured grid consists of 24257 vertices. The hydro-geological properties, in particular the permeabilities vary in space and follow a log-normal distribution. Two models are considered: The Mixed Model includes sand, fine sand and silt, whereas the Sand model has only one hydro-geological unit. A more detailled description is provided in /FEI 99/.

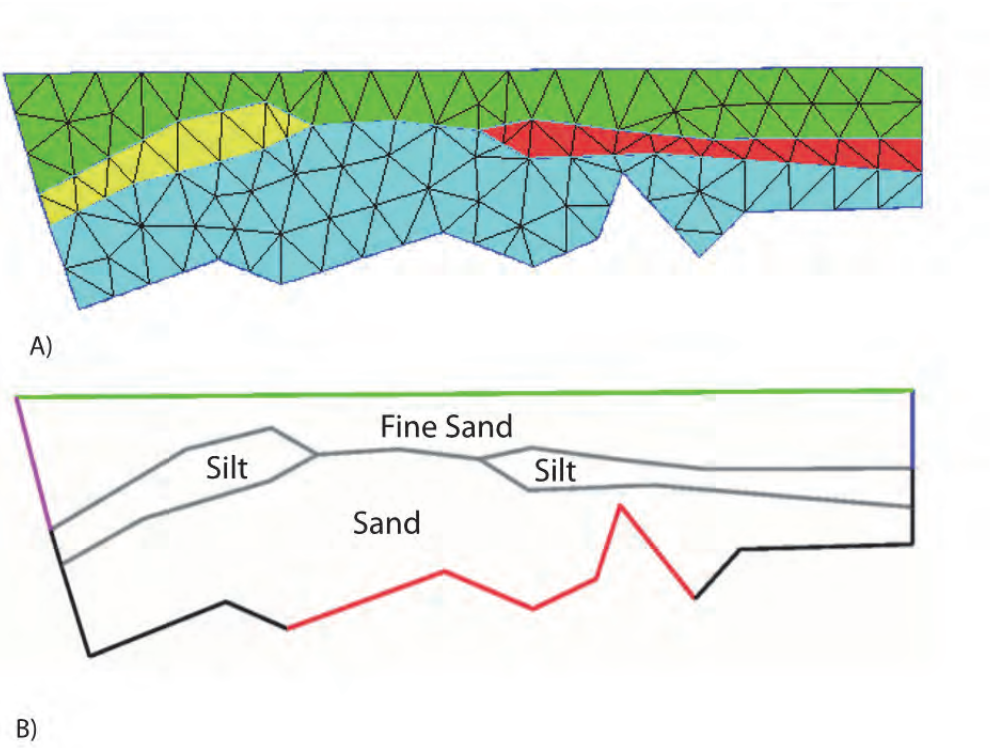


Fig. 8.9 A) Coarse grid and B) different hydro-geological areas and boundary conditions for the layered aquifer.
On the red surface on the bottom the fluid is in contact with a salt dome. Picture B): /FEI 99/

Numerical results for this model are provided in Tab. 8.3. First there was noted that the BGS method does not converge any more which indicates that problems are now completely coupled. The FAMG method tends to be a little bit more robust than geometric multigrid. In particular, it requires less time steps.

Tab. 8.3 Convergence results for the aquifer problem, with column data as in Tab. **8.2**. The block-Gauß-Seidel-scheme (BGS) does not converge

Medium	Method	Steps	#nl	#lin	#lin/#nl (max)	#lin/#nl (ave)
Mixed	GMG	18	157	4186	99	26.66
	FAMG	15	127	2048	87	16.13
Sand	GMG	40	326	15297	119	46.92
	FAMG	30	260	7125	97	27.40

8.2.7 Test 3: Norderney – Development of fresh water lenses

The last test problem is from a simulation of the development of a fresh water lense under the island of Norderney in the German North Sea. The problem and geometry definitions are given in /SCH 04/. The problem is defined on a hexahedral grid (2×12322 dof).

Tab. 8.4 Convergence results for the Norderney problem, with column data as in Tab. 8.2. The block-Gauß-Seidel-scheme (BGS) does not converge

Type	Method	Steps	#nl	#lin	#lin/#nl (max)	#lin/#nl (ave)
Boussinesq Approximation	GMG	85	171	7294	132	55.26
	FAMG	11	36	716	78	19.89
Full Equations	GMG	88	178	10890	160	61.18
	FAMG	11	30	807	71	26.90

Numerical results are shown in Tab. 8.4. In this case, the FAMG scheme is clearly superior to standard geometric multigrid. This is presumably due to a good resolution of the geometric anisotropies. This holds true for the Boussinesq approximation and the full equations as well. It should be noted that in this case, the additional setup pays off and the FAMG solver is faster than the geometric solver, even when the additional setup is considered (data not shown).

8.3 Parallelisation

8.3.1 Introduction

During the last decade the power and scale of large computer clusters has risen dramatically. This allows scientists of different fields to address their computational problems with an even greater attention to detail. However, constructing a general purpose simulation environment, which efficiently uses the power supplied by modern super computers, gets more and more challenging.

In the progress of enhancing the simulation framework UG with new solvers and refinement strategies on unstructured hierarchical grids, a challenge was to improve the existing parallelization framework, which was no longer suited for the arising challenges. The new parallelization layer should not only be easily usable in different modules of the main framework (e. g. the grid- and algebra libraries) but should also allow the construction of different and highly specific parallel solvers, like e. g. adaptive geometric multigrid solvers, algebraic multigrid solvers and domain decomposition methods, eventually allowing to further combine those. It thus was clear that a parallelization framework not only had to be highly adjustable to different data structures (e. g. hybrid adaptive multigrids, sparse matrices) but also to different algorithms, each requiring specific communication structures. To allow maximal flexibility and interoperability, there was decided against a central facility which handled all of the parallelization related issues and instead introduced a lightweight programming library, which allows to define communication structures on arbitrary distributed object sets and to perform communication between those objects using the established structures.

With the **parallel communication layer** (pcl) /GRO 99/ a set of concepts has been developed addressing the described considerations. An implementation using C++

template mechanisms has also been created and successfully been used in the simulation environment UG.

While loosely sharing some concepts with existing implementations for distributed graphs (e. g. DDD /BIR 94/) the pcl library takes a unique approach in its core design. Instead of specifying a black box which handles all of the parallelization issues, the pcl defines a set of basic classes, which can be freely used to organize distributed graphs and to communicate between overlapping or linked sections of those graphs. This lightweight concept is a strength of the pcl, since it allows to construct communication structures tightly adjusted to the specific needs of highly specialized algorithms (like e. g. algebraic multigrid solvers) without affecting communication structures required by other sections of the program (like e. g. the underlying distributed adaptive multigrid).

The lightweight nature of the pcl furthermore allows to use it to parallelize existing codes, since no specific demands are present for the nodes in the distributed graphs nor to the graphs themselves. This has the great advantage that parallel and serial codes can be clearly separated and that code repetition is not an issue.

One major difference to many existing libraries is especially noteworthy: the pcl library does not require global ids for the maintained distributed objects. While it is possible to generate global ids based on the given distributed graphs at any time, the pcl itself does not require them for its internal organization. As detailed below, this feature allows to minimize communication during dynamic tasks like adaptive parallel grid refinement etc.

8.3.2 Concepts

In the following section the key concepts of the *pcl* are briefly discussed. For a thorough discussion please refer to /GRO 99/.

Interface: In an Interface I_{AB} objects on a process A, which are related to objects on a process B, are stored in a determined order. This interface lies on process A. At the same time there has to be an interface I_{BA} on process $|I_{AB}| = |I_{BA}|$. The order of the objects in the interfaces is crucial, since during communication between processes A and B data is exchanged between the i-th object in interface I_{AB} and the i-th object in interface I_{BA}.

Layout: On a process A several Interfaces I_{AB}, I_{AC}, … can be grouped together in a Layout L. Layouts can e. g. be used to group all interfaces on a process that have a special property, e. g. master and slave interfaces. An arbitrary amount of layouts can exist, allowing to group interfaces not only by those properties, but also by the level in which associated objects lie in a multigrid hierarchy (both algebraic and geometric multigrid hierarchies). This allows to construct algorithms and solvers like smoothers or exact solvers, which only operate on a given level of a hierarchy. This again is crucial to guarantee good scalability in multigrid methods.

InterfaceCommunicator: This class performs the actual communication between distributed objects. Data exchange can be scheduled for separate interfaces or for complete layouts. It is notable that an interface can be used multiple times to schedule data in a single communication step. Data is not transferred to other processes until the method communicate has been called. During this method data is collected using communication policies, is transferred to the processes associated with the interfaces and is then extracted again using FK.

CommunicationPolicy: Through the concept of communication policies it is possible to adjust the data collection and extraction process to the structures and data-types used in the concrete application or library. Data is written and read from binary streams, which are supplied by a Communicator. This allows to reuse buffers in different communication steps, further minimizing the introduced overhead

ProcessCommunicator: This communicator allows to perform communication between processes (not on individual distributed objects). It contains methods to broadcast data, to gather and scatter data, to send and receive raw data and more. Process communicators are associated with a group of processes, to which the communication is restricted. They can e. g. be used to distribute grids and matrices between processes or to check whether all processes reached a given approximation threshold during the solution of a PDE.

8.3.3 Implementation Details

Since the pcl is a lightweight layer, and since only the InterfaceCommunicator and the ProcessCommunicator perform communication, it should be rather easy to supply implementations for different platforms and message-passing libraries.

Until now only an MPI /GRO 99/ based implementation exists. Due to the broad availability of MPI on cluster-systems, this should however fulfill the requirements of most users. Furthermore template classes are provided, which implement the concepts of an Interface and of a Layout. Those implementations are very flexible and can be adjusted to arbitrary structures and element-containers through template parameters and type traits. It is of course also possible to implement and use custom interfaces and layouts from scratch, as long as they implement the associated concepts.

Please note that no special object type has been defined above. Indeed, this is not required. The distributed objects are supplied directly by the user of the **pcl**. Two examples are given to illustrate this:

- The grid manager stores pointers to the grids elements (vertices, edges, faces and volumes) in the interfaces, allowing to access those elements during data communication without redirection through handles or lookup tables.
- The algebra library stores vector-indices in the interfaces. Algorithms that e. g. make vectors consistent can thus be implemented with minimal runtime overhead.

It may be interesting to point out that the concepts given above do not contain any form of node- or interface-attributes, like e. g. master- or slave-nodes. Indeed this is not necessary since the user is free to associate any kind of attribute with different layouts. Since there are no restrictions on the number of layouts, each application can freely construct layouts representing connections between objects associated by arbitrary attributes. Some examples are given below:

8.3.3.1 Parallel Multigrid

In the simulation environment the parallel multigrid was chosen to distribute the multigrid hierarchies with a low-dimensional horizontal overlap, e. g. when distributing a triangular grid, each triangle will exist only on one process, edges and vertices however share copies between several processes. Due to the hierarchical structure of the grids it was however also important to allow cuts between parents and children in different levels of the hierarchy. Thus also vertical interfaces and layouts, which allow to communicate data during prolongation and restriction, were introduced. For vertical layouts however a full-dimensional overlap is required, since data associated with all elements has to be prolongated/restricted during a multi-grid cycle.

In order to define a clear communication structure the following layout-types for the implementation of the parallel multi-grid were introduced:

- horizontal master layouts,
- horizontal slave layouts,
- vertical master layouts,
- vertical slave layouts.

Separate horizontal layouts exist for the vertices and edges – and in 3d faces – of a grid, separate vertical layouts exist for vertices, edges, faces and volumes.

In Fig. 8.10 an example is given on how the horizontal interfaces of a distributed triangular grid look like. To keep things simple only the interfaces for the grid vertices on a single level are depicted.

Note that while a slave node on each process is a member of exactly one slave interface, master nodes may be shared between several master interfaces. Also note that a node is either in a master or in a slave interface or in no interface at all. Master and slave interfaces are gathered in master- / slave-layouts on each process, which simplifies communication.

In Fig. 8.11 horizontal and vertical interfaces for a one dimensional hierarchical grid distributed on two processes are depicted.

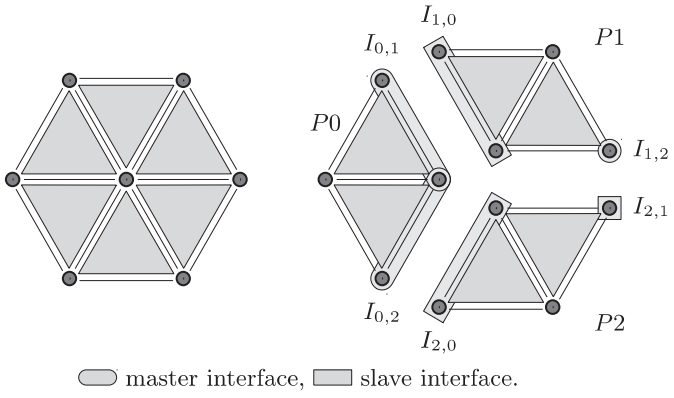


Fig. 8.10 Horizontal interfaces $\boldsymbol{I_{AB}}$ for the vertices of a distributed grid on three processes $\boldsymbol{P0}, \boldsymbol{P1}, \boldsymbol{P2}$. On the left hand the original serial grid is depicted

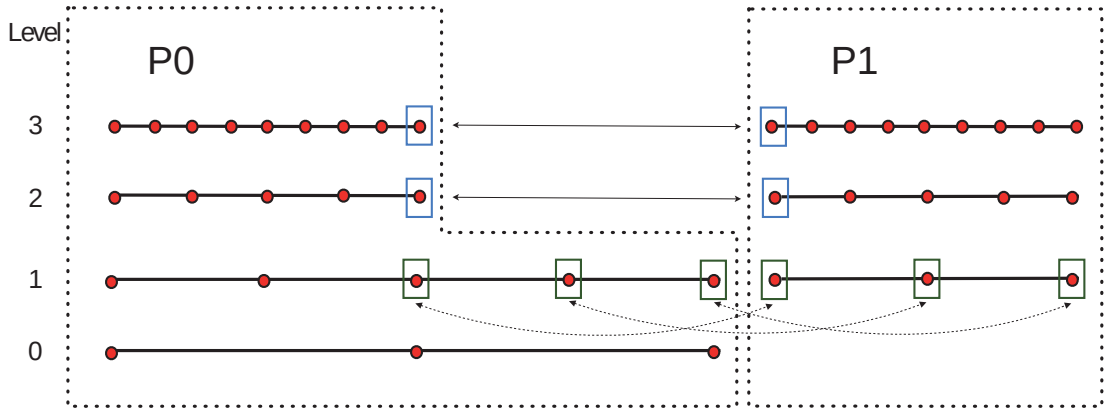


Fig. 8.11 Exemplary distribution of a 1d grid onto 2 processes
Starting with an initial grid on process 0, the grid may be refined until it is distributed to other processes. This introduces vertical interfaces (dashed arrows), that identify identical objects that are stored by several copies on both processes. Further refinement produces a grid hierarchy on each process. The arrows indicate horizontal interfaces.

It was decided here that slave objects in a grid should not be directly connected with each other through interfaces. While this would be possible using the pcl – and indeed is used in some solver implementations - there was decided against it for the grid parallelization. First of all those additional interfaces are not required for tasks like adaptive refinement or dynamic redistribution and would only introduce unnecessary maintenance. Secondly one can exchange data between slave nodes by first collecting data in the master nodes and distributing them to associated slaves afterwards. The two communication steps involved in this scenario are often preferable to direct slave to slave communication, since this would require even more data communication (each slave would communicate with all other associated slaves, which results in $(n-1)^2$ copy operations instead of $2n$ in the implementation, where n is the number of associated slaves). Note that the parallel layout defined on the grid elements can be viewed as minimal, since it only contains interfaces required to identify connected components on different processes. Starting from this minimal layout one can construct other layouts, which e. g. contain overlapping sections or slave to slave connections. This is e. g. used to construct interfaces for the parallel algebra module, which is described in more detail in section 8.3.3.2.

Note that other implementations, e. g. the old parallelization in UG using DDD, feature a similar structure, however, with subtle differences. While UG previously was using horizontal and vertical interfaces, too, the full dimensional overlap, the tight algebra to grid coupling and other restrictions of the underlying parallelization library, lead to the introduction of a variety of additional interfaces, leading to a more rigid setup than this approach. With the separation of components (grid, discretization and algebra) and the lightweight nature of the **pcl**, the implementation of different parallel algorithms is more flexible now.

8.3.3.2 Parallel Algebra

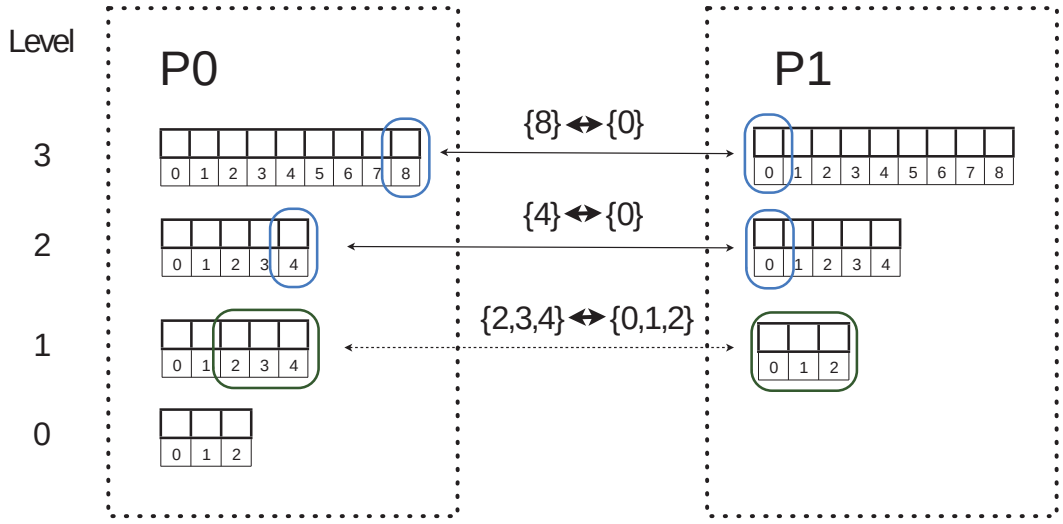


Fig. 8.12 1d Algebra hierarchy

Exemplary distribution of algebra vectors onto 2 processes arising from a linear ansatz function on the 1d grid in Fig. 8.11. Each process stores a usual vector for each grid level, that has the size of the number of nodes on the level. In addition interfaces between the processes are created, that associate identified values. This is shown by the arrows, with the actual identification written above. Note, that again horizontal and vertical interfaces are created.

The pcl has been used to parallelize the algebra structures of the simulation environment. The basic idea of the parallel algebra structures is to store a serial object (e. g. an algebraic vector) on each process and then enrich the serial object with IndexLayouts, that handle the information about those degrees of freedom that are stored on

more than one process. The Interfaces store in a vector-like structure the process-local indices, that are connected to other processes. This allows fast iteration over the indices, that are connected to another process.

Usually the degrees of freedom used in the algebra are somehow associated with the geometric objects of the grid that is used for the simulations. Since for those grid objects the layouts are already created, the algebra layouts can be build up using the information in the grid layouts. However, it is important to note, that the algebra layouts are completely independent from the grid layouts. Thus, if needed they can be build up without grid information or the grid information can be dropped once the algebra layouts have been created. This is essential to create a stand-alone parallel algebra environment and is very useful when purely algebraic sol,vers like algebraic multi-grid solvers, are intended to be implemented, since they can change or build up the interfaces separately.

Using the interfaces the usual operations on parallely stored vectors can be performed /WIE 99/. Each process simply writes the vector values for all indices of its interface into the binary stream. These values are sent to the associated process where the values are unpacked and set, added or subtracted to the process-local indices in the interface of the receiving process. Using such algorithms the storage type of a parallel vector can be changed between

- consistent (each process has the true value)
- additive (the sum over all process returns the true value)
- unique (true value in the master, but zero in the slave).

The chosen setup of the interfaces is very suitable for the implementation of a multigrid algorithm. The communication during the smoothing steps on each grid level is performed using the horizontal interfaces. The restriction and prolongation of the data may include communication when a layer is reached where the grid has been distributed. Here, the vertical interfaces are used.

8.3.4 Scaling Results

As detailed above, the parallel communication layer has been implemented as a lightweight and highly adjustable template library. Special attention has been payed to keep

the overhead required for parallelization as low as possible, while still giving users a library that vastly eases the creation of efficient parallel algorithms in the scope of a simulation framework.

Tab. 8.5 Strong Scaling of the laplace problem, regular 8x8 quads coarse grid, 8 refinements, 4,198,401 DoFs

PE	DoFs/PE	Grid Refinement	Assembling	MultiGrid Solver
8	525,737	5.96 s	1.35 s	5.97 s
16	263,169	2.95 s	0.69 s	2.90 s
32	131,797	1.45 s	0.34 s	1.42 s
64	66,049	0.70 s	0.17 s	0.70 s

In order to test the implementation the result of a simple test problem is shown. The Laplace equation on the unit square is used

$$-\Delta u = f \qquad on\ [0,1]^2 \tag{8.57}$$

and a regular grid of quadrilaterals. The equation is discretized using the vertex-centered finite volume scheme.

Tab. 8.5 shows the result of the strong scaling up to 64 processes. The assembling of the stiffness matrix is trivially parallelizable by adding contributions of each element to the process-local matrix and the measured scaling is optimal as expected. The same optimal scaling is found for the grid refinement and the application of the multigrid solver and shows that the pcl can be used efficiently on these processor numbers.

Tab. 8.6 Weak Scaling of the laplace problem, regular 8x8 quads coarse grid, 131.841 DoFs per PE

PE	Level	Grid Refinement	Assembling	MultiGrid Solver
32	8	1.30 s	0.29 s	1.42 s
128	9	1.33 s	0.29 s	1.43 s
512	10	1.31 s	0.29 s	1.55 s
2048	11	1.34 s	0.33 s	1.67 s

Tab. 8.6 shows the results for the weak scaling of the test problem up to 2048 processes, that has been performed on the NEC Nehalem Cluster at the HLRS, Stuttgart. The computations started with a 8x8 coarse grid and refined to the desired level as

indicated in the table. The measurements show that the *pcl* can be efficiently used to address such numbers of processes.

9 Preprocessing

9.1 Introduction

Grid construction for domains with thin fractures is a complex task. Several steps are required: First, a representation of the boundary of the regarded domain must be specified, then a description of all relevant fractures in the domain has to be given. Intersections between different fractures and between fractures and the domains boundary have to be calculated. Since intersection algorithms typically produce geometries with bad element aspect ratios, the shapes of the triangles have to be optimized by inserting new vertices and by moving old ones. After that the volume grid representing the medium itself has to be created. Special care has to be taken during this step to exactly preserve the specified fractures and boundaries. After the volume geometry has been created, finally the geometry at the fractures has to be adjusted, since additional degrees of freedom are required along the fractures by the used discretization scheme, as described in section 5.5.1.

Since each algorithm requires its own special set of parameters and since most of them can be fine-tuned to achieve optimal results, the tools were implemented as an extension to the general-purpose meshing application “ProMesh”. The tools and algorithms already available in ProMesh together with the newly developed specialized algorithms for fracture expansion and volume grid generation allowed the creation of a flexible pipeline, ready for geometries of vastly differing types.

While complex 3 dimensional simulations are the main goal of the project, the understanding, which can be obtained from a 2 dimensional simulation, should not be underestimated. It was thus important to support both the generation of 2d and 3d geometries, optimally with a very similar set of tools.

9.2 ProMesh

ProMesh is a cross platform tool for the creation, manipulation and optimization of 2 and 3 dimensional volume geometries, developed at the Goethe Center for Scientific Computing at the University of Frankfurt. Its user interface allows to visualize geometries, to select different parts and elements of those geometries and to apply a variety of algorithms on the whole geometry or on selected parts. Each geometry in ProMesh

can be divided in separate subsets, which can be used during simulation to define regions with different parameters or different boundary sections. A short overview over the graphical user interface is given in Fig. 9.1.

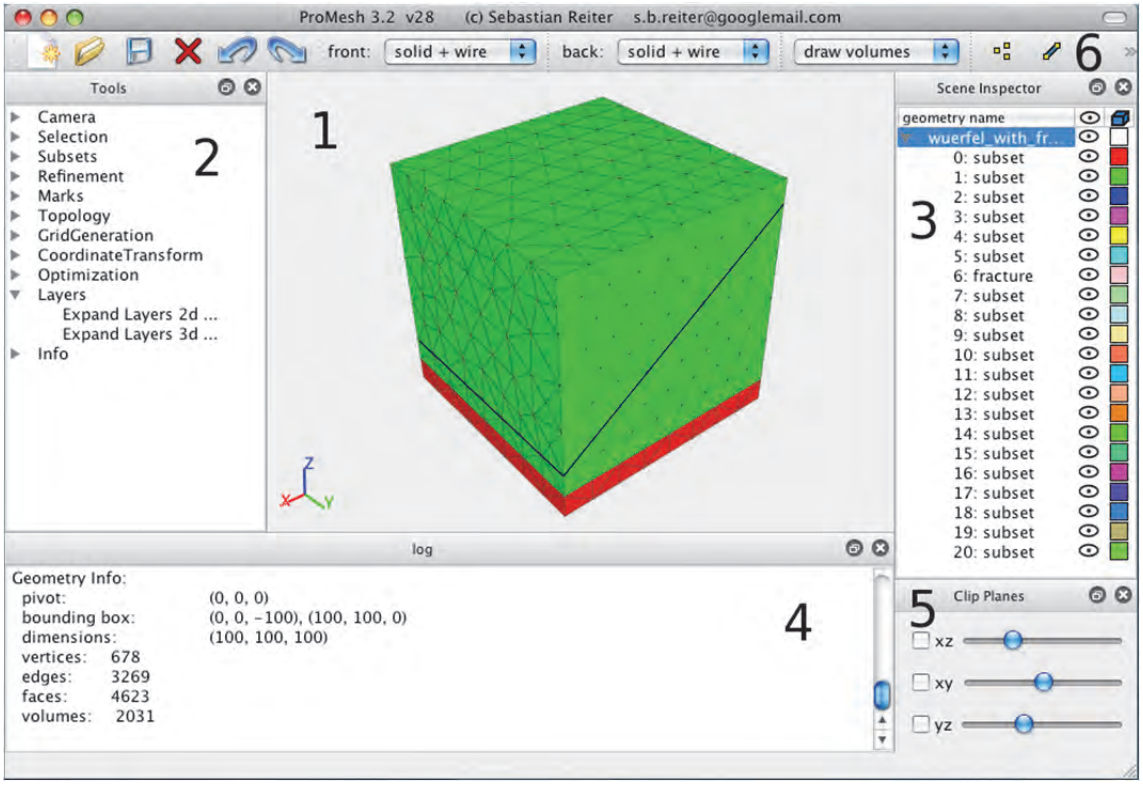


Fig. 9.1 ProMesh: A cube with two volume-subsets, divided by a fracture
(1): 3d view, (2): tools, (3): scene inspector, (4): log, (5): clip planes, (6) tool bar

In the 3d view (1) the currently loaded geometries are displayed. The displayed geometries can be rotated, scaled and moved by dragging the mouse in the 3d view. Vertices, edges, faces and volume-elements can be selected by mouse clicks. In the tools section (2) one can find a variety of different algorithms, gathered in different groups. The “Selection” group e. g. contains algorithms, which transform the current selection (e. g. extending or inverting it). The “Refinement” group contains algorithms to adaptively refine parts of the grid and the “Layers” group contains the expansion algorithms, which are used to prepare the fractures for simulation. The “Scene Inspector” (3) gives a list of all subsets (vertex, edge, face and volume subsets). It allows to choose a color for each subset and to hide different subsets. The “Log” window (4) is used to inform the user over currently executed tasks or to print specific information on current geometries. The “Clip Planes” window (5) is especially useful in 3d, where it allows to cut the

geometry along different planes, thus allowing to closer inspect the structure of the interior volume grid. The main tool bar (6) finally gives quick access to render options and holds shortcuts for commonly used actions (e. g. load, save, …).

9.3 Grid generation

As mentioned in the introduction, the task of constructing a geometry for the simulation of transport in fractured media consists of multiple steps. Some of those steps can be fully automated, others require input and guidance from a user.

First, the specification of the boundary and low dimensional fracture geometries is described, followed by a description how occurring intersections can be resolved. The operations involved in optimizing the shapes of the triangles of the grid are explained. The algorithms behind volume-geometry generation are examined. Finally will be described in detail, how the required fracture expansion can be realized.

9.3.1 Boundary and fracture descriptions

Boundary and fracture geometries can have a lot of different sources. If one wants to construct a simple example, he can construct such a geometry directly in ProMesh. This is most commonly done by first creating a plane or a box through one of the “Grid Generation” tools, followed by coordinate transformations, as well as refine and extrude operations.

Quite often boundary and fracture descriptions are encountered, which are specified as part of a problem description. Those geometries are then often stored in common formats such as the stl-format or the wavefront-obj-format. ProMesh can import those files, which gives the possibility to directly work on geometries, which were specified from external sources.

Since those formats are very common, they are also supported by many commercial and open-source modeling applications, like Blender, 3DS Max or Maya. Nearly all CAD programs also support an export to at least the stl-format. This again allows users to specify boundary and fracture descriptions in almost any tool they like, given that it can export its data to one of those common formats (or to a format which is convertible to “.stl” or “.obj”).

9.3.2 Resolving intersections

While the geometries from 9.3.1 describe the boundaries and fractures, they often do not contain information on how the fractures and boundaries are connected – i. e. intersections of the different fractures and of fractures with the boundary are not resolved in the grid. This however is required in order to guarantee a correct interconnection between all parts of a domain. A bunch of algorithms has thus been implemented, which assist in resolving those intersections. An automated version of the algorithm exists, however, in complicated geometries this algorithm would have to guess the desired result – manual guidance is thus preferable in such cases. The main tasks performed to resolve intersections are the following:

1. **Remove doubles:** Joins all vertices, which are closer to each other than a given threshold. This is especially important for geometries imported from ".stl" format, where each triangle defines its own vertices, leading to multiple vertices sharing one coordinate.
2. **Project vertices to close edges:** Projects vertices to edges, which are closer to the vertex than a given threshold. The edge is splitted and the vertex is used as endpoint for the newly created edges.
3. **Project vertices to close faces:** Projects vertices to faces, which are closer to the vertex than a given threshold. The face is splitted and the vertex is used as a corner for the newly created faces. Note – if this operation is executed after the vertices have been projected to close edges, then degenerate cases can be avoided.
4. **Resolve edge intersections:** This operation is mainly used for 2d geometries. Edges intersecting each other are splitted and a new vertex is introduced at the crossing. In most cases this step is obsolete in 3d, since edges are intersected automatically during step 5 ("resolve triangle intersections").
5. **Resolve triangle intersections:** Calculates intersections between triangles. New vertices, edges and faces are introduced as required. The algorithm is quite involved, since it requires a local retriangulation of intersecting triangles. However, this technique is by far more robust than other comparable techniques, like e. g. repeated edge splits. Note that if this algorithm is executed, step 4 ("resolve edge intersections") can in most cases be skipped.

   For optimal performance an octree is used, to quickly find triangles, which are close to each other.

Note that all of these algorithms are restricted to the set of currently selected elements in a given grid. This allows to further improve performance and to restrict all operations as required. Selecting all elements of a grid is of course valid too.

After all those operations have been carefully executed, one should obtain a regular grid, which does no longer contain unresolved intersections. However, the intersection algorithms often produce triangles of bad aspect ratios, since they try to only introduce a minimal number of new elements. Since elements of a good shape quality are required for simulation, and since the shapes of the fracture and boundary geometries have to be preserved, an optimization must be performed over the shapes of all triangles.

9.3.3 Triangle-grid optimization

In order to improve the aspect ratios of the triangles, one of the operations listed below is repeatedly performed on randomly chosen edges. An operation is thereby only performed, if it leads to an improvement of the aspect ratio of neighbored triangles.

The operations, which are repeatedly applied to different edges in order to improve the triangular grid, are the following (see Fig. 9.2):

- **Edge split:** An edge is splitted by inserting a new vertex. This affects adjacent triangles, which have to be splitted along that edge, too.
- **Edge swap:** Given two triangles (a, b, c) and (b, c, d), the common edge (b, c) can be swapped by removing the two triangles and by introducing new triangles (a, b, d) and (a, c, d), with a common edge (a, d).
- **Edge collapse:** During an edge collapse of an edge e, the adjacent triangles of e and e itself are removed from the grid. The two endpoints of e are merged. This leads to a reduction of the number of vertices, edges and faces.

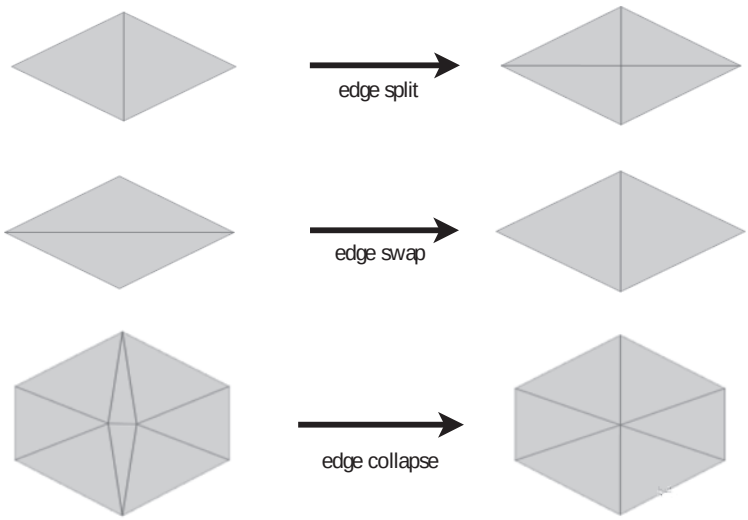


Fig. 9.2 Edge operations: Edge swap, edge split and edge collapse

While the edge operations are important to improve the topology of the grid, coordinate smoothing is used to further improve regularity of the triangles, analogous to /FRE 98/.

In Fig. 9.3 an example is given, that shows how the optimization algorithm improves the aspect ratio of the triangles in a grid.

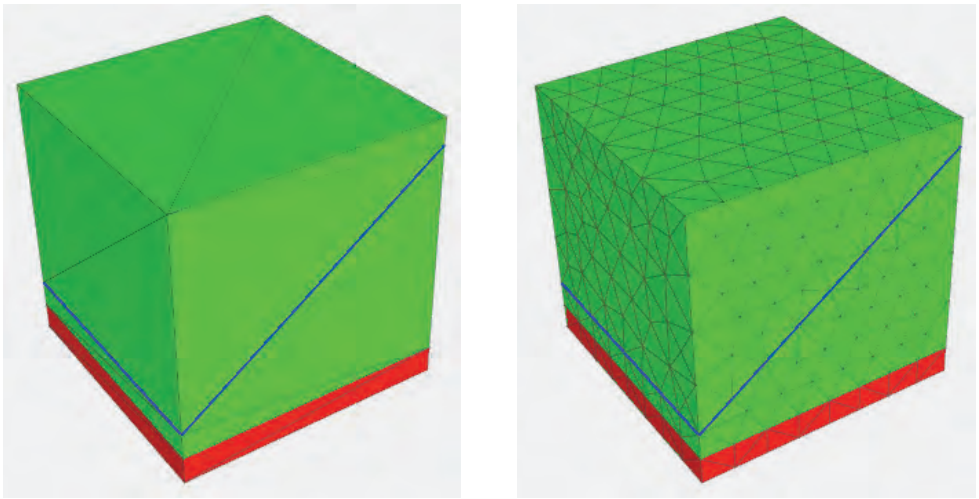


Fig. 9.3 The optimization algorithm applied to a cube with an internal fracture (blue)

9.3.4 Volume geometry generation

Now that the boundary and fracture geometries are fully prepared, it is time to construct the volume geometry.

In 2d a triangles grid is constructed using the **sweep line** method described in /BER 00/. This method preserves edges in the interior of a domain and is thus well suited. Mesh-optimization as detailed in section 9.3.3 is then used to further improve the aspect ratios of the resulting triangles.

In 3d a tetrahedral net has to be constructed, which preserves all given fractures and boundaries. For this task the ARTE grid generator was extended (see /FUC 01/) to support nested geometries. The ARTE grid generator is used, since it was carefully build to preserve all given geometrical structures, while still building tetrahedrons with a good aspect ratio.

During creation of the volume grid, one has to weight between the quality and the coarseness of the resulting grid. Since the geometric multigrid solver benefits from a coarse base grid, coarseness is desirable. However, a coarse representation of complex geometries can often only be achieved by using elements with a bad aspect ratio. This again has a negative influence on the convergence of the solver. Thus a balance has to be stricken between those counteracting goals – at least if we're using the geometric multigrid approach. Another promising approach is the use of algebraic multigrid solvers. Those solvers use a fine grid to assemble a matrix and perform matrix coarsening afterwards to construct a hierarchy of matrices, which is suited for a multilevel method. However, since algebraic multigrid methods are not easy to set up, the geometric multigrid method is still important and grids should not be constructed unnecessarily fine. In Fig. 9.4 two cuts of a cubic geometry with an inner fracture are shown.

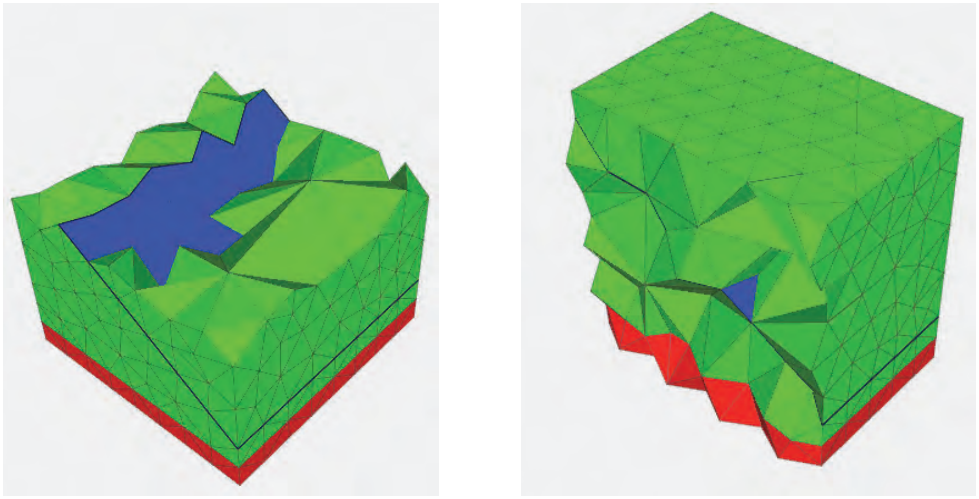


Fig. 9.4 Cuts of a cubic fractured geometry consisting of tetrahedrons

9.3.5 Fracture expansion

At this point one obtained a grid, which is in principle ready for simulation. However, the special properties required by the discretization scheme, as described in section 5.5.1, require the generation of additional degrees of freedom along the fractures. In this section an algorithm is detailed, which will generate those degrees of freedom by constructing additional flat elements.

The grid resulting from the steps above shall be denoted by $\mathcal{G} \coloneqq \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} \coloneqq \{v_1, \dots, v_{N_v}\}$ denotes the set of vertices and $\mathcal{E} \coloneqq \{\mathfrak{e}_1, \dots, \mathfrak{e}_{N_e}\}$ the set of d-dimensional elements in $\mathcal{G}$, N_v and N_e being the number of vertices and elements of $\mathcal{G}$. In this context an element $\mathfrak{e} \in \mathcal{E}$ is identified with a tuple of vertices which describe the corners of $\mathfrak{e}$. The following notation is used: $\mathfrak{e} \in \mathcal{E} \Rightarrow \exists v_{i_1}, \dots, v_{i_{d+1}} \in \mathcal{V}: \mathfrak{e} = [v_{i_1}, \dots, v_{i_{d+1}}]$. By $p: \mathcal{V} \to \Omega \subset \mathbb{R}^d$ the mapping is denoteed that associates a vertex with its position in $\mathbb{R}^d$ and by $\mathrm{conv}: \mathcal{E} \to \mathbf{T}_\Omega$ the mapping that associates an element $\mathfrak{e} \in \mathcal{E}$ with its convex hull. For the $(d-1)$-dimensional elements in the fracture analogous notation is used.

Let $\mathcal{G}_\mathcal{S} \coloneqq \{\mathcal{V}_\mathcal{S}, \mathcal{E}_\mathcal{S}\}$ with $\mathcal{V}_\mathcal{S} \coloneqq \{v \in \mathcal{V}: p(v) \in \mathcal{S}\}$ and $\mathcal{E}_\mathcal{S} \coloneqq \{[v_{i_1}, \dots, v_{i_e}]: v_{i_j} \in \mathcal{V}, \exists \mathfrak{e} \in \mathcal{E}: p(v_{i_j}) \in \mathrm{conv}(\mathfrak{e}) \cap \mathcal{S}, j = 1, \dots, d\}$ be the grid subset representing the fractures.

Our discretization requires additional degrees of freedom at the fracture vertices. An additional vertex is created for each additional degree of freedom required. In order to associate them with the surrounding grid, degenerated elements are introduced – elements with vertices having pairwise the same position so that the elements have zero width. In two dimensions, these are e. g. quadrilaterals resembling an edge, and in three dimensions prisms and pyramids are used that degenerate to a triangle. Note that, since additional degrees of freedom are necessary on both sides of the fracture, the fracture has to be expanded into two layers of degenerated elements (cf. Fig. 9.5).

One could think of constructing the degenerated elements by iterating over all fracture elements $\mathfrak{s} \in \mathcal{E}_\mathcal{S}$, creating degenerated elements on both sides of each $\mathfrak{s}$. However, this approach leads to complications at fracture junctions, where special cases have to be implemented depending on the number of intersecting fractures. This becomes even more evident in three dimensions, where a multitude of special cases would be required to handle the different intersection constellations.

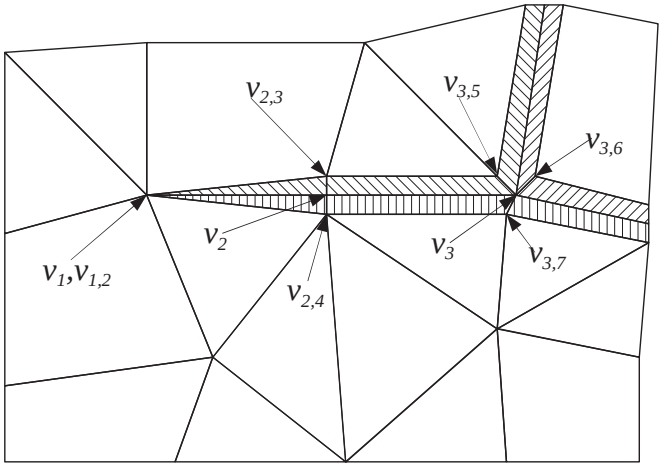


Fig. 9.5 Topological scheme of the degenerated elements and additional vertices in the fracture

Rather is operated on the elements adjacent to the fracture. To this end, for each $v_i \in \mathcal{V}_S$ and all the associated subsets $B_{i_1}, \dots, B_{i_k}$ such that $p(v_i)$ lies in the closure of B_{i_j} $(1 \le i_j \le N,\ j = 1, \dots, k,\ i_1 < \cdots < i_k)$, new vertices v_{i,i_j} are introduced at position $p\left(v_{i,i_j}\right) = p(v_i)$. Let then $\mathcal{V}^*$ be the set of those new vertices.

For each element $\mathfrak{e} \in \mathcal{E}$ that shares a side with a fracture, consider each side $\mathfrak{s} \in \mathcal{E}_S$ of $\mathfrak{e}$, so that $\mathrm{conv}(\mathfrak{s}) \subset \mathrm{conv}(\mathfrak{e})$. Let $v_{i_1}, \dots, v_{i_d} \in \mathcal{V}_S$, $v_{i_1}, \dots, v_{i_d} \in \mathfrak{s}$ be the corners of side $\mathfrak{s}$. Let $B_{j_1}, \dots, B_{j_d}$ be the subsets for which $B_{j_l} \cap \mathrm{conv}(\mathfrak{e}) \neq \emptyset$ and $p(v_{i_l})$ lies in the closure of B_{j_l}, $l = 1, \dots, d$. Then, a degenerated element is constructed from the vertices v_{i_l} and the newly inserted vertices v_{i_l,j_l}, $l = 1, \dots, d$. Let $\mathcal{E}^*$ be the set of all those new degenerated elements.

Finally the fracture neighboring elements have to be reconnected in such a way that they use the new vertices. Let $\mathcal{E}^{\mathcal{N}} \coloneqq \{\mathfrak{e} \in \mathcal{E} : \exists v \in \mathcal{V}_S : v \in \mathfrak{e}\}$ be the set of all elements that share a vertex with the fracture. For every $\mathfrak{e} \in \mathcal{E}^{\mathcal{N}}$, a new element $\mathfrak{e}'$ is created by replacing in $\mathfrak{e}$ each vertex $v_i \in \mathcal{V}$ such that $v_i \in \mathfrak{e} \cap \mathcal{V}_S$, with the associated vertex $v_{i,j} \in \mathcal{V}^*$, where the closure of B_j contains $p(v_i)$, $B_j \cap \mathrm{conv}(\mathfrak{e}) \neq \emptyset$. Let $\mathcal{E}'$ be the set of all those new elements $\mathfrak{e}'$.

By merging and reordering the vertices in $\mathcal{V}$ and $\mathcal{V}^*$ and by using the elements from $\mathcal{E}$ and $\mathcal{E}'$, one obtains the sets $\Omega_h \coloneqq \{p(v) \in \Omega : v \in \mathcal{V} \cup \mathcal{V}^*\}$ and $T_\Omega \coloneqq \left\{\mathrm{conv}(\mathfrak{e}) \subset \Omega : \mathfrak{e} \in \mathcal{E} \cup \mathcal{E}' \setminus \mathcal{E}^{\mathcal{N}}\right\}$ as introduced in section 5.5.1. The resulting grid has all the properties required for simulation.

To keep the special structure of the grid in the fracture during the creation of the grid hierarchy during simulation, anisotropic refinement rules must be used for the degenerated elements. Furthermore, the degenerated elements adjacent over their non-degenerated sides should be refined simultaneously. This maintains the two-layer topology of the grid in the fracture.

9.4 Graphical user interface

9.4.1 Introduction

The automated mapping of program functionality to intuitive user interfaces is a highly challenging task. Nevertheless, it is a promising way to significantly improve software quality by simplifying the development process.

In the last decades increasing computing power made more and more accurate simulations possible. Usually, these kind of simulations require the simultaneous control of numerous input parameters like the full geometry, including material properties, coupling of different physical processes, different grids, numerical methods, simulation software and finally computers.

To handle this complexity it is important to establish a visual and intuitive environment for the user. There are several types of users. The one that is expert in the application problem to be simulated, but not in numerics or computing, is an important one. Other users, however, have, despite of problem specific knowledge, also a lot of expertise in numerics. Thus, a flexible environment is necessary, which allows the set-up of a simulation in close analogy to the decisions and selections the user is taking when approaching the simulation of a model.

To accomplish this task, the **V**isual **R**eflection **L**ibrary (**VRL**) as Framework for automatic graphical user interface (GUI) generation is used. Based on VRL, a visual workflow-management software has been developed to allow the integration of the simulation system **U**nstructured **G**rids (**UG**) and applications that are based on UG, such as the upcoming version of d³f.

9.4.2 Declarative GUI programming

9.4.2.1 Definition

Declarative GUI programming is already used by several technologies such as Qt /NOK 09/ and JavaFX /ORA 10/. Those toolkits introduce specific scripting languages for defining custom GUI elements. While making the development of custom user interfaces more efficient, this approach does not solve the problem that the connection between the non-graphical backend of the application and the frontend (GUI) has to be created manually and changed whenever the application structure changes.

Declarative Programming as used by VRL is focused on the last aspect. A mechanism was created that allows to keep the backend and frontend in synchronization. VRL does not introduce a specific language for GUI development. Rather than that, interactive visualizations of Java /ORA 96/ objects were automatically generate, i. e., their public interface. This totally decouples the GUI generation from the language, i. e., objects from every language that can be accessed via the Java Reflection API can be visualized. Throughout this paper Groovy code /COD 09/ was used unless noted otherwise.

As mentioned before VRL GUI generation is based on the Java Reflection API. There are other tools and frameworks that make use of this information as well to create user interfaces. A common use of this is a property editor as used in development environments such as the Netbeans Integrated Development Environment (IDE) /ORA 10a/ or Eclipse /ECL 11/. But a property editor is optimized for manipulating data and does not give interactive access to the functionality of an object. The BlueJ IDE /KOL 03/ provides interactive access to the functionality of an object. The intention here is to give an introduction to object-oriented programming. This is an attempt to allow visual interaction with Java objects. However, it does not provide a full mapping from functionality to graphical interfaces. These usages of GUI generation via Reflection do not solve the problem of automatically generating high-quality interfaces. The challenge is to create a general-purpose framework for declarative GUI programming.

9.4.2.2 VRL component types

To accomplish this task VRL uses three types of visual components (see Fig. 9.6). An object representation is a container, comparable to a program window that can group several child components. A method representation is a container component inside an object representation. It can also group child components and provides elements for calling the represented method. To represent variable data VRL provides type representations. In most cases they allow interaction with the visualized data.

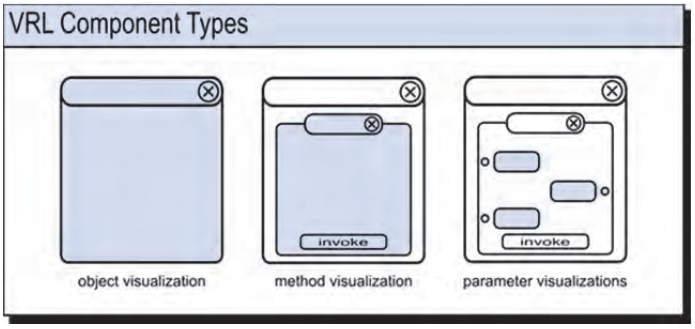


Fig. 9.6 VRL Component Types

9.4.2.3 Object visualization

Fig. 9.7 shows the visualization result for a simple class that provides a method that is capable of adding two integer numbers. Based on the component types of Section it is possible to create interfaces with only a minimal amount of code. To create a graphical user interface it is only necessary to specify the functionality. In this case a GUI is requested that is capable of adding two integer values.

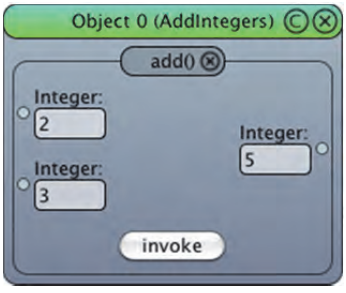


Fig. 9.7 Visualization of a simple Java object

Compared to Java code that does not only implement the functionality but also involves GUI generation, this example shows that declarative GUI development can be very efficient. It has to be mentioned that this example visualization has several extra features such as error messages for incorrect data etc.

```
public class AddIntegers {
  public Integer add(Integer a, Integer b) {
    return a+b
  }
}
```

Fig. 9.8 Code used to create the visualization shown in Fig. 9.7

For a common Java implementation using the Swing /LOY 02/ toolkit much more code is required. A sample is shown in Fig. 9.9. (This is not necessarily the shortest possible implementation.)

```
public class Main {
  public static void main(String[] args) {
    javax.swing.JFrame frame = new javax.swing.JFrame("Add Integers");
    frame.setLayout(new java.awt.GridLayout());
    final javax.swing.JTextField input1 = new javax.swing.JTextField();
    final javax.swing.JTextField input2 = new javax.swing.JTextField();
    final javax.swing.JTextField output = new javax.swing.JTextField();
    frame.add(new javax.swing.JLabel("Integer"));
    frame.add(input1);
    frame.add(new javax.swing.JLabel("Integer"));
    frame.add(input2);
    frame.add(new javax.swing.JLabel("Result"));
    frame.add(output);
    javax.swing.JButton btn = new javax.swing.JButton("invoke");
    btn.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(java.awt.event.ActionEvent e) {
        output.setText(new Integer(new Integer(input1.getText()) +
          new Integer(input2.getText())).toString());
      }
    });
    frame.add(btn);
    frame.pack();
    frame.setVisible(true);
    }
}
```

Fig. 9.9 Compact implementation of a Swing application that is able to add two numbers

Furthermore VRL can do the interface generation without any additional interface related commands. But by defining the problem domain, interfaces can be customized (see section 9.4.2.4). This is one of the major advantages. The VRL user only provides functionality and optionally some details about the problem domain. These details are not commands. They are meta information, i. e., annotations that do not influence the func-

tionality of the given code. The absence of GUI related commands keeps the code simple and clean. Even more important is the fact that the code can be used for other purposes as d³f well. That is, the implemented functionality can be used in any Java program or library without changing the code. For source code that does contain GUI related commands other than annotations this is in often impossible.

Even though architectural patterns such as the Model-View-Control pattern (MVC) /GAM 95/ improve the development process and help to separate the functionality from the graphical interface, it is assumed that the application logic should be completely independent from the user interface and should not be a part of the implementation. Namely, if not automatically generated, the graphical user interface and the application logic will always diverge to some degree and a lot of work has to be done to prevent that. In many projects this results in user interfaces that cannot provide the newest functionality of the application backend. In these cases users have to disregard the graphical user interface and use the backend directly (programmatically). This adds even more features to the backend that are not accessible through the user interface.

Under the assumption that UG provides a service that is able to create such an interface description for its functionality, VRL visualization can be done without an extra implementation. For the upcoming UG version such a service is in preparation. The early tests show that this is a highly promising approach.

9.4.2.4 Domain specific GUI elements

While object representations and method representations are mostly generic elements, type representations are individual components. Their appearance depends on the data type of the visualized variable and the problem domain. The example in Fig. 9.8 from Section 9.4.2.1 is recalled here. The code does not include any GUI related commands. Thus, VRL uses a default type representation for the parameter types. For numbers and strings this is relatively easy.

Although this seems to be a reasonable approach for simple objects, it is not clear that this is true for complex cases as well. Compared to classical GUI development this seems rather inflexible. To overcome those problems VRL supports multiple type representations per data type. By supplying information about the problem domain, i. e., the context of the variable to visualize, it is possible to advise the system to choose

between different type representations. For this example one might want to use a slider instead of an edit field. VRL supports such meta information via parameter annotations. Parameter annotations can be used to request a specific visualization and to define problem specific properties such as value ranges. If the requested visualization is available it will be preferred over the default visualization. Fig. 9.10 shows a customized version of Fig. 9.8. The resulting visualization is shown in Fig. 9.11.

Defining properties of the type representation may be impractical in some cases because the annotations will become rather complicated. In this case it is suggested to define a custom type representation which will be discussed in section 9.4.2.

```
public class AddIntegers {
  public Integer add(
     @ParamInfo(style="slider",options="min=0;max=100") Integer a,
     @ParamInfo(style="slider",options="min=0;max=100") Integer b) {
    return a+b;
  }
}
```

Fig. 9.10 Illustration of parameter annotations

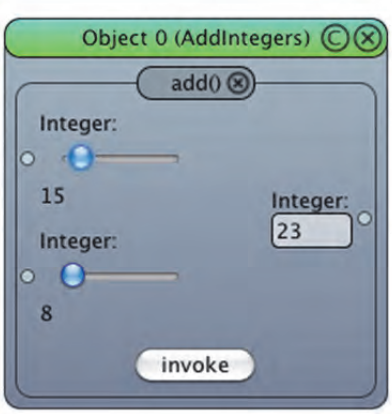


Fig. 9.11 Visualization of a simple Java object using parameter annotations

9.4.2.5 Custom type representations

To extend the number of known problem domains, VRL can be extended by adding custom type representations. Currently type representations for common variable types such as `Integer, Float, Boolean` and `String` exist. In addition it provides interactive type representations for 2D and 3D visualizations. The UG specific extensions enable UG script generation and include MathML /WOR 99/ based rendering for math-

ematical formulas (see Section and). Extensions for modifying surface properties such as boundary conditions are in development.

Technically a type representation is a Swing component that provides additional methods for data processing and visualization including data specific error handling. Defining custom type representations is a problem specific task. Thus, except from basic understanding of the Swing framework, only problem specific knowledge is required.

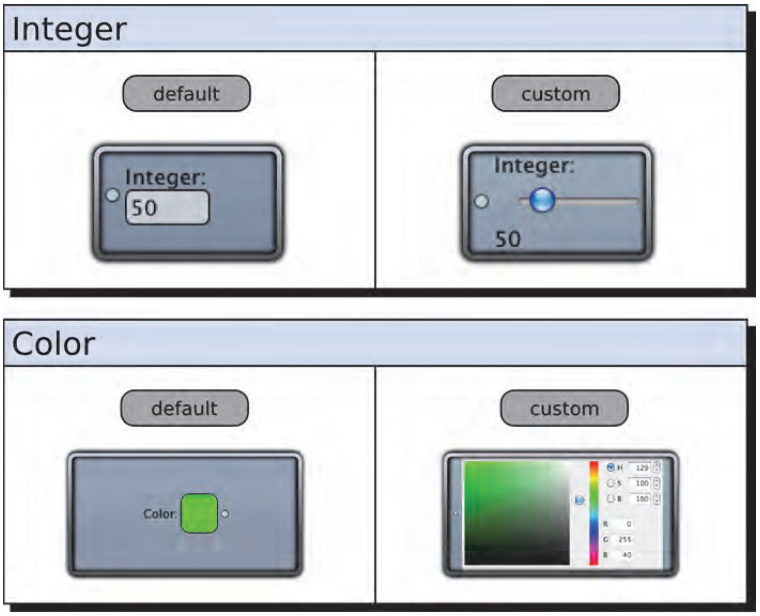


Fig. 9.12 Type representations for `java.lang.Integer` and `java.awt.Color`

An overview of different type representations for `java.lang.Integer` and `java.awt.Color` is shown in Fig. 9.12. Now these features are discussed with the help of an example. Fig. 9.13 shows a sample class wanted to provide a type representation for.

```
class Circle {

    public int radius

    public Circle(Integer radius) {
        this.radius = radius
    }
}
```

Fig. 9.13 Circle Class

If VRL visualizes an object that uses the Circle class from Fig. 9.13, it uses a default type representation that only shows the class name. Defining a custom type representation enables VRL to use an improved visualization. Fig. 9.14 shows a possible implementation of a type representation for the class defined in Fig. 9.13. The constructor defines the class to visualize and defines the name to use (an empty name is used). The `setViewValue( Object o )` method defines how to visualize an object. For every class that uses the `Circle` class in its public interface VRL will use the type representation defined in Fig. 9.14 unless requested otherwise. The `getViewValue()` method can be used to create an object based on the current visualization, e. g., user input. In this case an interactive visualization is not provided. Thus, the value that is currently visualized will be returned. In many cases it is sufficient to implement a constructor, the `setViewValue( Object o )` and `getViewValue()` methods. Thus, no special knowledge of the internal implementation is required.

```
class CircleType extends BufferedImageType {

    public CircleType(){
        setType(Circle.class)
        setValueName(" ")
    }

    public void setViewValue(Object o) {
        def circle = o as Circle
        def image  =
            ImageUtils.createCompatibleImage(300,300)
        def g2 = image.createGraphics()

        g2.setColor(Color.green)
        def thickness = 5
        g2.setStroke(new BasicStroke(thickness))

        int centerX=image.getWidth()/2
        int centerY=image.getHeight()/2

        int x = centerX + thickness/2 - circle.radius
        int y = centerY + thickness/2 - circle.radius

        int width = 2 * circle.radius-thickness/2 - 1
        int height = 2 * circle.radius-thickness/2 - 1

        g2.drawOval(x,y,width,height)
        g2.dispose()

        super.setViewValue(image)
    }

    public Object getViewValue() {
        return this.@value
    }
}
```

Fig. 9.14 Type representation for the Circle class

```
class CircleCreator {

    public Circle createCircle(
        @ParamInfo(name="Radius") Integer radius) {
        return new Circle(radius)
    }
}
```

Fig. 9.15 Class that uses the Circle class in its public interface, listing

Fig. 9.15 shows a class that creates an instance of the circle class and returns it. The corresponding VRL visualization is shown in Fig. 9.16.

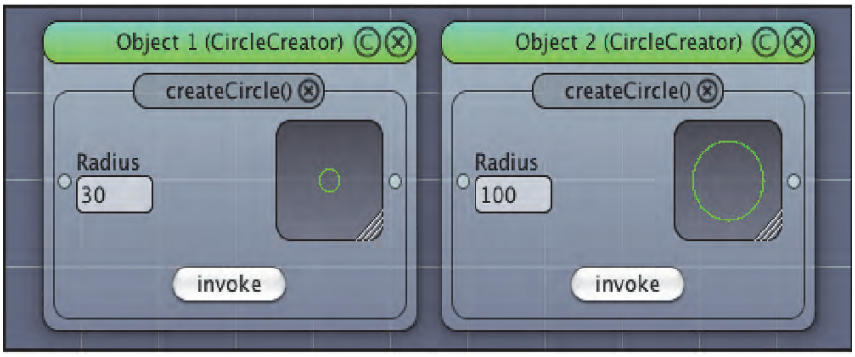


Fig. 9.16 Class that uses the `Circle` class in its public interface

9.4.3 Visual programming

9.4.3.1 Data dependencies

Each visual programming environment needs a method to define data dependencies. Therefore, VRL components can be connected via wires (see Fig. 9.17). Defining dependencies by connecting components is a technique that is used by several tools such as the Visualization Data Explorer from IBM /IBM 91/. Data dependencies are defined by connecting return values and parameters of methods. VRL connections are type-safe. To evaluate data dependencies it is necessary to define a sequence of method calls, i. e., to determine which methods have to be called to compute the requested result.

Interactive type representations notice every value change. All dependent type representations will be emptied to prevent data inconsistencies. The return value type repre-

sentations of all dependent methods are marked as out of date. If the user invokes a method VRL will compute the dependencies and call all required methods.

But, determining the sequence of method calls by evaluating the data dependencies limits the user in several ways. It is not possible to freely define a deterministic algorithm. In common programming languages method call sequences can be defined independently from data dependencies.

9.4.3.2 Codeblocks

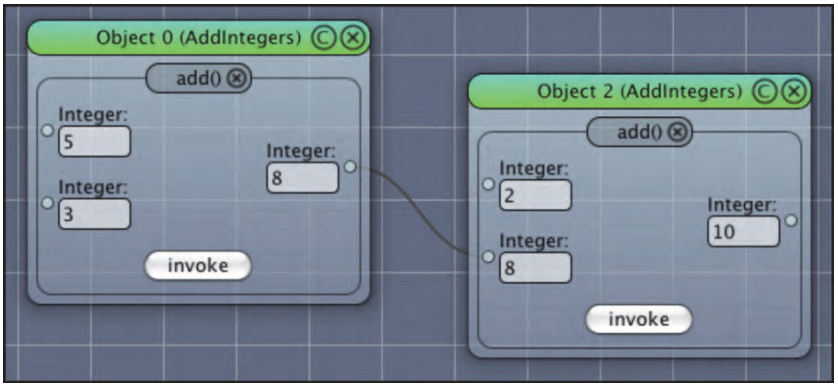


Fig. 9.17 Data dependency between two objects

To enable the feature of defining method call sequences, VRL supports codeblocks. A VRL codeblock is equivalent to a block in C++ or Java. VRL contains code generators that can map a sequence of method calls to Groovy code, see Fig. 9.17. This sequence is defined by selecting method representations via mouse gestures.

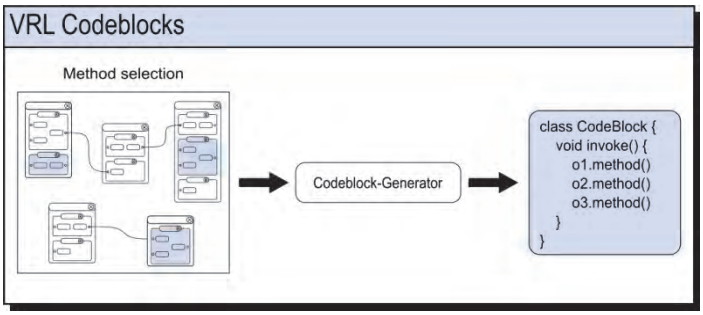


Fig. 9.18 Custom sequence of method calls via codeblocks

The corresponding Java object can be visualized like any other object. Hence, VRL provides mechanisms for defining custom method call sequences. This is important to enable the definition of complex workflows. For UG this feature is essential, as it is necessary to allow custom computation workflows, like the current UG scripting language does.

One problem with the current approach is that it is not easily possible to invoke a method multiple times, each time with different parameters. This issue was addressed by supporting multiple object visualizations. These visualizations can be used to invoke the same method with different parameters.

9.4.3.3 IDE features

As the Java VM allows dynamic class loading /HAL 02/ it is also possible to define the functionality as Java class and visualize it with VRL at run-time. In addition to that VRL provides Groovy support. An integrated editor enables the user to write custom components. Even type representations can be developed at run-time. This allows interactive GUI development and extends the visual programming features. The Groovy code does not have to be added via the editor. It can also be retrieved from a code generator such as the codeblock generator described in section 9.4.3.2. To simplify the development process VRL-Studio was created, a small VRL based IDE. VRL-Studio is also used to create the UG frontend.

9.4.3.4 Persistence

VRL uses a XML based persistence model. It stores canvas properties, objects (instances of classes) and the state of their visualization. The source code of classes defined with the Groovy editor is stored as well. Such a configuration is called a session.

Most VRL based programs are based on a session file. To ensure that the final program can be deployed, VRL sessions can be exported. The exported file contains the session file and the VRL runtime, including external dependencies.

9.4.3.5 Creating applications

9.4.3.5.1 Problem definition

Even though several features that improve the creation of user interfaces have been decribed, the question whether application development is possible is not answered yet. Creating the workflow is done by connecting objects (some of their type representations) and creating codeblocks. But what about the application itself? In many cases it is necessary to create a reduced user interface, designed for a specific purpose. This is what defines the application from a user point of view. VRL provides several features to achieve this (see Sections 9.4.3.5.2 and 9.4.3.5.3).

9.4.3.5.2 Parameter groups

After defining the workflow of an application one usually wants to group the most important parameter visualizations to simplify the user interface. With the methods delineated so far, the only choice is to change the classes that define the application functionality or to add classes specifically designed to group selected parameters.

But in some cases this is either impossible or impractical as one does not want to change the code only to achieve a specific grouping of object parameters. This leads to code that is only used to create the GUI itself. However, this is exactly what shall be avoided. Thus, VRL provides a feature called Parameter Groups. This means that, parameters from object visualizations can be selected and grouped in a separate window. This is shown in Fig. 9.19 and Fig. 9.20.

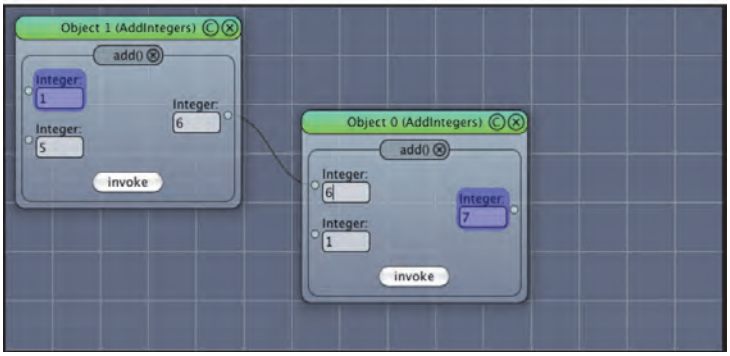


Fig. 9.19 Parameter groups (selection)

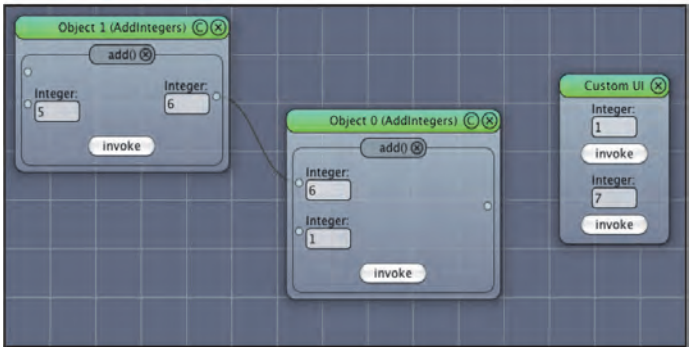


Fig. 9.20 Parameter groups (result)

9.4.3.5.3 Window groups

The problem now is that all the objects are visible, whether important or not. Therefore VRL allows the definition of window groups. A window group defines the location and the visibility of each window that is part of the group. This enables the user to hide all objects that shall not be part of the reduced user interface.

9.4.3.5.4 Example

Combining both features, an application workflow can be simplified significantly. Fig. 9.21 shows a simple function plotter. It provides objects for defining the function that shall be plotted, properties that define the function variables and the visual appearance of the output. Finally it shows the output itself. It is an interactive 3D visualization of the evaluated function.

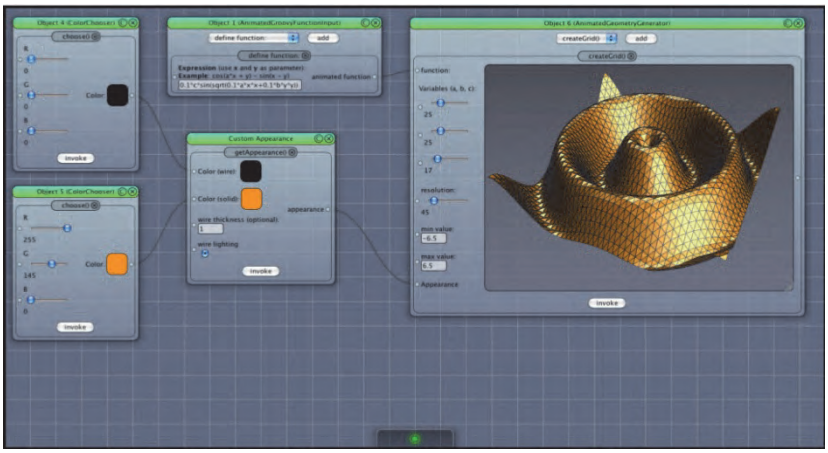


Fig. 9.21 Function plotter

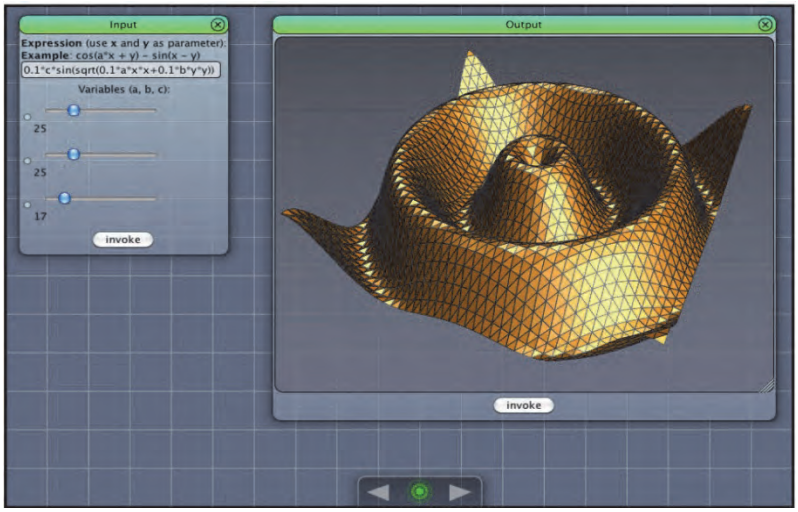


Fig. 9.22 Reduced interface for the function plotter from Fig. 9.21

It is assumed that the only important task is the definition of the function and their parameters. The final application interface is shown in Fig. 9.22.

9.4.3.5.5 Multiple views

As was to be seen in section 9.4.2.4, VRL enables the definition of a task specific view. However, it is also possible to define multiple window groups. This enables the definition of multiple task specific views. In the function-plotter-example this could be a separate view for changing the appearance of the visualization.

9.4.3.5.6 Limitations

Currently the definition of a reduced user interface has some limitations. When grouping parameters it is not possible to choose between different component layouts and a parameter cannot be grouped twice. Because this is an important feature it is planned to integrate advanced layout support in the near future.

9.4.4 VRL applications

Now some real examples are shown to demonstrate the current state of VRL.

9.4.4.1 BasicMath

BasicMath is an extension for VRL. BasicMath provides several mathematical objects such as scalar, vector and matrix and corresponding functions such as vector norm or matrix vector multiplication etc. The mathematical objects are represented as visual instances. BasicMath enables visual formulation of mathematical expression. Object names can be displayed as mathematical expressions, i. e., special characters such as integral, norm are supported.

To render mathematical expressions BasicMath integrates the MathML renderer JEuclid. However, it is not necessary to use MathML code for object names. Plain text is also supported. All functions/operators can automatically create the name of their result. By combining several functions/operators the name of the last result consists of the complete expression. But the result name can be overriden with an arbitrary expression.

Functions can be added from a popup menu to the canvas of VRL-Studio. Objects like e. g. a matrix will be created and added to canvas by a so-called MatrixGenerator. In addition to Functions/Operators and data elements, BasicMath provides so-called generators, one for each element type (matrix, vector, etc.). The usage of generators and functions/operators is illustrated with the help of the following example.

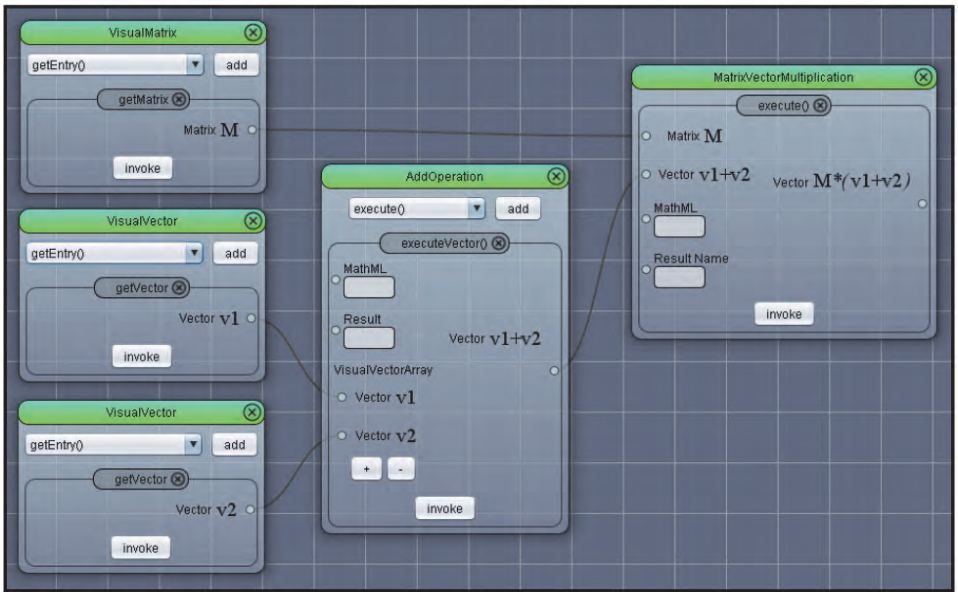


Fig. 9.23 BasicMath Sample Session

After creating two vector objects and one matrix object via corresponding generators the objects are used for calculations. In Fig. 9.23 objects for vector addition and matrix-vector multiplication are used.

In section a component is described that uses the data objects shown in the example to interact with UG. Fig. 9.24 shows a scalar, matrix and a vector object. They are used by the component that interacts with UG.

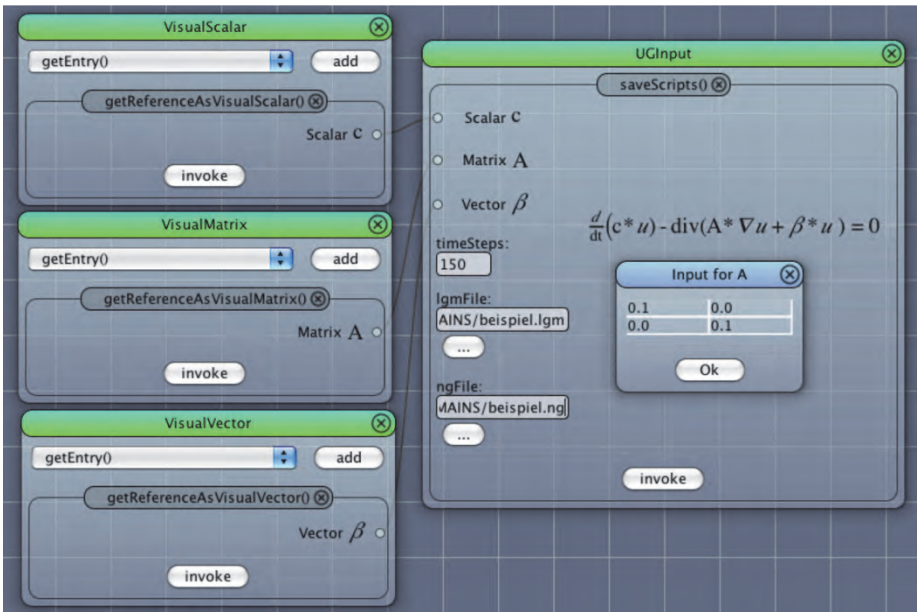


Fig. 9.24 UG input with three BasicMath elements

It is possible to visually access data elements in different ways, e. g., the data of a matrix object can be accessed by a vector. An example is the manipulation of the matrix diagonal. To enable this feature BasicMath uses so called mappings, i. e., bijective mappings between two index sets. Visual changes of the data elements affect all visualizations that use these data elements.

9.4.4.2 UG 3

In the following part of this article it will be shown which graphical components have been created for the current version of UG. Furthermore, their functionality is described and how they can be used to simplify the UG workflow. The assumption was made that a common user does not want to implement the mathematical algorithms himself. That is, the user wants to use the existing functionally to solve a specific problem without

deeper understanding of the workflow internals. The workflow itself is rather static. But the parameters are subject to change and depend on the specific problem. Therefore all components allow to interactively specify selected parameters.

Additional type representations and components allow the visualization and interactive manipulation of mathematical objects such as matrices, vectors and scalars. The existence/availability of these visualizations and corresponding data structures enable the visual formulation of mathematical contents/relations/subjects. This functionality is part of the VRL extension BasicMath.

9.4.4.3 Line of action

9.4.4.3.1 Creating a VRL based graphical frontend

The development process of a VRL module that integrates a specific UG workflow could be classified as follows:

- identify the problem specific parameters
- create custom type representations for special parameter types
- create the Java classes that implement the necessary functionality (will be visualized by VRL)
- create custom script files to store the user-specified parameters
- include the custom script files into the existing UG scripts

9.4.4.3.2 Components for the diffusion-convection-equation

To improve the handling of the UG components for the d iffusion convection equation

$$\frac{d}{dt}(c * u) - div(A * \nabla \mathrm{u} + \beta * \mathrm{u}) = 0$$

and their results special components were created.

a) UGInput

As shown in Fig. 9.24 on the right side UGInput allows to set the parameters of the diffusion convection equation and visualizes the equation with user defined parameter names. The equation is based on automatic generated MathML code. This allows a flexible adaption of the visualization of the equation.

Timesteps and geometry can be set by typing the number respectively the path to the wanted geometry file.

b) LivePlotter

The LivePlotter allows to observe the evolution of the geometry over time during the calculation. Furthermore, the geometry can be freely translated, rotated and zoomed. With this component it was tried to enable the user to evaluate the current solution.

c) SolutionPlotter

As shown in Fig. 9.25 the SolutionPlotter has been created to visualize the evolution of the geometry after the competition is finished. SolutionPlotter allows the same interaction with the geometry like LivePlotter but he can additionally replay the development or visualize the geometry at a specific timestep.

The application area of the SolutionPlotter are e. g. iterative processes where a big interest in the evolution of the geometry and or the visual presentation of it.

LivePlotter was developed to valuate an intensive calculation at nearly real time, if the evolution of the corresponding geometry is known or estimated to be of a special kind. So the calculation can be aborted if the development did not fit the wished progress.

Fig. 9.25 SolutionPlotter component visualizing a calculated geometry

d) PickPlotter

PickPlotter shown in Fig. 9.26 allows different operation states and actions. They are represented by buttons that are shown on the left side of the visualization area.

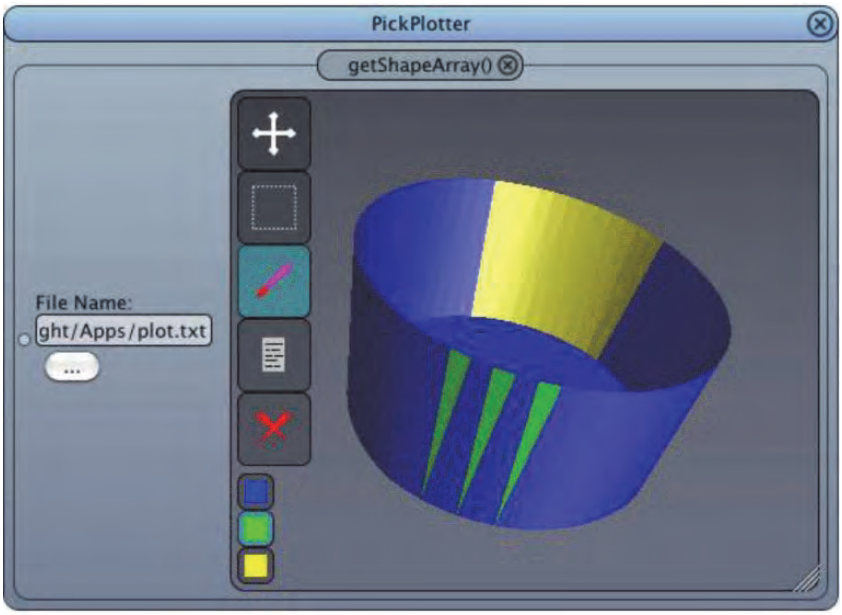


Fig. 9.26 The PickPlotter component showing a geometry file and two selected areas

In the translation-rotation mode it is possible to define a custom rotation point on the geometry surface. When defining the rotation center the geometry will be translated. That is, the rotation point will be moved to the center of the visualization area. Additionally, there is the possibility to define boundary conditions on the visualized geometry, which are also graphically represented. There is one state to select all triangles along the deep axis in a selection rectangle and one state to select only the first visible triangle under the mouse pointer.

The type of the marginal condition can be selected by pressing one of the three smaller buttons on the left lower corner of the visualization area. To notify UG about the selection this component can write the custom information to a script file.

9.4.4.4 d³f

9.4.4.4.1 Automatic registration of functionality with the VRL

UG and d3f provide a broad spectrum of functionality that must be accessible via the VRL user interface. These programs consist of several small logic components (such as discretization, boundary condition, numerical solvers) that must be combinable in many ways to adapt to the specific needs of a single simulation. In order to allow this kind of flexibility it is evident that an automatic generation of visualization as provided by the VRL must be combined with a automatic binding of UG/d3f functionality to the VRL. While Java has native reflection support, this is not true for C/C++. Since UG/d3f is written in C/C++ for performance reasons, a binding software called UGbridge has been developed to furnish UG and d3f with this property. The implementation uses C++-template techniques and provides a typesafe binding.

9.4.4.4.2 Visual components for d³f

In the new version of d³f/UG for the VRL several components have been made available in the graphical interface. These components are:

- Loading and Saving of a physical Domain from file
- Vertex-centered finite volume discretization of the equations of density driven flow
- Dirichlet boundary conditions, Flow boundary conditions

- Setting of physical parameters such as molecular diffusion, gravity, permeability, viscosity and porosity
- Selection of different numerical solvers such as BiCGStab and preconditioners like geometric multi-grid, incomplete Cholesky factorization, Gauss-Seidel
- Output of the computed solution for brine mass fraction and pressure to vtk-file format

In Fig. 9.29 and Fig. 9.30 an example session is presented to illustrate the software. The well established Elder problem /ELD 67/ is computed for a two dimensional domain. In Fig. 9.29 the simplified user interface is shown that is designed to allow users to change several physical parameters of the simulation without going into the details of the numerical simulation. The core of the session is the component "d³f Component" that groups the whole setting of the density driven flow problem solved in this example. Invoking the "filename" button the user can choose a physical domain by a fileselection-dialog as it is shown in Fig. 9.27.

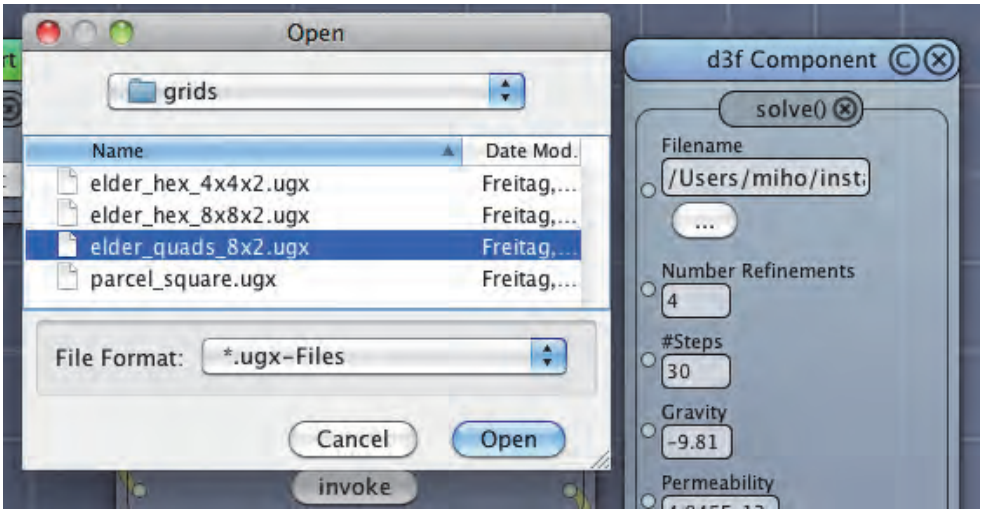


Fig. 9.27 Interactive selection of physical domain

The selection of the physical parameters is shown in Fig. 9.28. The simulation can be started by the "start" button in the upper left corner. Detailed information about the current status of the solution process is displayed in the message box at the bottom (shown in Fig. 9.29). The solution can be viewed in an interactive preview window on the upper right corner. More advanced users have the possibility to control further details of the numerical simulation. In Fig. 9.30 some hidden components of the simulation core are shown that can be opened by a user if needed. In this session, the user

can specify the boundary conditions for pressure and brine mass fraction as well as the initial conditions. Other non-displayed parts of the setting such as the setup of the numerical solvers can be opened and influenced as well.

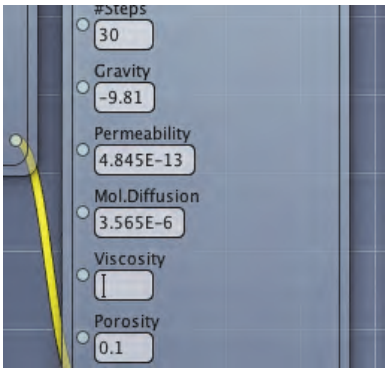


Fig. 9.28 Selection of physical parameters

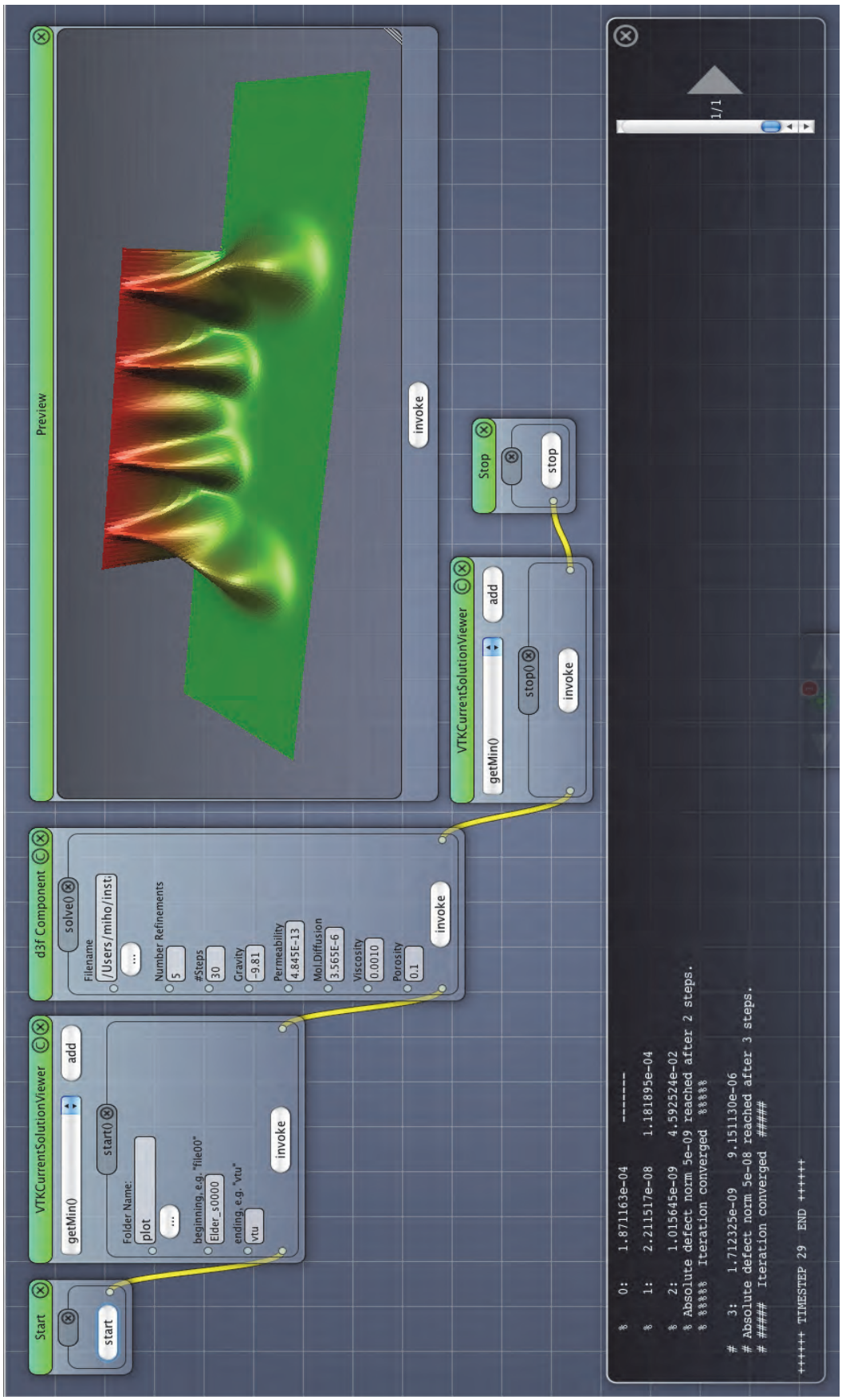


Fig. 9.29 d³f sample application (a)

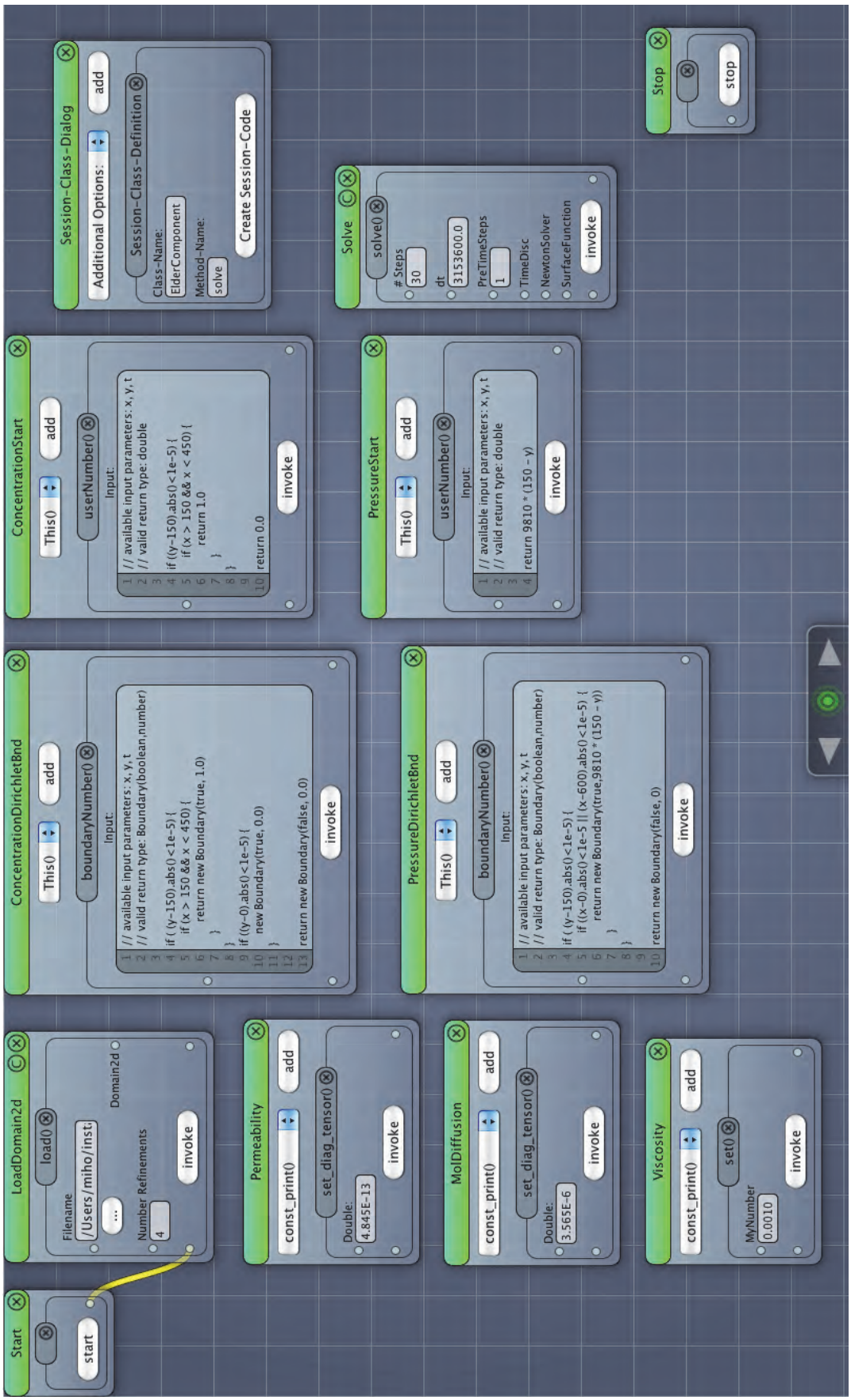


Fig. 9.30 d³f sample application (b)

10 The postprocessor

The post processor is based on the post processing package developed for the software packages d^3f and r^3t. In this project, the key aspects were the adaptation of the data interface and the visualization and data analysis methods dealing with the newly added computational concepts (most notably the addition of fractures) and the extension of existing methods to new requirements and the expansion of the computational framework (such as the representation of free boundaries by level sets, the visualization of sources and sinks, new methods for function evaluation and integration etc.). The most important tasks were:

- **The visualization of data in fractures.** On the one hand, the elements and nodes belonging to a fracture have to be identified by the post processor. On the other hand, the lower dimensional structure must be visualized in a way that facilitates distinguishing it from the visualization of bulk properties.
- **Free and moving, implicitly described boundaries.** Besides the visualization of the boundary itself, whose location is described by a scalar level set function, the primary task is to take into account the free boundary as a description of the domain to be considered when visualizing other quantities.
- **Including sources and sinks in the visualization.** The position and magnitude of the sources and sinks should be represented by appropriate visual markers in the graphical output. The visualization must allow these to be time-dependent.
- **Unifying and extending the probing and integration methods.** The method for probing at points, lines and curves, integrating scalars over lines and sub domains and flows over boundaries and hyper surfaces, both in 2d and 3d, stationary and time-dependent, should be restructured and extended to form a complete set of tools with a unified work flow.

The post processor is based on the library GRAPE (Graphical Programming Environment). It stores the full hierarchical computational data sets in an efficient data structure tailored to the requirements of post processing, the visualization and data analysis methods access the data through GRAPE's procedural interface. This allows the memory- and time-efficient handling of large – stationary and time-dependent – data sets and capitalizing on the hierarchical structure for adaptive post processing methods. The interfaces have been tailored to the specific needs of the E-Dur project. In the following an overview of the new functionality will be given.

10.1 Fractures

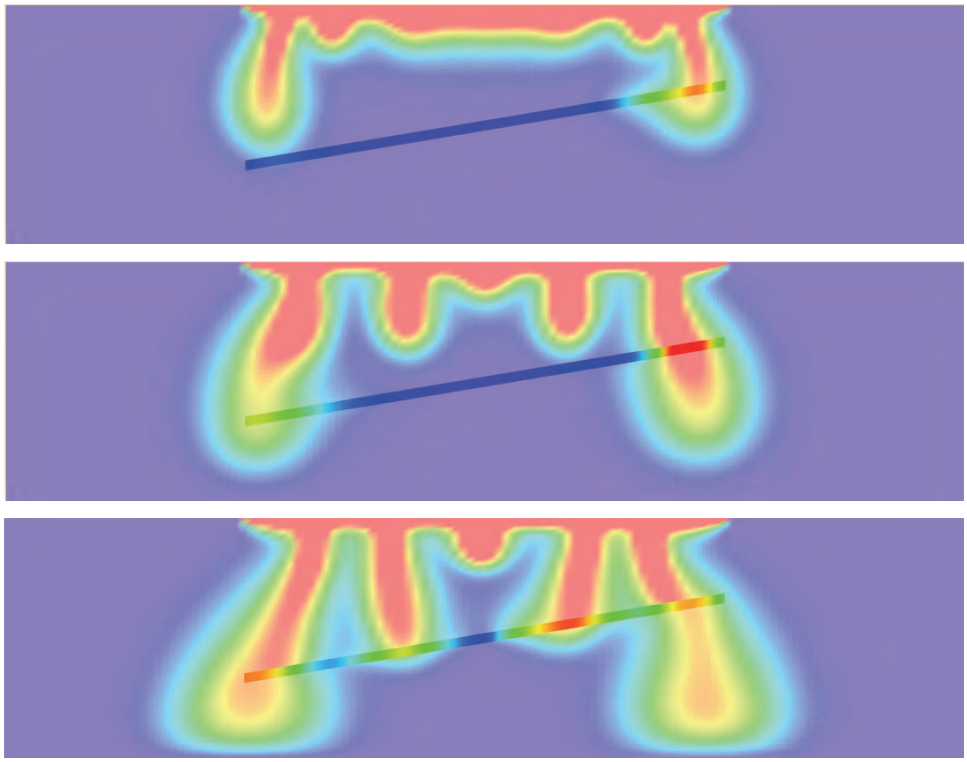


Fig. 10.1 Salt concentration at three time steps of the Elder problem with a diagonal fracture
Concentration is depicted in the fracture (dark colours) and in the bulk (light colours).

The display methods for fractures are realized by a substantial extension of the concept of sub domains, taking into account that fractures are objects which are essentially of one dimension lower than the computational domain. Fig. 10.1 illustrates the visualization of a scalar function that has values in the fracture and outside. Visualization of the function in the bulk and in the fracture is performed by separate display methods to increase the post processing flexibility. Here, the same colour bar is used by both methods, but the saturation of colours is decreased in the bulk to facilitate the distinction from the values in the fracture. On the other hand, the one-dimensional fracture has been inflated in normal direction to allow the visualization of a scalar quantity in the fracture by colour shading.

In the computational mesh fractures are represented by so called degenerated elements (see section 5.5.1). These are elements for which the vertices of two opposite sides may coincide. From a geometrical point of view these elements are lower dimen-

sional objects, as they possess no thickness, but they are still full dimensional elements of the mesh in the topological sense. Indeed, for example in two space dimensions this means that a quadrilateral belonging to a fracture has no area but it gives access to four adjacent elements. Each fracture consists of two layers of degenerate elements, each associated with one of the two sides of the fracture. In the following figures this situation is sketched topologically. Thereby red vertices indicate the “inner” nodes associated with the fracture and the blue vertices are shared by the bulk elements adjacent to the fracture.

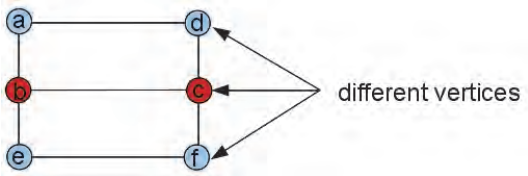


Fig. 10.2 Topological structure of two elements in a fracture

Fig. 10.3 Geometrical representation of the two elements from Fig. 10.2

As the data format for exchanging data between the simulator and the post processor does not differentiate between non degenerate and degenerate elements, the location of the fracture has to be reconstructed using the topological and geometrical information stored together with the mesh.

This algorithm consists on the following steps:

- Find all degenerated elements on the macro level.
- Determine for all degenerated elements which faces have neighbours belonging to the bulk media. Vertices belonging to these faces are called “outer vertices”. Calculate the normal vectors on these faces.
- If for a vertex there are no faces with a bulk neighbour, this vertex is called “inner vertex”.

This distinction is necessary because numerical data representing vector- and scalar-valued function is attached to the vertices of the mesh, i. e. that the degrees of freedoms representing quantities within a fracture are associated with “inner vertices”,

whereas the quantities at each side of the fracture are given by the degrees of freedom at the “outer vertices” of a given fracture.

The reconstruction of the fracture has to be done only on the coarsest level of the mesh, where all the topological information is available. For elements belonging to more refined levels of the mesh, the information whether an element belongs to a fracture or not and whether a vertex is an inner or an outer vertex is inherited using the hierarchical architecture of the mesh.

To visualize the degenerated elements, they have to be artificially enlarged in the direction given by the normal.

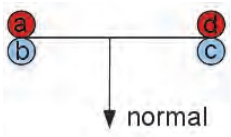


Fig. 10.4 Fracture element with normal pointing outwards

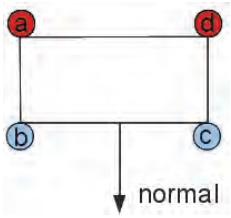


Fig. 10.5 The outer vertices are displaced in normal direction

The new display methods are capable of visualizing scalars as well as vector valued functions within the fracture. This functionality is implemented in the following methods:

fracture: colour shading of a scalar function on a 1d fracture in a 2d domain. The fracture is colour shaded based on the values of the scalar-valued function provided in the slot **scalar** of the function selector. The user can specify the colour bar and amount by which the fracture is inflated in normal direction. Cf. Fig. 10.1, where it is employed in conjunction with the method **isoline** to show the same function in the bulk.

fracture3d: colour shading of a scalar function on a 2d fracture in 3d. As above, the method operates on one scalar function. Colour bar and (artificial) thickness of the fracture can be selected. Cf. Fig. 10.7.

clip-fracture3d: colour shading of a scalar function on a cross section through a 2d fracture in 3d. This method has the same set of parameters as **fracture3d** above. In addition, the clipping plane can be selected. Cf. Fig. 10.8.

vec_fracture: plotting a vector field in a 1d fracture in 2d with arrows. A vector valued function – from the slot **vector** of the function selector – is visualized using arrows. The user can control the width of the inflated fracture and the scaling of the arrows. Additionally, the arrows can be coloured according to the absolute value of the function, cf. Fig. 10.6.

vec_fracture3d: plotting a vector field in a 2d fracture in 3d with arrows. This is the 3d equivalent to **vec_fracture**, the parameters are as above.

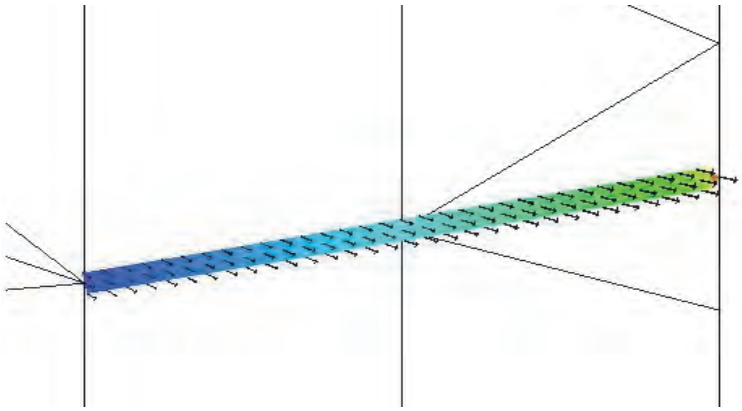


Fig. 10.6 A vector field in a fracture

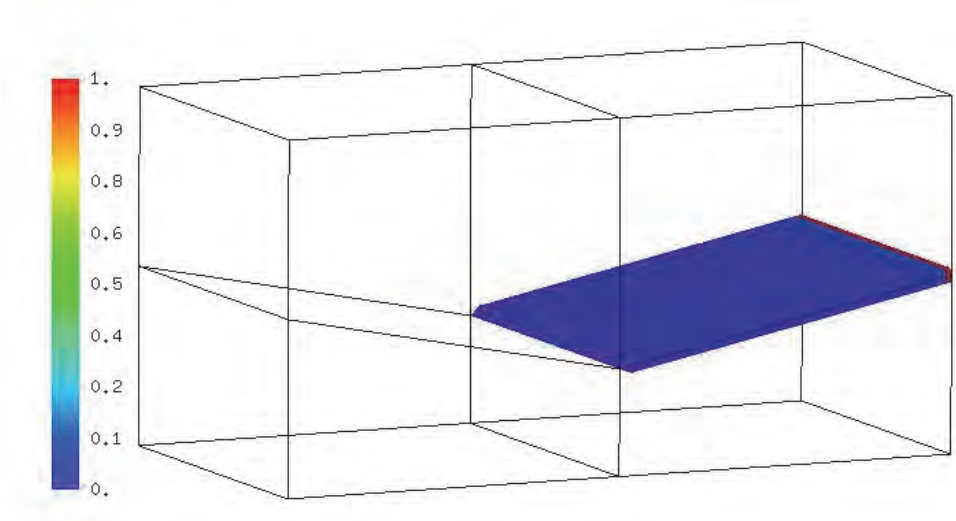


Fig. 10.7 A scalar field in a 2D fracture in three dimensions

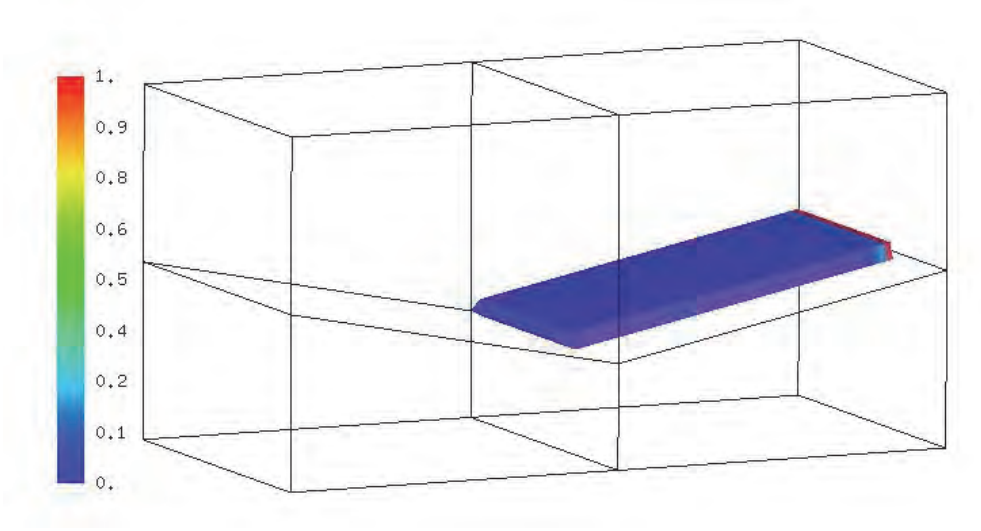


Fig. 10.8 Cross section through a two dimensional fracture in 3D

10.2 Visualization on free and moving boundaries

Another new aspect in this project was the introduction of free boundaries. These a priori unknown boundaries are represented as the zero level set of a utility function, the "level set function" (see section 6).

From the perspective of post processing, there are two important aspects of free boundaries. On the one hand, one wants to visualize the geometry of the free boundary. This can be done by the display methods **isoline** (in two dimensions) and **level** (in the three-dimensional case). It is now also possible to colour surfaces given as level sets of one specific function depending on the value of another scalar function, as is shown in Fig. 10.9 for an artificial, spherical level set function. This allows in an effective way to colour code data such as a concentration of the boundary implicitly described as a level set.

On the other hand, especially since on the outside of the free boundary (where there is no water) functions (such as concentration of salt or radionuclides) usually do not have meaningful values, one may want to restrict the visualization of functions to the subset of the level set actually representing the inside of the domain. This can in general be achieved using the Data Analysis Tool (DAT), which allows to restrict the visualization to a part of the mesh in a very general and flexible way. This approach works together with nearly all display methods (with the exception of those where the restriction to a part of the domain does not make sense). Selected display methods (**clip-isoline-**

multi and **spades**) have been remodelled to increase the performance and visual quality when used in conjunction with the DAT, an example – again using an artificial, spherical level set function – is shown in Fig. 10.10.

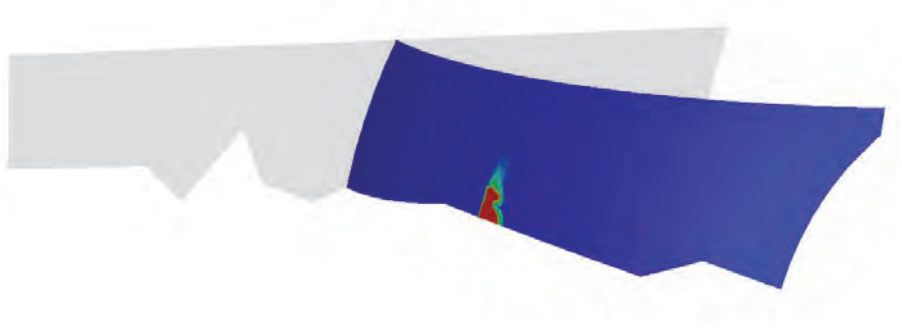


Fig. 10.9 Salt concentration on the level set of an artificial level set function; display method **level**

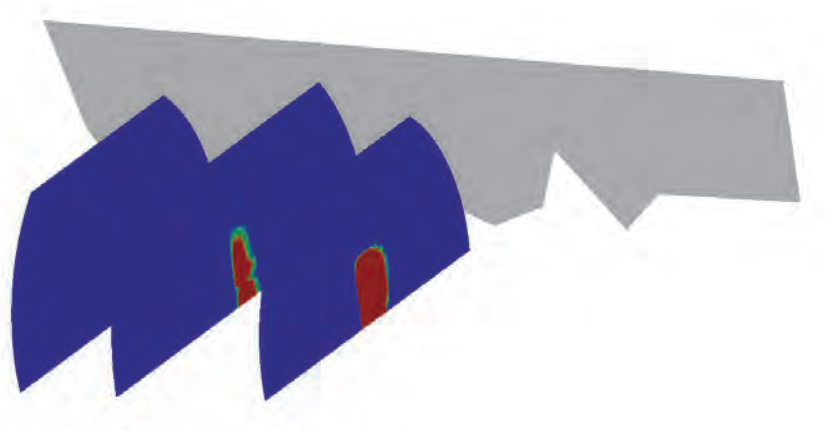


Fig. 10.10 Salt concentration on parallel clipping planes; the drawing region is determined as the sub level set of an (artificial) level set function; display method is **clip-isoline-multi**

For the exploration of free boundaries, especially in conjunction with additional data, the following methods have been implemented or adapted:

expand-to-dat-send: call on a **Scene** or **TimeScene** to activate the Data Analysis Tool (DAT). The DAT allows the user to select and modify the function to be displayed and the part of the domain to be displayed. If one enters **LEQ(phi,0)** – assuming **phi** is the level set function – as the *set* parameter to be parsed and interpreted by the DAT, visualization will be restricted to the sub-levelset of **phi** in all compatible display methods.

clip-isoline-multi and **spade:** colour shading on several clipping planes. These two display methods have been modified for increased performance and optimized graphical quality when working with domains with free and moving boundaries.

level: draw a selected level surface of a scalar function in 3d. The user can specify the level set to be drawn. In the context of free boundaries, the parameter **level** has to be set to zero. In addition, the method **level** has been modified to allow the colour shading of the free boundary according to another scalar function, e. g. a concentration. The function used for colour shading is taken from the slot **texture** of the function selector. A button **texture** allows to activate or to deactivate this feature.

10.3 Visualization of sources and sinks

The extension of d^3f and r^3t features the treatment of source respectively sink terms. Sources and sinks are represented by lines or points and they are accessed via an additional data file, where position, length, and capacity of the sources and sinks are stored. Sources or sinks are displayed by red or blue tubes, respectively, showing the position of the sources or sinks. Additionally arrows are displayed above the domain, pointing downwards for sinks and upwards for sources. The length of these arrows is scaled with respect to the amount of discharge and can be animated to reflect time dependent discharge. The functionality allowing to display sources and sinks is encapsulated in the project "**source**". This project manages the reading of the input file, the calls for the actual display-methods and options for the visualization:

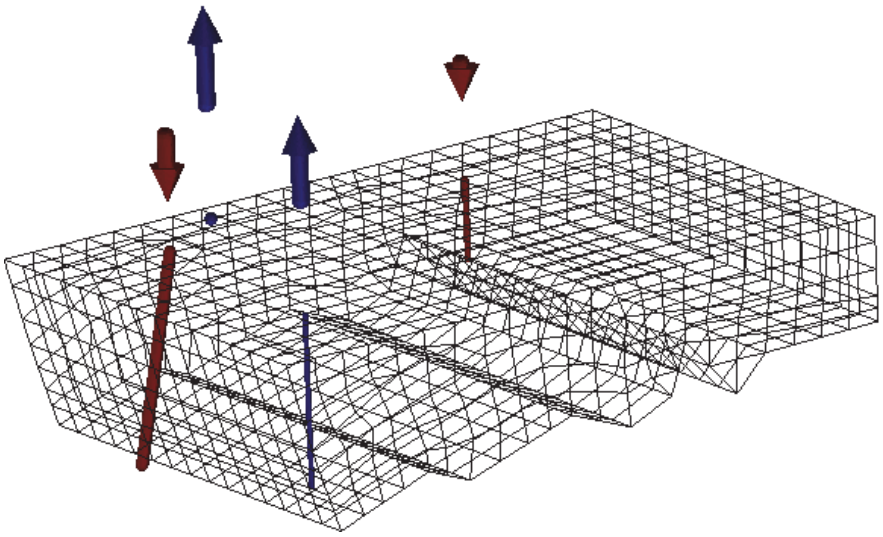


Fig. 10.11 Sources and sinks in a model geometry

source: project for the visualization of sources and sinks. With the button **Load source**, the user can specify the **source** file with the description of sources and sinks. The radius of the tubes used in the visualization can be adjusted via **pipe radius**.

10.4 Function evaluation probing and integration

Visualization methods on the whole computational domain (such as colour shading of scalar functions) give a good qualitative measure of data functions. If one needs more precise quantitative results it is useful to determine the exact value at distinct points or more general evaluated along curves or planes.

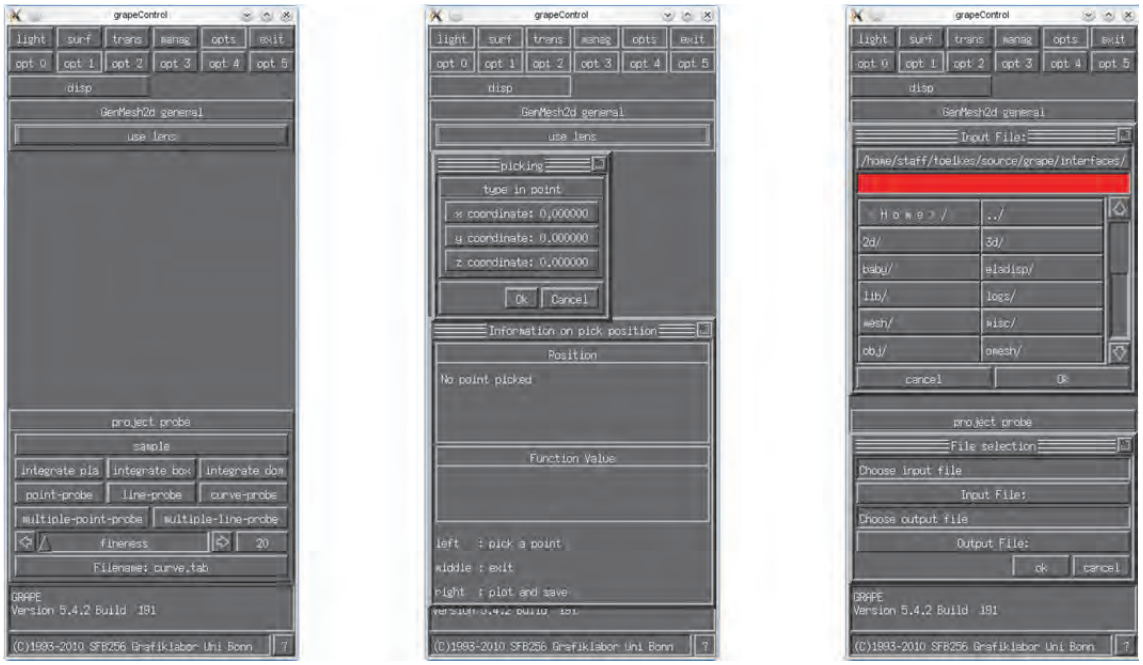


Fig. 10.12 Several probe input methods

The "**probe**" module provides this functionality. The values can be displayed directly or saved in plain ASCII files for easy storage, reference and further processing. For probing multiple points or lines it is possible to read coordinates from files. Algorithmically this is based on a hierarchical search in the element tree, exploiting data locality when evaluating on nearby points.

Points can be probed by either clicking into the visualization or by entering coordinates, either by hand or by parsing a coordinate file, which provides the coordinate of the points to probe. Similarly it is possible to probe over straight lines.

Furthermore it is possible to probe arbitrary curves by providing the necessary data via an input file. Again the result can be displayed and saved to a data file for further processing.

The possibility to sample functions on an equidistant Cartesian grid and write the results to a plain text file can be used to import data into other programs for comparison, further processing etc.

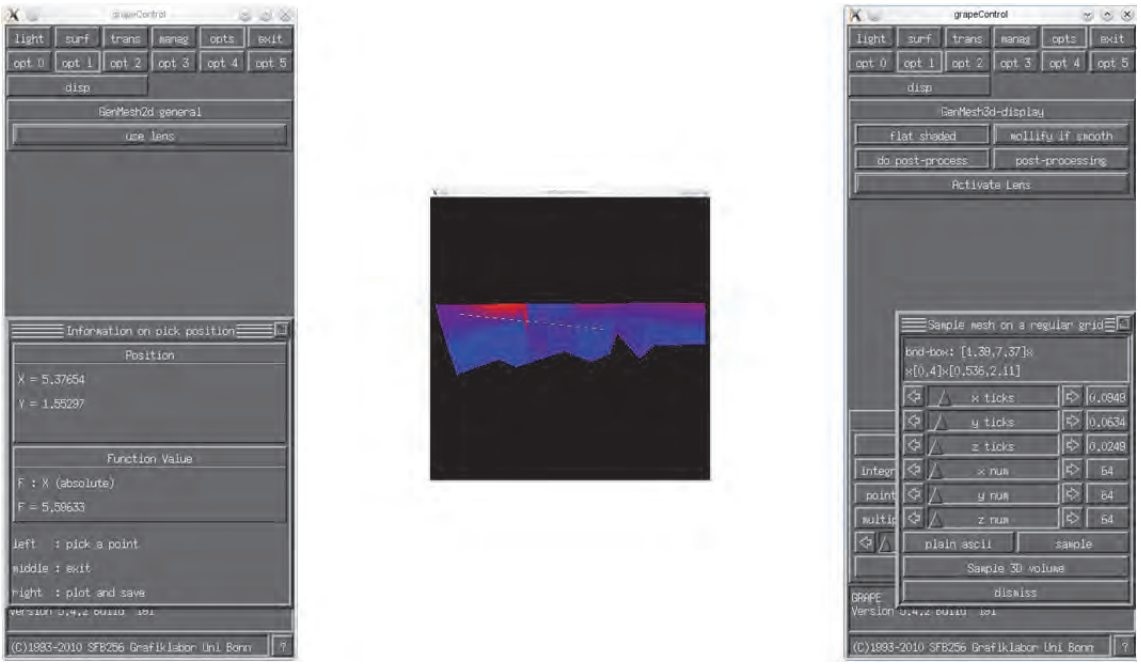


Fig. 10.13 Probe output (left) and grid sampling (right)

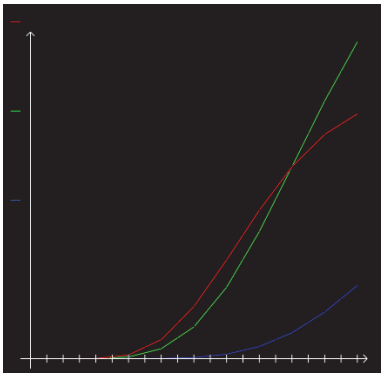


Fig. 10.14 Plot from probing at three different points

A second class of functionality provided by the **probe** module are the integration methods: It is possible to integrate over two dimensional planes or three dimensional boxes or even over the whole domain. This is needed e. g. when determining flows over

boundaries or planes within the domain. Integrating over full dimensional boxes is a useful addition to this functionality. The integration methods where substantially extended in their functionally with the E-Dur project. Several options were added in 2D and/or 3D, for example there now is a method to integrate the flow over an arbitrary user defined line in 2D. Furthermore the user interface was unified and modified to improve the work flow when using this methods. The probing interface is now a powerful ingredient of the post processing.

All this functionality is provided for 2D and 3D domains as well as for scalar- and vector valued function data enabling to extract in a flexible interactive way quantitative information and generate corresponding lower dimensional plots of the post processed data. Additionally this includes the possibility to evaluate (and plot) the function over the time or at a specific time, and allows to create multiple outputs when necessary (e. g. vector valued data). The structures and methods involved are:

probe: project for probing and integration. This project collects all methods for evaluating functions at specified sets of points and for integration. The user can select the **Filename** of the output file and control the **fineness** of the discretization. All methods can be used on stationary and time-dependent data, for scalar and vector valued data (with some obvious exceptions) and (if appropriate) in two and three space dimensions.

point-probe: evaluating at single points. A scalar or vector valued function can be evaluated at points selected by the user. Coordinates can be specified by clicking in the visualization window or entered manually.

line-probe: evaluating along a line. A function is evaluated at a number of points along a line. The end points of the line are entered in the same way as in **point-probe**, the discretization is given by the ruler **fineness**.

curve-probe: evaluating along an arbitrary curve. This method evaluates the function along a curve given as a polygonal object. The function is evaluated at all points of the polygon. The polygon has to be provided as a **Triang1d** object.

sample: evaluate on a Cartesian grid. The selected function is sampled on a Cartesian grid. The discretization of this grid can be controlled by specifying the number of points or the point distance in each direction. Output can be ASCII or binary.

integrate plane integrate over a plane. This method integrates the flow over a plane (i. e. it integrates $c\ v \cdot n$ where c is a concentration, v a velocity and n the normal to the plane) in a three dimensional domain. The functions c and v are specified in the slots **scalar** and **vector** of the function selector, the plane is selected by the plane editor. The analogue is possible for lines in 2d.

integrate box: integrate a function over a box. This method integrates a function over box. The box can be specified by giving two corners (assuming the edges to be aligned to the coordinate axes) and / or interactively shifted and rotated. Integration is done by an adaptive method based on point evaluations, the number of **initial levels** and the **epsilon** error threshold can be selected.

integrate domain: integrate over the whole domain. Integrates the selected function over the full computational domain. If some sub domains have been deactivated, these are also excluded from the integration.

integrate-bnd-send: integrate over the boundary. Integrate the flow over the boundary of the computational domain analogously to **integrate plane**. As in **integrate domain**, only active sub domains are considered. The functions are given in the same way as for **integrate plane**.

10.5 Visualization of stochastic data

A particular computational challenge in E-Dur are domain descriptions which incorporate stochastic data, such as statistical values for the permeability of different materials. Such data usually does not fit the underlying computational mesh. Typically this type of data has to be displayed on a certain cross section of the bulk domain. If the user has specified such a cross section the stochastic data is mapped to that cross section via graphic hardware acceleration using the 3D texture functionality provided by OpenGL. Thereby, a three dimensional texture is a set of colour values in a three dimensional array. The texture mapping maps points to be rendered to an index in the three dimensional array and therefore to the colour value for that point. The input files have to provide the dimensions of the 3D array and list of period statistical data values which are then converted by the Grape routines to a 3D texture that is passed to the OpenGL routines. This functionality is provided by the method

3dtexture: 3d texture mapping for stochastic permeabilities. The display method **3dtexture** maps scalar data given in 3d on a Cartesian grid to a cross section of the computational domain. The user can load the permeability file with the button **Load File**. The colour mapping can be specified by selecting and / or modifying a colour bar.

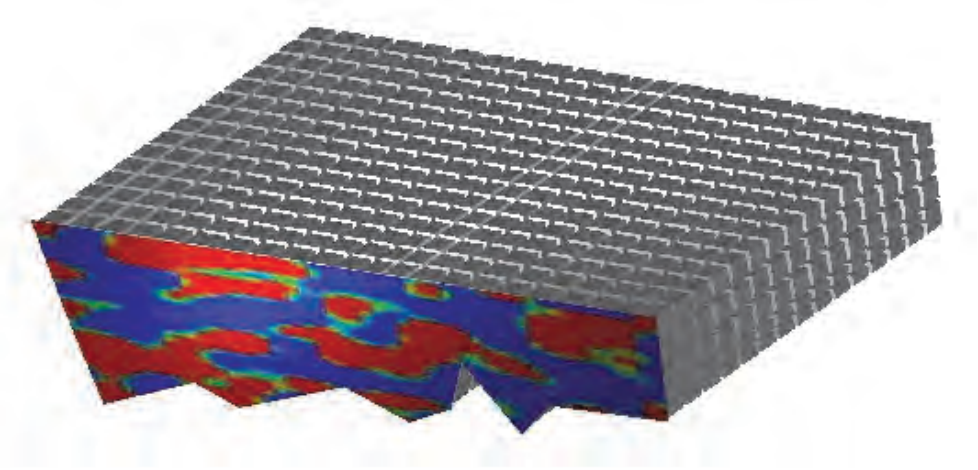


Fig. 10.15 Stochastic data mapped to a cross section of the model geometry

10.6 Visualization of sub domains

The visualization methods for sub domains have been extended, improved, and adapted in several aspects. E. g. it is now much easier to specify a “realistic” colouring for the sub domains corresponding to the different media:

subdomain display sub domains. This display method draws all or selected sub domains. With the button **subdomains**, you can open a window that allows sub domains to be be blanked out. Additionally, each sub domain can be assigned a RGB colour value at this point. The buttons **Save / Load File** make it possible to save a manually specified colouring of the set of sub domains and to reload it later.

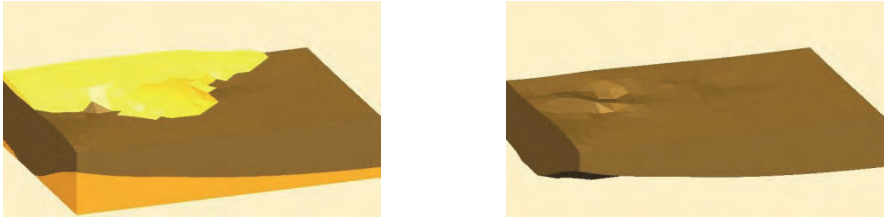


Fig. 10.16 Visualization of sub domains with manually selected colouring

11 Code verification and applications

To verify the code advances of d³f and r³t various test cases were performed. These were test cases with existing analytical solution as well as such that might be compared with results got by other software codes.

11.1 Test cases with thermohaline-driven flow

11.1.1 Test case „1d-heat transport“

11.1.1.1 Balance equation

The time-dependent amount of heat *Q(t)* in a moving domain *B(t)* can be written as

$$Q(t) = \int_{B(t)} w(\mathbf{x}, t)\, dV \tag{11.1}$$

Q(t)- heat in *B(t)* [J]

B(t)- time-dependent 3D-domain [m³]

w(t)- heat density [J m^{-3}]

Allowing for a heat flux *J* across the moving surface of *B* and including a heat source *r* within *B,* Reynolds' transport theorem yields for a fixed domain *G* (e. g. /GAR 87/):

$$\int_G \left[\frac{\partial w}{\partial t} + \nabla \cdot \left(\mathbf{v} w + \mathbf{J} \right) - r \right] dV = 0 \tag{11.2}$$

G - fixed domain [m³]

v - pore velocity [m s^{-1}]

J - non-advective heat flux across the surface of *G* [J m^{-2} s^{-1}]

r - sink/source of *Q* in *G* [J m^{-3} s^{-1}]

The velocity $\mathbf{v}$ represents here the pore velocity which is related to the Darcy flux $\mathbf{q}$ by the porosity as

$$\mathbf{v} = \frac{\mathbf{q}}{\phi} \tag{11.3}$$

$\mathbf{q}$ - Darcy flux [m s^{-1}]

ϕ - porosity [-]

However, the source term *r* is not of interest here and will be dropped in the following. The process of heat transport covers the fluid-filled pore space as well as the solid matrix in *G* which has to be reflected in the balance equation:

$$\int_G \left[\frac{\partial w_\alpha}{\partial t} + \nabla \cdot \left(\mathbf{v}_\alpha w_\alpha + \mathbf{J}_\alpha \right) \right] dV = 0 \tag{11.4}$$

α - Index; $\alpha \triangleq f$ for fluid, $\alpha \triangleq s$ for solid

Introducing porosity as the ratio of pore volume to total volume

$$\phi = \frac{V_f}{V_f + V_s} \tag{11.5}$$

V_f - volume of the pore space [m³]

V_s - volume of the solid part (matrix) [m³]

allows to quantify the complementary fractions of domain *G*

$$G = G_f + G_s = \phi G + (1 - \phi) G \tag{11.6}$$

G_f - Volume of the fluid [m³]

G_s - Volume of the solids [m³]

Assuming a rigid matrix the two balance equations (11.4) can be added as

$$\phi \left[\frac{\partial w_f}{\partial t} + \nabla \cdot \left(\mathbf{v_f} w_f + \mathbf{J_f} \right) \right] + (1 - \phi) \left[\frac{\partial w_s}{\partial t} + \nabla \cdot \left(\mathbf{v_s} w_s + \mathbf{J_s} \right) \right] = 0 \tag{11.7}$$

For heat conduction in solid matter and immobile fluids Fourier's law applies which is formally identical with Fick's law of diffusion:

$$\mathbf{J}_{\mathbf{T}\alpha} = -\boldsymbol{\lambda}_\alpha \cdot \nabla T_\alpha \tag{11.8}$$

$\mathbf{J}_\mathbf{T}$ - heat flux due to thermal conductivity [J m^{-2} s^{-1}]

$\boldsymbol{\lambda}$ - thermal conductivity [J m^{-1} K^{-1} s^{-1}]

Spreading of heat according to the same principles as mechanical dispersion has to be considered also:

$$\mathbf{J}_{\mathbf{D}\alpha} = -\mathbf{D}_{\mathbf{m}\alpha} \cdot \nabla w_\alpha \tag{11.9}$$

$\mathbf{J}_{\mathbf{D}}$ - heat flux due to mechanical Dispersion [J m² s^{-1})]

$\mathbf{D}_{\mathbf{m}}$ - mechanical dispersion [m² s^{-1}]

Heat flux due to mechanical dispersion adds to the conductive heat flux:

$$\mathbf{J}_{\alpha} = \mathbf{J}_{\mathbf{T}\alpha} + \mathbf{J}_{\mathbf{D}\alpha} \tag{11.10}$$

Next, the flow processes will be looked upon closely. Contrary to advection of the fluid phase no movement of the solid phase is considered here

$$\mathbf{v}_{\mathbf{s}} = 0 \tag{11.11}$$

from which directly follows

$$\mathbf{J}_{\mathbf{D}\,\mathbf{s}} = 0 \tag{11.12}$$

Equations (11.8) to (11.12) transform balance equation (11.7) into

$$\begin{aligned} &\phi\left[\frac{\partial w_f}{\partial t} + \nabla \cdot \left(\mathbf{v}_{\mathbf{f}} w_f - \boldsymbol{\lambda}_{\mathbf{f}} \cdot \nabla T_f - \mathbf{D}_{\mathbf{m}\,\mathbf{f}} \cdot \nabla w_f\right)\right] \\ &+ \left(1-\phi\right)\left[\frac{\partial w_s}{\partial t} - \nabla \cdot \left(\boldsymbol{\lambda}_{\mathbf{s}} \cdot \nabla T_s\right)\right] = 0 \end{aligned} \tag{11.13}$$

In the next step the heat density can be eliminated from balance equation (11.13). The equation contains heat density only as a subject to derivatives and differences of the heat density can be related to differences of temperature (e. g. /FLU 61/) by:

$$\partial w_\alpha = C_\alpha \partial T_\alpha \tag{11.14}$$

C - heat capacity [J m^{-3} K^{-1}]

T - temperature [K]

Heat capacity in turn can be expressed by the specific heat capacity and the mass density:

$$C_\alpha = c_\alpha \rho_\alpha \tag{11.15}$$

c_α - specific heat capacity [J kg^{-1} K^{-1}]

ρ - mass density [kg m^{-3}]

Inserting (11.14) and (11.15) in balance equation (11.13) yields

$$\begin{aligned}&\phi\left[\frac{\partial(c_f \rho_f T_f)}{\partial t} + \nabla\cdot\left(\mathbf{v_f} c_f \rho_f T_f - \boldsymbol{\lambda}_\mathbf{f}\cdot\nabla T_f - \mathbf{D_{mf}}\cdot\nabla\left(c_f\rho_f T_f\right)\right)\right]\\&+\left(1-\phi\right)\left[\frac{\partial(c_s\rho_s T_s)}{\partial t} - \nabla\cdot\left(\boldsymbol{\lambda}_\mathbf{s}\cdot\nabla T_s\right)\right] = 0\end{aligned} \tag{11.16}$$

In the literature local thermal equilibrium between fluid and solids is usually assumed (e. g. /HYA 83/). Therefore only one temperature has to be considered. Note that it thus does not make sense to differentiate between a thermal source in the fluid and a source in the solid:

$$\begin{aligned}&\phi\left[\frac{\partial(c_f \rho_f T)}{\partial t} + \nabla\cdot\left(\mathbf{v_f} c_f \rho_f T - \boldsymbol{\lambda}_\mathbf{f}\cdot\nabla T - \mathbf{D_{mf}}\cdot\nabla\left(c_f\rho_f T\right)\right)\right]\\&+\left(1-\phi\right)\left[\frac{\partial(c_s\rho_s T)}{\partial t} - \nabla\cdot\left(\boldsymbol{\lambda}_\mathbf{s}\cdot\nabla T\right)\right] = 0\end{aligned} \tag{11.17}$$

Transforming the first two terms in (11.16) by means of the product rule leads to

$$\begin{aligned}&c_f T\left(\phi\frac{\partial\rho_f}{\partial t} + \phi\nabla\cdot\left(\mathbf{v_f}\rho_f\right)\right)\\&+\phi\left[\rho_f\frac{\partial\left(c_f T\right)}{\partial t} + \rho_f\mathbf{v_f}\cdot\nabla\left(c_f T\right) - \nabla\cdot\left(\boldsymbol{\lambda}_\mathbf{f}\cdot\nabla T + \mathbf{D_{mf}}\cdot\nabla c_f\rho_f T\right)\right]\\&+\left(1-\phi\right)\left[\frac{\partial(c_s\rho_s T)}{\partial t} - \nabla\cdot\left(\boldsymbol{\lambda}_\mathbf{s}\cdot\nabla T\right)\right] = 0\end{aligned} \tag{11.18}$$

where the expression in round brackets represents the left hand side of the continuity equation for fluids in a rigid porous medium (e. g. /KRO 91/):

$$\phi \frac{\partial \rho_f}{\partial t} + \phi \nabla \cdot \left(\mathbf{v_f} \rho_f \right) = \rho \dot{V} \tag{11.19}$$

$\dot{V}$ - volumetric fluid sink/source [m³ m^{-3} s^{-1}]

ρ - density of the fluid crossing the domain boundary [kg m^{-3}];
for a sink: $\rho = \rho_f$, for a source ρ can assume any value

Since no sinks or sources are considered here the right hand side of (11.19) equals zero. Balance equation (11.18) can thus be simplified to

$$\begin{aligned} &\phi \left[\rho_f \frac{\partial \left(c_f T \right)}{\partial t} + \rho_f \mathbf{v_f} \cdot \nabla \left(c_f T \right) - \nabla \cdot \left(\boldsymbol{\lambda}_\mathbf{f} \cdot \nabla T + \mathbf{D_{mf}} \cdot \nabla c_f \rho_f T \right) \right] \\ &+ \left(1 - \phi \right) \left[\frac{\partial (c_s \rho_s T)}{\partial t} - \nabla \cdot \left(\boldsymbol{\lambda}_\mathbf{s} \cdot \nabla T \right) \right] = 0 \end{aligned} \tag{11.20}$$

Now the derivatives are transformed once more:

$$\begin{aligned} &\phi \left[\begin{array}{l} \rho_f \left\{ T \frac{\partial c_f}{\partial t} + c_f \frac{\partial T}{\partial t} \right\} + \rho_f \mathbf{v_f} \cdot \left\{ T \nabla c_f + c_f \nabla T \right\} \\ - \nabla \cdot \left(\boldsymbol{\lambda}_\mathbf{f} \cdot \nabla T + \mathbf{D_{mf}} \cdot \left\{ T \nabla c_f \rho_f + c_f \rho_f \nabla T \right\} \right) \end{array} \right] \\ &+ \left(1 - \phi \right) \left[\left\{ T \frac{\partial \left(c_s \rho_s \right)}{\partial t} + c_s \rho_s \frac{\partial T}{\partial t} \right\} - \nabla \cdot \left(\boldsymbol{\lambda}_\mathbf{s} \cdot \nabla T \right) \right] = 0 \end{aligned} \tag{11.21}$$

Assuming that

- changes in rock density,
- changes in the specific heat capacity of water, and
- changes in the specific heat capacity of rocks

can be neglected in comparison to the changes in temperature (for the referring data see /KRO 10/) equation (11.21) simplifies to

$$\begin{aligned} &\phi \left[\rho_f c_f \frac{\partial T}{\partial t} + \rho_f c_f \mathbf{v_f} \cdot \nabla T - \nabla \cdot \left(\boldsymbol{\lambda}_\mathbf{f} \cdot \nabla T + c_f \rho_f \mathbf{D_{mf}} \cdot \nabla T \right) \right] \\ &+ \left(1 - \phi \right) \left[c_s \rho_s \frac{\partial T}{\partial t} - \nabla \cdot \left(\boldsymbol{\lambda}_\mathbf{s} \cdot \nabla T \right) \right] = 0 \end{aligned} \tag{11.22}$$

or rearranged to

$$\begin{aligned}&\left\{\phi\rho_f c_f + (1-\phi)c_s\rho_s\right\}\frac{\partial T}{\partial t} + \phi c_f \rho_f \mathbf{v_f}\cdot\nabla T \\ &-\nabla\cdot\left(\left\{\phi\boldsymbol{\lambda}_\mathbf{f} + (1-\phi)\boldsymbol{\lambda}_\mathbf{s}\right\} + \phi c_f\rho_f \mathbf{D_{mf}}\right)\cdot\nabla T = 0\end{aligned} \tag{11.23}$$

The terms in curly brackets represent weighted means of the heat capacity and of the thermal conductivity or their bulk value, respectively:

$$\begin{aligned}\overline{C} &= \phi\rho_f c_f + (1-\phi)c_s\rho_s \\ \overline{\boldsymbol{\lambda}} &= \phi\boldsymbol{\lambda}_\mathbf{f} + (1-\phi)\boldsymbol{\lambda}_\mathbf{s}\end{aligned} \tag{11.24}$$

$\overline{C}$ - bulk heat capacity [J m^{-3} K^{-1}]

$\overline{\boldsymbol{\lambda}}$ - bulk thermal conductivity [J m^{-1} K^{-1} s^{-1}]

With these abbreviations (11.23) is transformed into

$$\frac{\partial T}{\partial t} + \phi\frac{c_f\rho_f}{\overline{C}}\mathbf{v_f}\cdot\nabla T - \nabla\cdot\left(\frac{\overline{\boldsymbol{\lambda}}}{\overline{C}} + \phi\frac{c_f\rho_f}{\overline{C}}\mathbf{D_{mf}}\right)\cdot\nabla T = 0 \tag{11.25}$$

leading to a form that compares nicely to the tracer transport equation

$$\frac{\partial c}{\partial t} + \mathbf{v_c}\cdot\nabla c - \nabla\cdot\left(\mathbf{D_c}\cdot\nabla c\right) = 0 \tag{11.26}$$

c - concentration [kg$_{tracer}$ kg$_{fluid}^{-1}$]

$\mathbf{v_c}$ - pore velocity (solute transport) [m² s^{-1}]

$\mathbf{D_c}$ - hydrodynamic dispersion [m² s^{-1}]

where terms are related as follows

$$\begin{aligned}&\text{a)}\quad \mathbf{v_c} = \frac{1}{R}\mathbf{v_f} \\ &\text{b)}\quad \frac{1}{R} = \phi\frac{c_f\rho_f}{\overline{C}} \\ &\text{c)}\quad \mathbf{D_c} = \frac{1}{\overline{C}}\left(\overline{\boldsymbol{\lambda}} + \phi c_f\rho_f\mathbf{D_{mf}}\right)\end{aligned} \tag{11.27}$$

R - retardation coefficient [-]

If (11.26) is interpreted as a heat transport equation $\mathbf{D_c}$ has the meaning of a thermal dispersion and $\mathbf{v_c}$ is a retarded velocity.

11.1.1.2 Analytical solution for advective-diffusive transport

For an instantaneous increase of concentration at the boundary of a one-dimensional semi-infinite domain an analytical solution of the advection-diffusion equation (11.26) was derived by Ogata and Banks /OGA 61/:

$$\frac{c-c_0}{c_1-c_0}=\frac{1}{2}\left[erfc\left(\frac{x-v_c t}{2\sqrt{D_c t}}\right)+e^{\left(\frac{v_c x}{D_c}\right)}erfc\left(\frac{x+v_c t}{2\sqrt{D_c t}}\right)\right] \qquad (11.28)$$

c_0 - initial concentration [kg_{tracer} kg_{fluid}^{-1}]

c_1 - boundary concentration [kg_{tracer} kg_{fluid}^{-1}]

x - distance from the boundary [m]

This solution has already been applied to verify other numerical heat transport models (SWIFT: /WAR 84/, HydroGeoSphere: /THE 10/) using the parameters listed in Tab. 11.1. Note that in these cases the pore velocity was used for the Ogata-Banks solution (OBs) instead of the Darcy velocity. Flow parameters and boundary conditions were chosen accordingly for the d^3f-model.

Tab. 11.1 Parameter values for the Ogata-Banks model after /THE 10/

Parameter	Symbol	Value	Unit
bulk thermal conductivity	$\overline{\lambda}$	2.16	J m^{-1} K^{-1} s^{-1}
specific heat capacity of solid	c_s	1254.682	J kg^{-1} K^{-1}
specific heat capacity of water	c_f	4185	J kg^{-1} K^{-1}
solid density	ρ_s	1602	kg m^{-3}
fluid density	ρ_f	1000	kg m^{-3}
matrix porosity	ϕ	0.1	(-)
longitudinal dispersivity	α_L	14.4	m
thermal dispersion coefficient	D_c	1.15 10^{-5}	m^2 s^{-1}
retardation coefficient	R	5.323	(-)
Darcy flux	q_f	3.53 10^{-7}	m s^{-1}
retarded pore velocity	v_c	6.63 10^{-7}	m s^{-1}
initial temperature	T_0	37.78	°C
boundary temperature	T_1	93.33	°C
domain size	L	600	m
output times	t1, t2	2148, 4262	d

The data given in Tab. 11.1 is partially redundant and thus can be used to check their consistency as well as the understanding of the parameters and of the derived equations. Checks are possible calculating

- the retardation coefficient using (11.27) b),
- the retarded pore velocity using (11.3) and (11.27) a), and
- the thermal diffusion coefficient (11.27) c).

While retardation coefficient and retarded pore velocity can be reproduced the calculated thermal diffusion coefficient of 1.05 10^{-5} m s^{-1} differs slightly for unknown reasons from the tabulated data.

11.1.1.3 Analytical solution for heat conduction

Close inspection of equation (11.28) reveals that the OBS is equivalent with the solution of /CAR 59/ for pure heat conduction if the flow velocity is set to 0. In an even shorter form using dimensionless variables it is presented for solute transport in /TUR 82/ as

$$\hat{c} = erfc(\xi) \tag{11.29}$$

$\hat{c}$ - dimensionless concentration [-]

ξ - dimensionless spatial coordinate [-]

$$\hat{c} = \frac{c - c_0}{c_1 - c_0} \tag{11.30}$$

c - concentration [kg_{tracer}/kg_{fluid}]

$$\xi = \frac{x}{2\sqrt{D_c t}} \tag{11.31}$$

ξ - dimensionless spatial coordinate [-]

The OBS can thus be safely used for checking numerical solutions for pure heat conduction.

11.1.1.4 Test of heat conduction in d³f

In a preliminary test, the OBS was used to test the implementation of heat conduction in d³f. To ensure the equivalence of the heat conduction problem to the problem described by /OGA 61/ a rigid matrix with constant density, viscosity, and porosity was assumed. The data listed in Tab. 11.2 was used as input for the d³f-simulation. The thermal dispersion coefficient was not assigned explicitly but resulted from the values set for the heat capacities, densities and the bulk thermal conductivity.

The one-dimensional problem formulated by Ogata and Banks was represented by a two-dimensional vertical model with stagnant pore water. Length and height amounted to 100 m and 10 m, respectively. Boundary conditions for temperature, pressure, concentration and flow velocity were assigned according to Fig. 11.1.

The thermal dispersion coefficient D_c used for the calculation of the analytical solution had to be adapted to the actual problem. Since mechanical dispersion is omitted without convection, only the bulk thermal conductivity and the bulk heat capacity contribute to the calculation of the thermal dispersion coefficient (cp. (11.27) c)). All parameters used for the analytical solution are listed in Tab. 11.2.

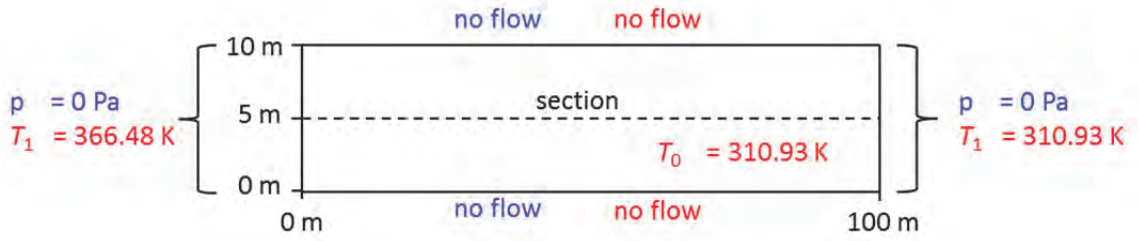


Fig. 11.1 Model geometry, initial and boundary conditions for the two-dimensional conductive heat transfer problem

Tab. 11.2 Parameter values for the Ogata-Banks model for purely convective heat transfer, modified after /THE 10/

Parameter	Symbol	Value	Unit
bulk thermal conductivity	$\overline{\lambda}$	2.16	J m^{-1} K^{-1} s^{-1}
specific heat capacity of solid	c_s	1254.682	J kg^{-1} K^{-1}
specific heat capacity of water	c_f	4185	J kg^{-1} K^{-1}
solid density	ρ_s	1602	kg m^{-3}
fluid density	ρ_f	1000	kg m^{-3}
matrix porosity	ϕ	0.1	(-)
longitudinal dispersivity	α_L	14.4	m
thermal dispersion coefficient	D_c	9.79 10^{-7}	m² s^{-1}
retardation coefficient	R	5.323	(-)
Darcy flux	q_f	0	m s^{-1}
retarded pore velocity	v_c	0	m s^{-1}
initial temperature	T_0	37.78	°C
boundary temperature	T_1	93.33	°C
domain size	L	300	m
output times	t1, t2	2148, 4262	d

Temperature profiles were extracted at two time steps along the dashed line illustrated in Fig. 11.1. Heat is transported diffusively into the area starting from the left boundary. During this process the temperature gradient becomes less steep (cp. Fig. 11.2). The numerical results show an excellent agreement with the analytical solution. This leads to the conclusion that d^{3}f reproduces the heat conduction process correctly.

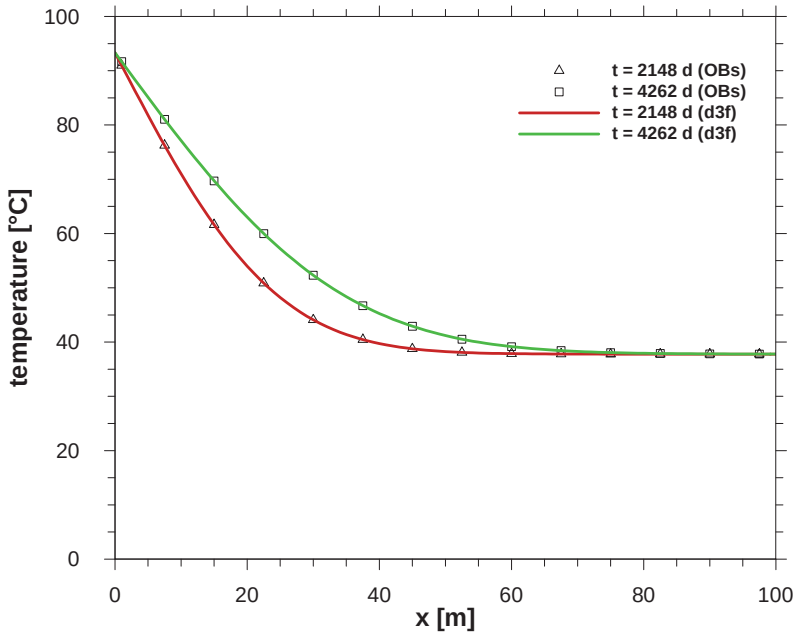


Fig. 11.2 Analytical solution (OBs) and the referring numerical results from d³f for the test of heat conduction

11.1.1.5 Test of the advective heat transport in d³f

After the pure heat conduction mechanism had been checked, the interaction of heat conduction, heat diffusion and thermal dispersion were investigated. The advection-diffusion problem set up for this purpose is equivalent to the one-dimensional heat flow problem described in section 11.1.1.2. The data listed in Tab. 11.1 was used as input for the d³f-simulation.

The one-dimensional problem formulated by Ogata and Banks was represented by a two-dimensional vertical model with a constant and strictly horizontal flow all over the model. Length and height amounted to 600 m and 10 m, respectively. Boundary conditions for temperature, pressure, concentration and flow velocity were assigned according to Fig. 11.3

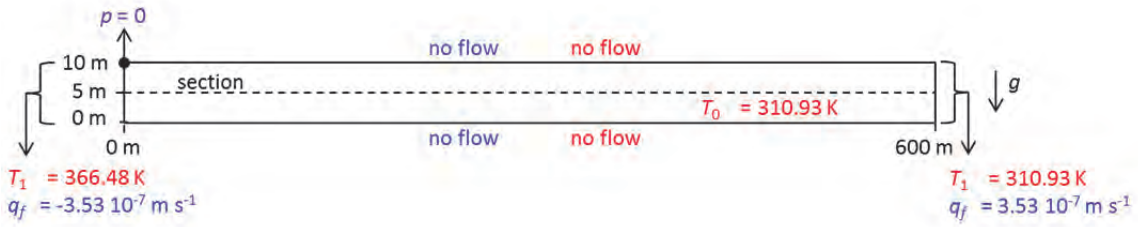


Fig. 11.3 Geometry, initial and boundary conditions for the Ogata-Banks model

Constant velocity was attributed to both vertical boundaries so that a steady flow from the left to the right was prescribed. At the top of the left hand side boundary, the pressure was set to zero as an anchor point for the pressure distribution. The upper and lower boundaries were impermeable both for the fluid and the temperature.

On the inflow boundary, the temperature was set to 366.48 K. The temperature on the outflow boundary equaled the initial temperature of 310.93 K within the area. Generally, a constant temperature at the outflow boundary is unphysical because arriving heat cannot leave the model domain. However, the more appropriate in-/ outflow boundary condition had not yet been implemented in d³f at the time of the simulation. The model was therefore run for a period of time in which the heat wave did not reach the outflow boundary yet as in earlier works like /WAR 84/ or /THE 10/. The outflow boundary condition did therefore not affect the shape of the curve.

At two points of model time temperature profiles were extracted along the dashed line drawn in Fig. 11.3. They are compared in Fig. 11.4 with the results from the OBS solution for which the retarded pore velocity and the thermal dispersion coefficient were used as input. Both simulated curves produced with d^{3}f show a very good agreement with the analytical solution. It can thus be concluded that d^{3}f reproduces the interaction of heat convection, heat conduction, and thermal dispersion correctly. Note that the shape of the curves did not change with a higher refinement level of the calculation grid.

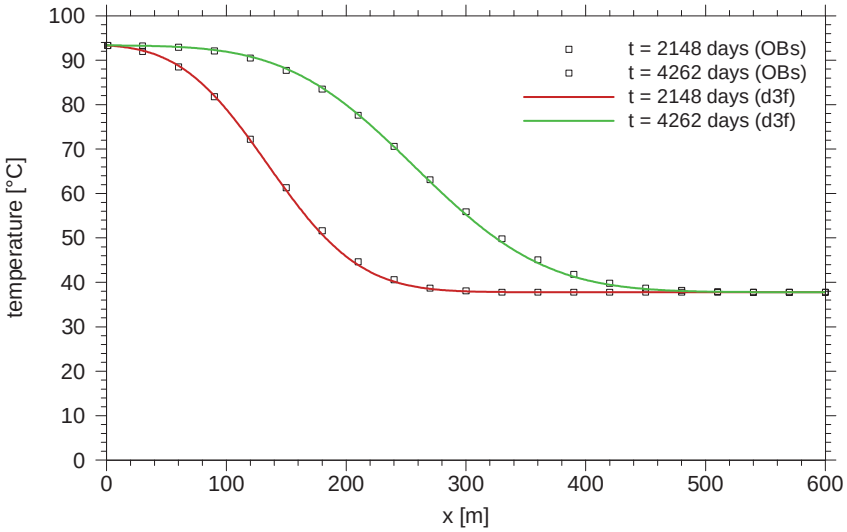


Fig. 11.4 Analytical solution (OBs) and referring numerical results from d³f

11.1.2 Test case heat transfer in anisotropic porous media

A 2D example including heat transfer with temperature-dependent fluid properties was presented by /YAN 00/. The example is based on geological observations at the White-shell Research Area in Southern Canada /DAV 94/ and represents the scenario of a radioactive waste disposal in low-permeability anisotropic granite. This test case has already been used to verify the numerical heat transport model HydroGeoSphere /THE 10/, /GRA 05/.

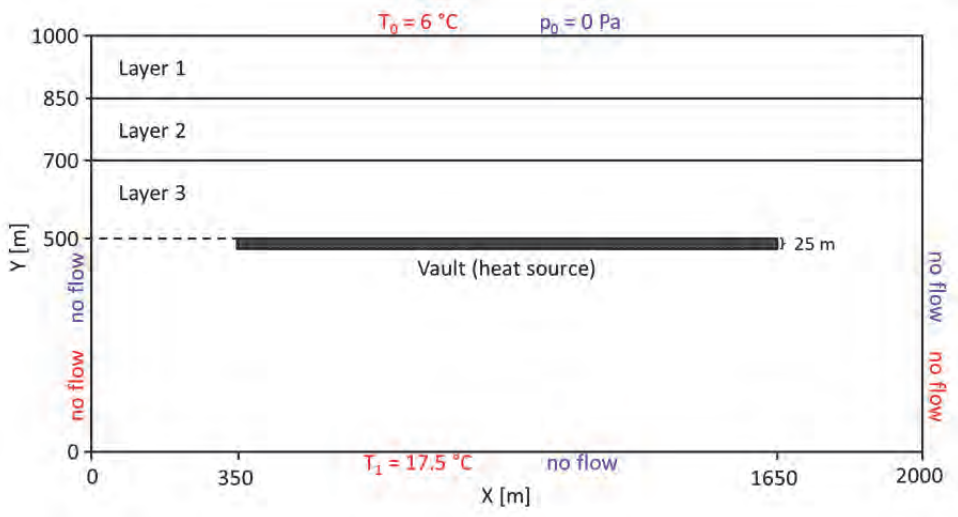


Fig. 11.5 Model geometry and boundary conditions for heat transfer in anisotropic porous media

11.1.2.1 Model description

The model region consists of a vertical cross-section of 2000 m × 1000 m including three different layers (cf. Fig. 11.5). The upper two layers are characterised by a moderate (Layer 1) to an intermediate (Layer 2) anisotropic permeability, whereas the permeability of the bottom layer (Layer 3) is very low and isotropic. In this low permeability layer a horizontal vault with a cross-sectional area of 1300 m × 25 m is positioned at a depth of 500 m. The vault is assumed to have the same thermal properties as the surrounding rock while being impermeable to water flow. The radioactive waste within the vault produces heat at an exponentially decreasing rate. The surficial heat output of the vault is described by the following function (cf. /DAV 94/):

$$\Gamma = 11.59 \exp(-5.5 \times 10^{-10} \cdot t) \tag{11.32}$$

where Γ is the heat flux in W m^{-2} and t the time in seconds. The temperature at the top boundary is set to 6 °C and at the bottom boundary to 17.5 °C, according to the average geothermal gradient of 11.5 °C km^{-1} at the Whiteshell Research Area /DAV 94/. The bottom boundary is assumed to be impermeable to flow, whereas a fixed head is assigned to the top boundary. A no-flow boundary condition for fluid flow as well as for heat flow is assigned to the vertical boundaries. All parameters used for the simulation are listed in Tab. 11.3.

Tab. 11.3 Parameter values for the Yang-Edwards model after /THE 10/

Parameter	Symbol	Value	Unit
Bulk thermal conductivity	$\bar{\lambda}$	2.0	kg m s^{-3} K^{-1}
Heat capacity of solid	c_s	800	m^2 s^{-2} K^{-1}
Heat capacity of water	c_f	4174	m^2 s^{-2} K^{-1}
Solid density	ρ_s	2630	kg m^{-3}
Matrix permeability	κ_{xx}, κ_{yy}	Layer 1: 1.0×10^{-15}, 5.0× 10^{-15} Layer 2: 1.0×10^{-17}, 5.0× 10^{-17} Layer 3: 1.0×10^{-19}, 1.0× 10^{-19} Vault: 1.0×10^{-25}, 1.0× 10^{-25}	m^2
Matrix porosity	ϕ	0.004	-
Domain size	L_x, L_y	2000, 1000	m
Output times	t1, t2, t3	10^4, 3×10^5, 7×10^6	d

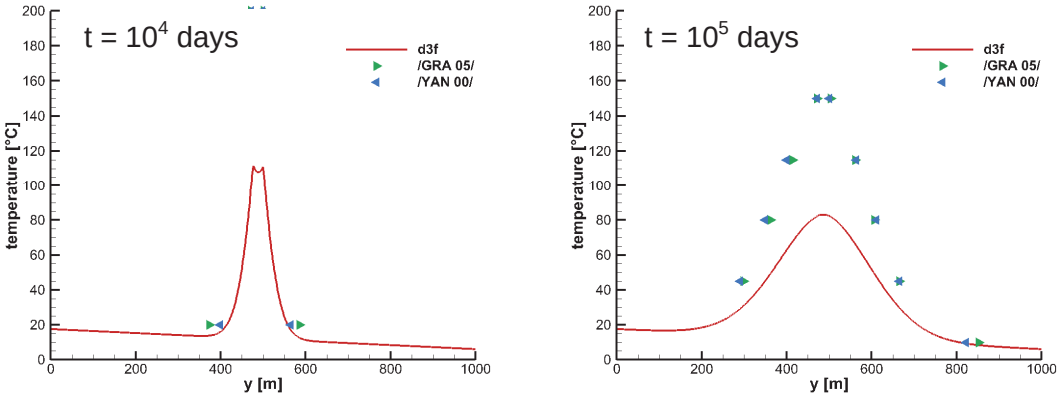


Fig. 11.6 Temperature profiles in the vertical symmetry axis of the model

11.1.2.2 Results and Discussion

In the d^3f-model, heat-producing waste was represented by line-shaped sources on all four surfaces of the vault emitting the heat as defined by eq. (11.32). Simulations with this configuration showed a poor match with the results from /YAN 00/ (cf. Fig. 11.6). Generally, the temperatures calculated with d^3f were considerably lower than those reported in /GRA 05/ and /YAN 00/. After 10^4 and after 10^5 days, the maximum temperatures in the d^3f-model were only half as high as the maximum temperatures in the models of /GRA 05/ and /YAN 00/. Apparently too little heat energy had entered the system in the d^3f-simulation. Therefore the fluxes emitted from the line-shaped heat sources were doubled in another simulation. The results using the modified heat source showed an excellent agreement with the results of /GRA 05/ and /YAN 00/ (cf. Fig. 11.7).

The temperature rose first in the vicinity of the heat sources. After 10^4 days, a temperature of 200 °C was reached at the edge of the vault accompanied by a steep negative temperature gradient towards the surrounding area. The temperature gradient lessened during the simulation due to a) the spread of the heat and b) the decreasing heat input. After 10^5 days the temperature at the vault amounted to 150 °C and after $3 \cdot 10^5$ only a temperature of 80 °C was reached in the proximity of the vault.

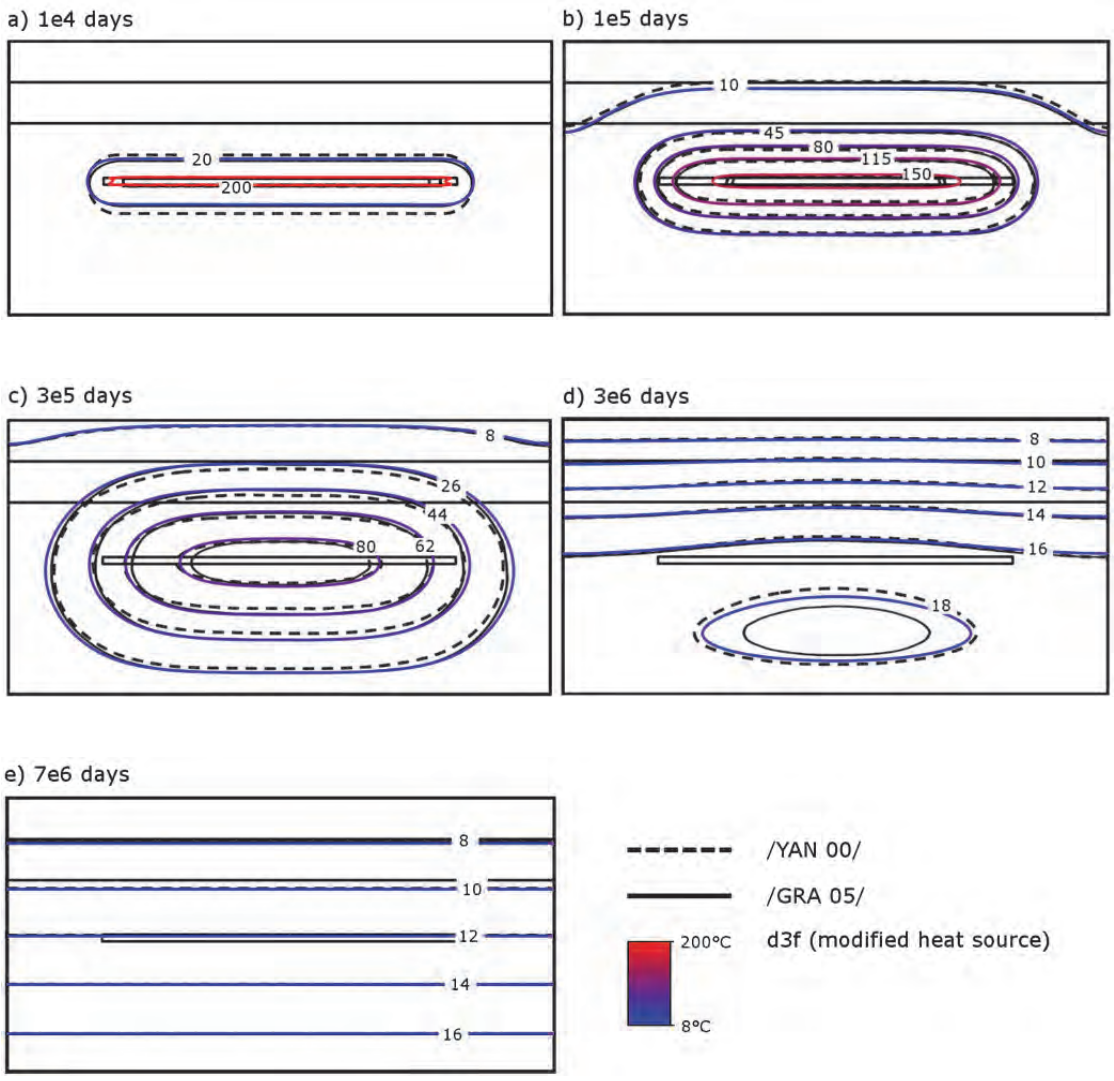


Fig. 11.7 Comparison of the temperature fields

After $3 \cdot 10^6$ days, the isoline of the maximum temperature forming an ellipsoidal shape was not located around the vault anymore but had moved downwards. The initial geothermal gradient was recovered after $7 \cdot 10^6$ days.

The flow field of the modelling system was plotted for two time steps to visualise effects of the heat dynamic on the hydraulics. It was expected that the strong heating at the beginning of the simulation would cause the fluid to expand and therefore to induce an upward flow above the vault. A downward flow was expected later in the simulation, when temperature at the vault would decrease again thus leading to an increase of water density. The flow fields after 10^4 days and 10^5 days show these effects (Fig. 11.8) and thereby demonstrate the coupling of heat transport and fluid flow in d^3f.

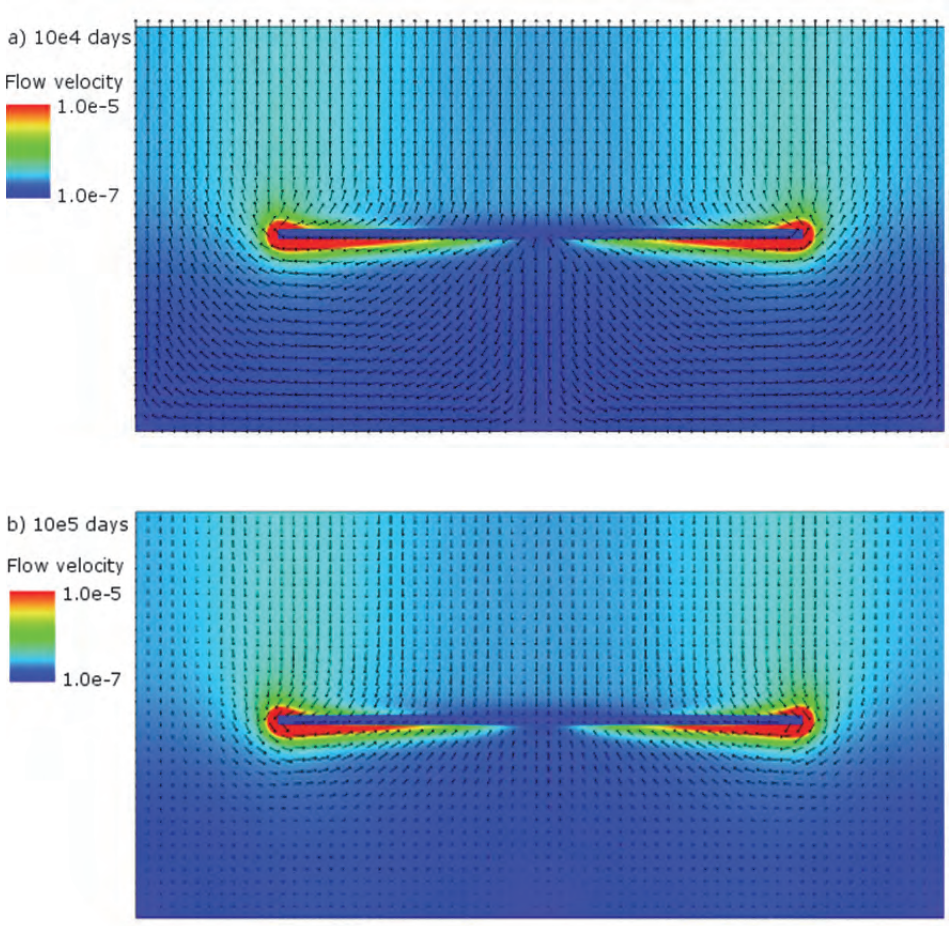


Fig. 11.8 Flow fields calculated with d^3f

Then the question remained whether the line heat sources were actually emitting the prescribed amount of heat energy. To answer this question the amount of heat energy in the model domain at a certain time was estimated and compared with the theoretical value based on the integrated total heat inflow rate over time. Both values were found to be in good agreement. Since the process of heat conduction was already qualified with the help of the Ogata-and-Banks model (see section 11.1.1) this result confirms correct handling of the line heat source in d^3f.

While working on this test case it turned out that /GRA 05/ had also found discrepancies between the heat source described in /YAN 00/ and the resulting temperature field (T. Graf, personal communication). In combination with the fact that no exact information was available about the way the heat source had been implemented in the model of /YAN 00/, this casts suspicion on the comparability of the heat sources in the different models in question. The worth of the model of /YAN 00/ as a test case appears therefore to be rather limited. Nevertheless, some conclusions can be drawn:

- Two-dimensional simulation of heat conduction with d^{3}f provides reasonable results.
- The calculated flow patterns stand to reason. Coupling of heat transport to fluid flow using temperature-dependent flow parameters thus appears to be correctly implemented.
- The resulting convection is very small, so no effect of the hydraulics on the thermal processes can be observed in this model.

11.2 Density-driven flow in fractured media

11.2.1 Test case „matrix diffusion“

11.2.1.1 Physical system

Several analytical solutions concerning the phenomenon of matrix diffusion have been developed since the early 1980's. Among them is the solution of /TAN 81/. Here, a fracture with the aperture *2b* in a porous rock as shown in Fig. 11.9 is considered. Flow velocity *v* in the fracture is assumed to be constant. More simplifying assumptions are listed in /TAN 81/ to enable the formulation of manageable analytical solutions:

- Complete mixing across the fracture width is assured by transverse diffusion and dispersion.
- Transport in the matrix is only due to molecular diffusion due to a very low permeability.
- Transport along the fracture is much faster than transport within the matrix.

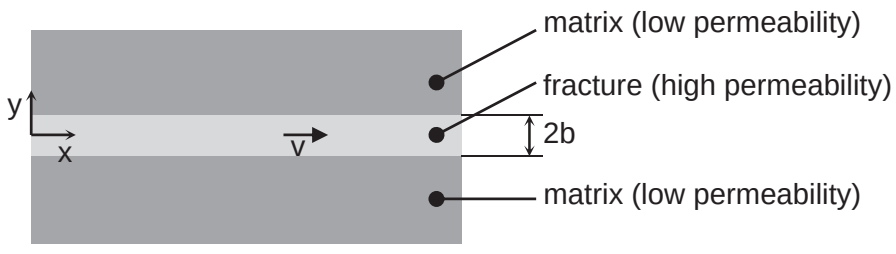


Fig. 11.9 Physical system underlying the analytical solutions of /TAN 81/

This setup allows to describe solute transport by two one-dimensional equations - one for solute transport in the fracture and one for solute transport in the matrix - which are coupled at the fracture surface. The directions of the two referring mass fluxes are assumed to be orthogonally orientated. Diffusion in the matrix in the direction of the fracture is thereby neglected.

In case of the solution of /TAN 81/ the following processes are considered:

- in the fracture
  - advective transport along the fracture
  - longitudinal mechanical dispersion
  - molecular diffusion
  - adsorption at the fracture surface
  - radioactive decay
- in the matrix
  - molecular diffusion
  - adsorption within the matrix
  - radioactive decay

Solute is injected with a constant rate at the origin of the coordinate system providing a maximum solute concentration of c_0 at the point of injection. Consequently a transient solute distribution develops at the beginning. However, the combined effects of all processes considered lead eventually to a steady-state distribution in fracture and matrix.

11.2.1.2 General transient solution

In the solution of /TAN 81/ longitudinal mechanical dispersion and molecular diffusion in the fracture are combined to the hydrodynamic dispersion expressed by the dispersion coefficient D_f

$$D_f = \alpha_l v + D_{mol} \tag{11.33}$$

D_f - coefficient of hydrodynamic dispersion in the fracture [m² s^{-1}]

α_l - longitudinal dispersion length [m]

v - flow velocity in the fracture [m s^{-1}]

D_{mol} - molecular diffusion coefficient in water [m^2 s^{-1}]

This formulation appears to be incomplete because it ignores tortuous effects in the fracture on the molecular diffusion. Eq. (11.33) is thus modified here by a fracture tortuosity τ_f:

$$D_f = \alpha_l v + \tau_f D_{mol} \tag{11.34}$$

τ_f - tortuosity of the fracture [-]

This modification does not affect the analytical solutions since only the coefficient of hydrodynamic dispersion enters these solutions. Radioactive decay is characterized by the decay constant λ which is defined as

$$\lambda = \frac{\ln(2)}{t_{1/2}} \tag{11.35}$$

λ - decay constant [1 s^{-1}]

$t_{1/2}$ - half life [s]

The effect of linear sorption in the fracture is introduced by a retardation coefficient R_f according to

$$R_f = 1 + \frac{K_f}{b} \tag{11.36}$$

R_f - retardation coefficient for the fracture [-]

K_f - distribution coefficient for the fracture [m]

b - half of the fracture width [m]

Note: The dimension of the distribution coefficient K_f is defined as the ratio of concentration of adsorbed solute c_a to the concentration of solute in the solution c. From this definition it is clear that the dimension of K_f depends on the dimensions in which the two above mentioned concentrations are given. While solute concentration c is always treated as a volume concentration in /TAN 81/ the adsorbed solute concentration is

given as solute mass per surface area in case of the fracture. In case of the matrix solute concentration is given as solute mass per mass of matrix.

In the matrix the retardation coefficient R_m is thus calculated differently:

$$R_m = 1 + \frac{\rho_b}{\Phi_m} K_m \tag{11.37}$$

R_m - retardation coefficient for the matrix [-]

ρ_b - bulk density of the matrix [kg m^{-3}]

ϕ_m - matrix porosity [-]

K_m - distribution coefficient for the matrix [m^3 kg^{-1}]

In order to include the effect of tortuosity in the matrix explicitly an effective diffusion coefficient D_m is defined for the matrix:

$$D_m = \tau_m \, D_{mol} \tag{11.38}$$

D_m - effective diffusion coefficient [m^2 s^{-1}]

τ_m - tortuosity of the matrix [-]

The solution for the concentration in the matrix reads

$$\frac{c}{c_0} = \frac{e^{(\mu x)}}{\sqrt{\pi}} \int_l^{\infty} \left[\begin{array}{l} e^{\left(-\xi^2 - \frac{\mu^2 x^2}{4\xi^2} \right)} e^{\left(-\eta x^2 \right)} \\ \cdot \left\{ e^{\left(-\sqrt{\lambda} Y \right)} erfc\left(\frac{Y}{2T} - \sqrt{\lambda} T \right) + e^{\left(+\sqrt{\lambda} Y \right)} erfc\left(\frac{Y}{2T} + \sqrt{\lambda} T \right) \right\} \end{array} \right] d\xi \tag{11.39}$$

c - solute concentration [$kg_{solute}/kg_{solution}$]

c_0 - maximum solute concentration [-]

μ - supplementary parameter (q. v. (11.40))

l - supplementary variable (q. v. (11.41))

η - supplementary variable (q. v. (11.42))

T - supplementary variable (q. v. (11.43))

Y - supplementary variable (q. v. (11.44))

x - coordinate [m]

ξ - integration variable [-]

$$\mu = \frac{v}{2D_f} \tag{11.40}$$

$$l = \frac{x}{2D_f}\sqrt{\frac{R_f}{D_f t}} \tag{11.41}$$

t - time [s]

$$\eta = \frac{\lambda R_f}{4D_f \xi^2} \tag{11.42}$$

$$T = \sqrt{t - \frac{R_f\, x^2}{4D_f \xi^2}} \tag{11.43}$$

$$Y = \frac{\mu^2 \beta^2 x^2}{4A\xi^2} + B(y-b) \tag{11.44}$$

β - supplementary parameter (q. v. (11.45))

A - supplementary parameter (q. v. (11.46))

B - supplementary parameter (q. v. (11.47))

$$\beta^2 = \frac{4R_f D_f}{v^2} \tag{11.45}$$

$$A = \frac{bR_f}{\phi_m \sqrt{R_m D_m}} \tag{11.46}$$

$$B = \sqrt{\frac{R_m}{D_m}} \tag{11.47}$$

The solution for the concentration in the fracture can be found by setting $y = b$.

11.2.1.3 Numerical model

11.2.1.3.1 Inconsistencies between analytical solution and numerical model

Several simplifying assumptions underlie the analytical solution of /TAN 81/ as described above. They are not necessarily the same as those that were used for the development of the codes d³f and r³t:

The assumption of pure diffusive transport in the matrix cannot be reproduced exactly with the numerical codes. However, choosing a very low permeability for the matrix in comparison to fracture permeability yields a sufficiently close approximation.

The restriction that transport along the fracture is much faster than transport within the matrix is only attributed to the analytical solution. This assumption provides the basis for approximating diffusive transport in the matrix as a one-dimensional process that runs exclusively orthogonally to the fracture /TAN 81/. Such a consideration about the ratio of process velocities is not required for the numerical model. But of course it has nevertheless to be taken into account for the definition of a test case forming the basis of a comparison between analytical and numerical solution.

11.2.1.3.2 Model setup

In /TAN 81/ the analytical solution is evaluated based on two sets parameters that differ only in the flow velocity in the fracture. The high velocity case was adopted here. Due to the extensive fracture description in d³f some additional parameters were required. All parameters are listed in Tab. 11.4.

A two-dimensional domain is defined as shown in Fig. 11.10. No-flow boundary conditions are assigned to all boundaries except where the fracture is located. Here, inflow and outflow velocity, respectively, are prescribed. In order to prevent significant flow from the fracture into the matrix (and vice versa) the permeability in the matrix is chosen to be nine orders of magnitude lower than the permeability in the fracture. To the inflow boundary of the fracture a solute concentration of 1 is assigned. All other boundaries show a “concentration out”-condition. Initially, the whole domain contains no solute.

Tab. 11.4 Parameters for the matrix diffusion model according to /TAN 81/, additional parameters for d³f are written in blue

Quantity	Dimension	Value
fracture		
length of fracture (*x*-direction)	m	7.5
width of fracture (*2b*)	m	0.0001
flow velocity in the fracture	m d^{-1}	0.01
longitudinal dispersion length	m	0.5
tortuosity	-	1
retardation coefficient	-	1
porosity		1
isotropic permeability	m^2	10^{-10}
orthogonal permeability	m^2	10^{-10}
orthogonal tortuosity	-	1
transversal dispersion length	m	0.01
matrix		
depth of the matrix (*y*-direction)	m	1.20
porosity	-	0.01
tortuosity	-	0.1
retardation coefficient	-	1.0
isotropic permeability	m^2	10^{-19}
longitudinal dispersion length	m	0.5
transversal dispersion length	m	0.01
fluid		
diffusion coefficient	m^2 s^{-1}	1.6 10^{-9}
density	kg m^{-3}	1000
viscosity	Pa s	0.001
solute		
half life	a	1.2 10^{21}
end of simulation	a	5

11.2.1.3.3 Results

The comparison of analytical and numerical solution is given in terms of solute distributions in the fracture as well as in the matrix at 1, 3, and 5 years simulation time. Fig. 11.11 depicts the concentration in the fracture calculated from the analytical solution and with d^3f. Minimal deviations in the solutions for 1 year disappear apparently with time.

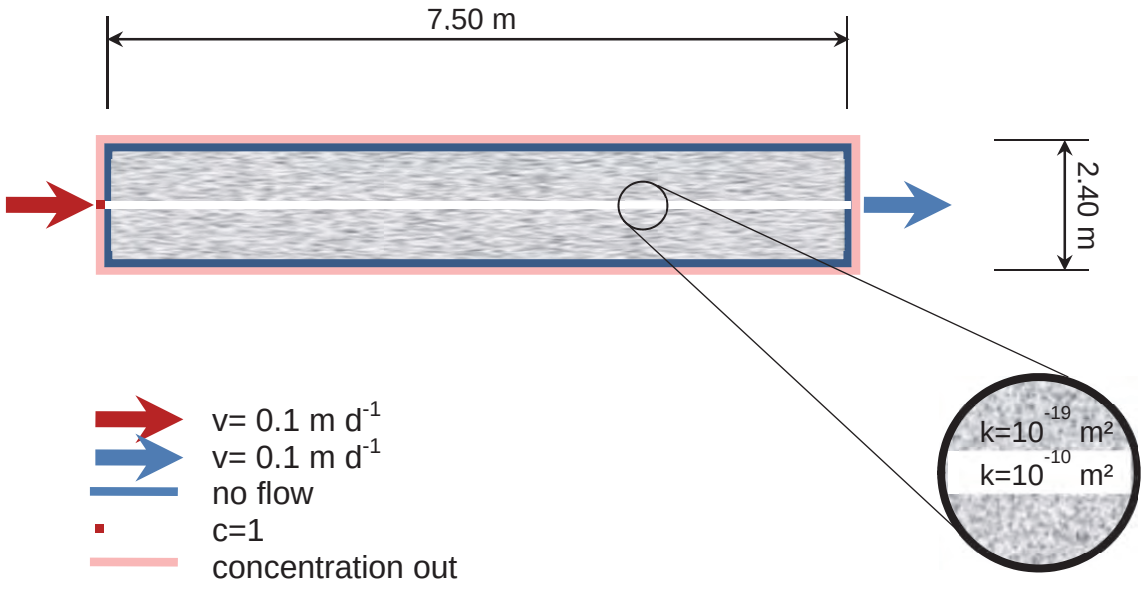


Fig. 11.10 Geometry and boundary conditions for the numerical model

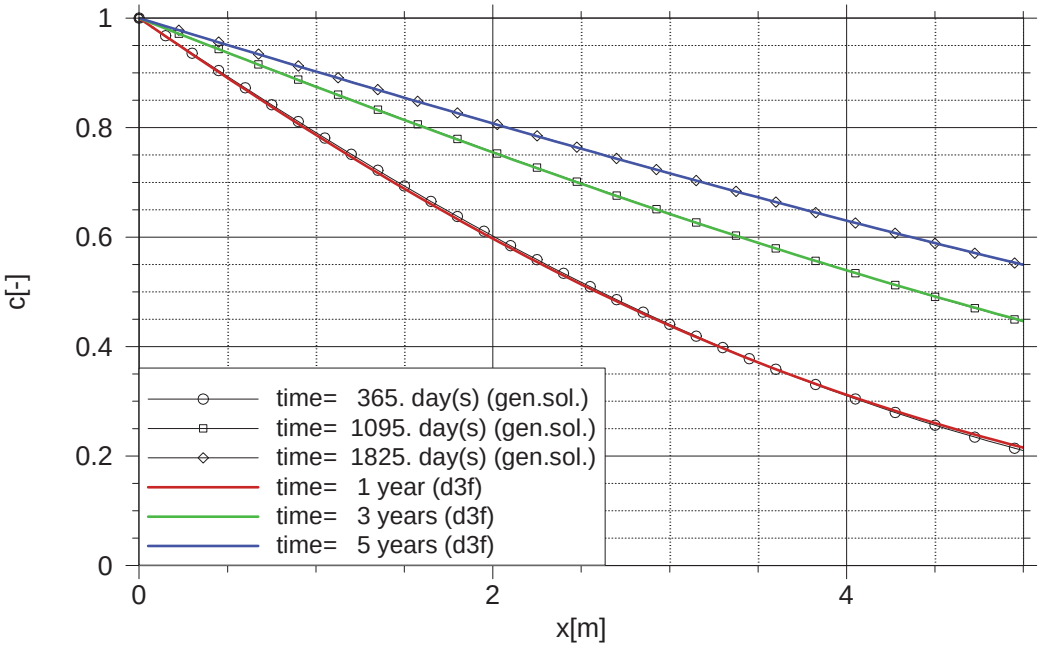


Fig. 11.11 Concentration in the fracture: analytical and numerical solution

In Fig. 11.12 the contour lines for the analytical solution are drawn in black, the contour lines from the d^{3}f-model are colour coded. In all cases the match is satisfying. Only at the left hand side boundary some systematic deviations can be observed. While the analytical solution predicts straight concentration contour lines d^{3}f-results provide lines that are rounded out in such a way that they connect orthogonally with the boundary.

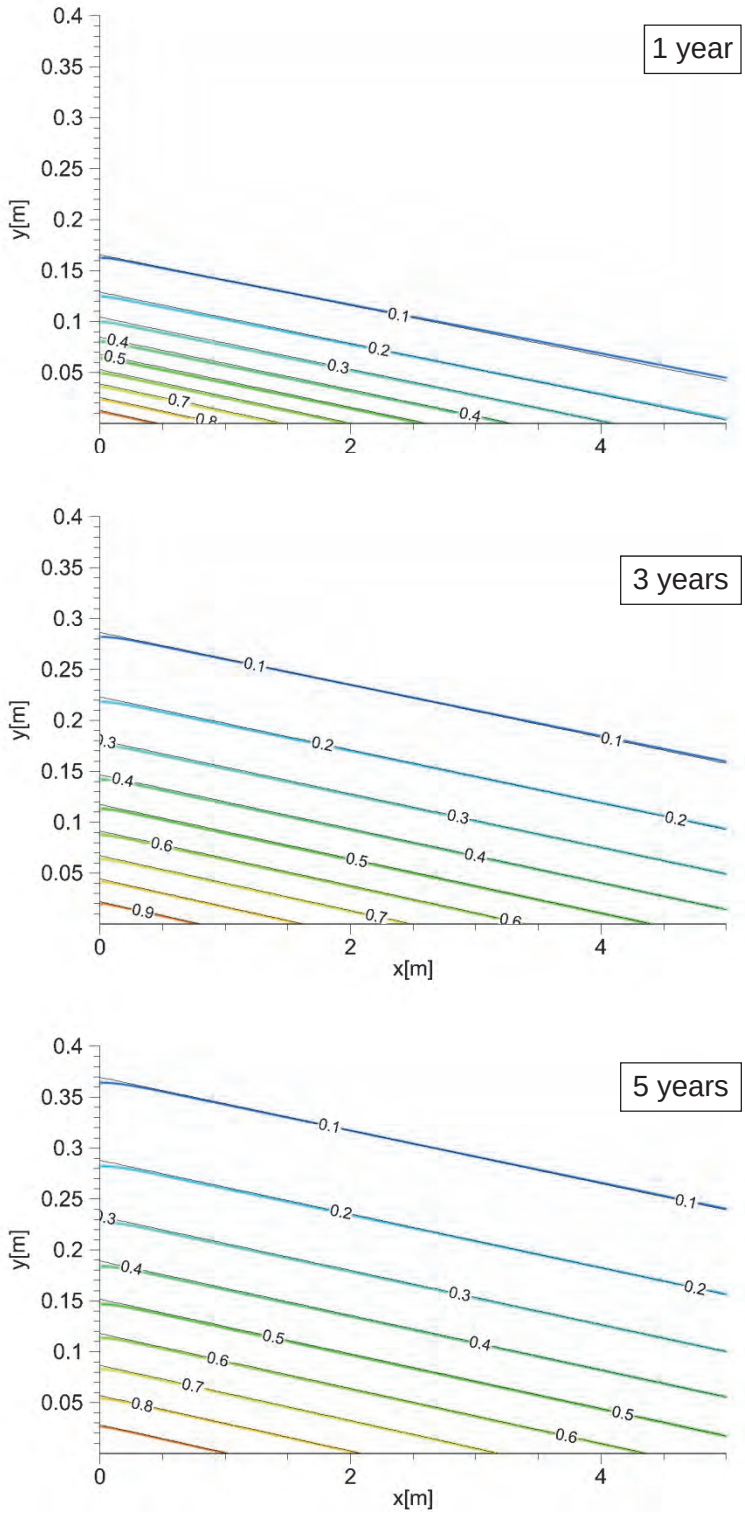


Fig. 11.12 Concentration in the matrix after 1, 3, and 5 years;
black lines: analytical solution, coloured lines: numerical solution

This little deviation is the consequence from neglecting diffusion in the matrix in the x-direction (direction of the fracture). The analytical as well as the numerical solution agree in that the concentration gradient is not oriented exactly in the y-direction (orthogonally to the fracture) but turned a bit in the direction of fracture flow. But the two solutions differ in the way these gradients are evaluated in terms of diffusive mass flux. This becomes evident at the inflow boundary of the fracture. Transport of solutes entering the matrix at this point is strictly in the y-direction according to the analytical solution. But real diffusion has also a component in the x-direction. In other words, the solute flux in the matrix originating right from the fracture at the inflow boundary spreads out and thereby reduces the flux component in the y-direction. The analytical solution thus overestimates the concentration at the left hand side boundary of the model letting the results of d^3f appear to be more realistic.

11.2.2 Testcase Majak

Model calculations for a two-dimensional, vertical groundwater flow and transport model were conducted in the course of the project ASTER) /WAL 05/ using the code FEFLOW 5 /DIE 04/. The investigation area is located in the area of Chelyabinsk, southeast of the Ural Mountains in the West Siberian Plateau. Exploratory analyses were conducted concerning the storage of radioactive waste in a fractured, porphyric host rock formation.

11.2.2.1 Site description

Flow and transport calculations are based on a schematic cross section through the area (cf. Fig. 11.13 /WAL 05/). The fractured porphyric rock can be divided into a lower zone with very low water circulation and an upper zone with low water circulation. A weathering zone forms the transition to a layer of clay and gravel at the surface. The main structural characteristic of the cross section is the intensive fracturing of the two lower layers with different dip directions and dip angles of the fractures.

The clay and gravel in the upmost layer as well as the weathering zone are subject to the main groundwater circulation in the model area, which is indicated by the blue arrow in Fig. 11.13. Groundwater flow and transport of contaminants within the lower fractured porphyric units is mainly bound to the fracture network, shown by the black arrows in Fig. 11.13.

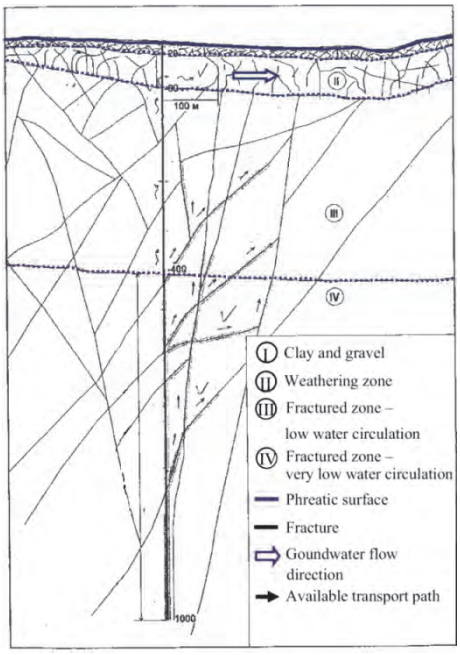


Fig. 11.13 Schematic cross section through the investigation area after /WAL 05/

11.2.2.2 Model set-up

A schematic two-dimensional model was set up based on the cross section given in Fig. 11.13. Model calculations presented in the ASTER final report /WAL 05/ were reviewed and adapted. Updated model calculations were conducted using the code FEFLOW Classic 6.0 /DHI 10/.

11.2.2.2.1 Geometry and model grid

The model represents an area of about 814 m width and of about 1057 m depth (cf. Fig. 11.14).

The four geological units as well as five explicit fractures (F1 to F5) are described by the model set-up. The fractures can be traced through the two lower units of porphyric rock and are dipping at different angles between 50 and 90 degrees. They do not intersect with the model boundaries. The coarse grid was generated using the code ProMesh 3.2, see section 9.2. It contains 660 vertices, 1710 edges and 1051 faces. For the simulations it was refined to 21,310 vertices, 84,898 edges and 55,840 elements. The FEFLOW model consists of 45,176 nodes and 89,356 elements.

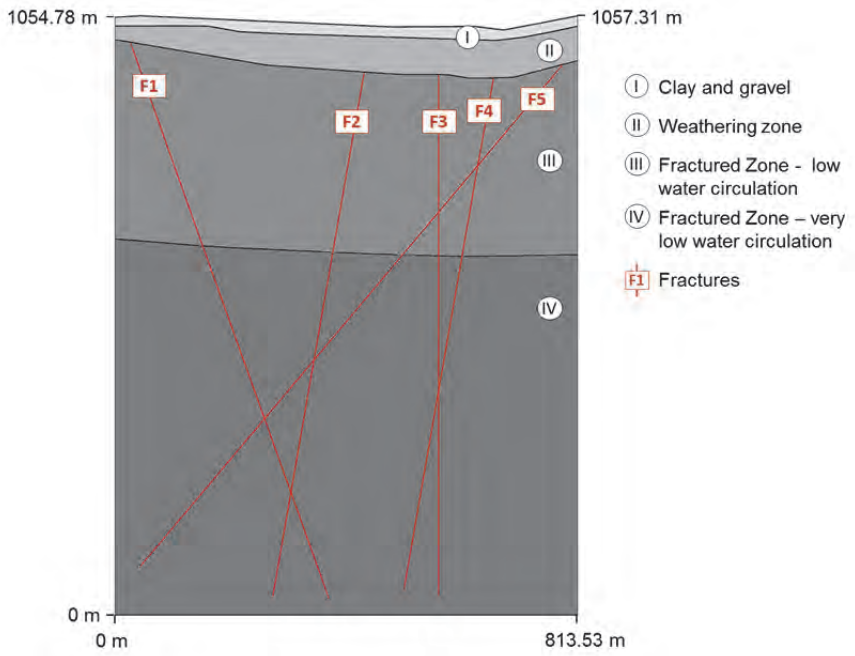


Fig. 11.14 Model geometry based on the schematic cross-section with identifiers of the units and fractures

11.2.2.2.2 Groundwater flow simulations

Hydrogeological parameters were assigned to the flow model according to the ASTER final report. Some of the parameters were modified to meet more common geological settings. Parameters used to calculate the groundwater flow without considering heat transport are given in Tab. 11.5 and Tab. 11.6. The simulations were run assuming a steady-state flow in a saturated medium with a confined groundwater table.

Tab. 11.5 Hydrogeological parameters for the four geological units

ID	Geological unit	Conductivity [m s^{-1}]	Permeability [m^2]
I	Clay and gravel	$2.08 \cdot 10^{-5}$	$2.08 \cdot 10^{-12}$
II	Weathering zone	$5.79 \cdot 10^{-6}$	$5.79 \cdot 10^{-13}$
III	Fractured zone (low water circulation)	$1.16 \cdot 10^{-10}$	$1.16 \cdot 10^{-17}$
IV	Fractured zone (very low water circulation)	$1.16 \cdot 10^{-14}$	$1.16 \cdot 10^{-21}$

Tab. 11.6 Hydrogeological parameters for the five fractures

Parameter	Fractures
Aperture [m]	$1.0 \cdot 10^{-2}$
Conductivity [m s^{-1}]	$7.52 \cdot 10^{-6}$
Permeability [m^2]	$7.52 \cdot 10^{-13}$

The initial salt concentration was set to 0 kg m^{-3}, according to the very low groundwater mineralisation. The temperature was globally set to 293.15 K for first model calculations. The boundary conditions for the d³f simulations are illustrated in Fig. 11.15. The groundwater recharge of $1.31 \cdot 10^{-9}$ m s^{-1} was converted to $1.13184 \cdot 10^{-4}$ m d^{-1} for the FEFLOW simulations. The pressure boundary conditions in d³f were chosen to match the constant hydraulic heads of 1048.8 m at the left and 1048.4 m at the right model boundary assigned in FEFLOW. The conversion was based on

$$P = \rho g(h - y) \tag{11.48}$$

with h = hydraulic head [m], P = pressure [Pa], ρ = fluid density [kg m^{-3}], g = gravitation vector [m^{-2}], and y = height [m] of the considered point in the model domain.

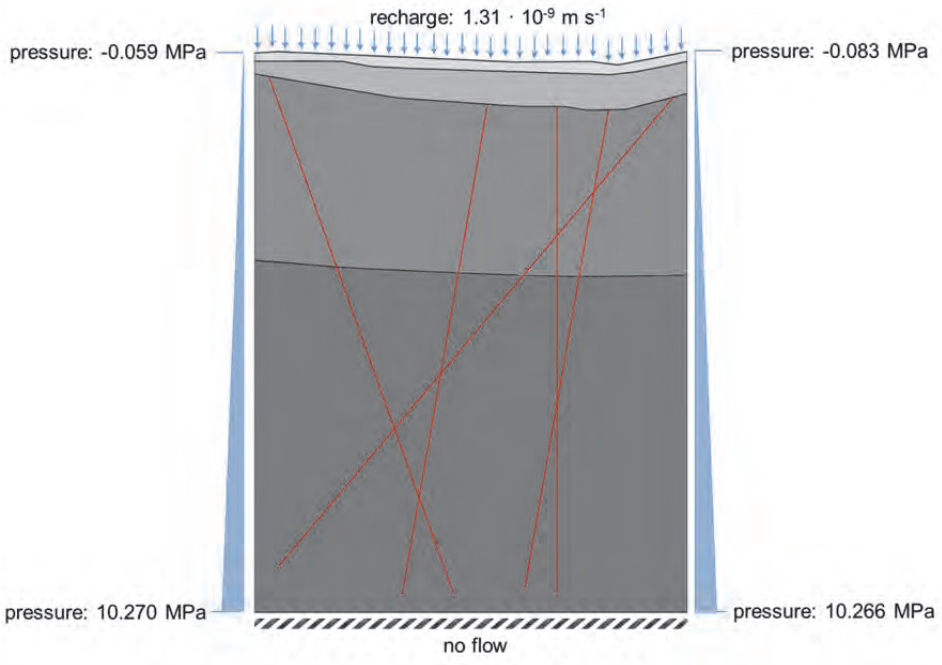


Fig. 11.15 Boundary conditions for the groundwater flow simulations with d³f

Both the FEFLOW and the d^{3}f model were run assuming steady-state flow conditions. The resulting pressure distribution, flow velocities and flow fields are used to evaluate and to compare the model results.

According to the total depth of the model domain the pressure difference between the top and the bottom of the model domain amounts to $1.03 \cdot 10^4$ kPa (cf. Fig. 11.16). The FEFLOW and d^{3}f model show a very good agreement.

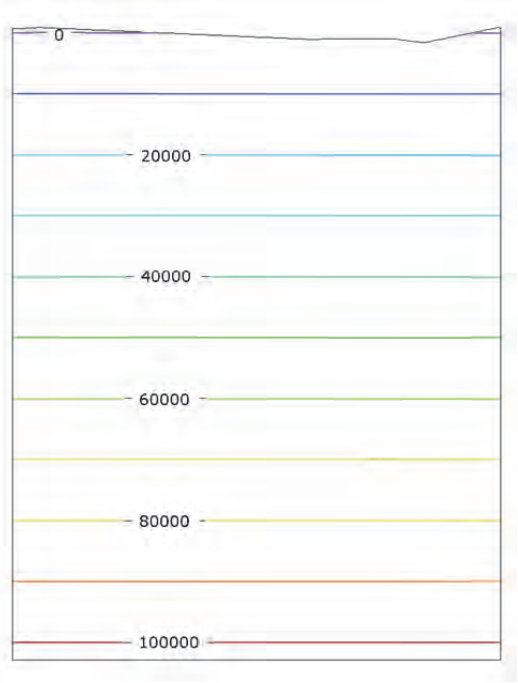


Fig. 11.16 Pressure distribution in the model area in [kPa]
Coloured isolines were computed with d^{3}f and black isolines were computed with FEFLOW.

The highest flow velocities can be observed in the upper two units, whereas low and very low flow velocities occur in unit III and IV, respectively. The groundwater flow velocities in the matrix mainly correspond to the permeabilities that were assigned to the different hydrogeological units. The strongest deviations from this pattern can be observed in unit IV where exceptionally low flow velocities occur between the bottom parts of the fractures.

The flow velocity ranges from d^{3}f and FEFLOW simulations given in Tab. 11.7 as well as the velocity distributions (Fig. 11.17) show a very good agreement. The flow velocities in the upper three units range within the same orders of magnitude and also their

spatial distribution agrees quite well. Only between the fractures in deepest layer the velocities calculated by d^{3}f are two orders of magnitude smaller than the velocities calculated by FEFLOW.

Tab. 11.7 Velocity ranges for the different units from d^{3}f and FEFLOW simulations

	v_{min} (d^{3}f)	v_{max} (d^{3}f)	v_{min} (FEFLOW)	v_{max} (FEFLOW)
Unit I	$2.6 \cdot 10^{-8}$ m s^{-1}	$6.8 \cdot 10^{-10}$ m s^{-1}	$2.9 \cdot 10^{-8}$ m s^{-1}	$6.5 \cdot 10^{-10}$ m s^{-1}
Unit II	$1.3 \cdot 10^{-8}$ m s^{-1}	$2.5 \cdot 10^{-11}$ m s^{-1}	$9.5 \cdot 10^{-9}$ m s^{-1}	$1 \cdot 10^{-11}$ m s^{-1}
Unit III	$2 \cdot 10^{-13}$ m s^{-1}	$5 \cdot 10^{-16}$ m s^{-1}	$2 \cdot 10^{-13}$ m s^{-1}	$8 \cdot 10^{-16}$ m s^{-1}
Unit IV	$4 \cdot 10^{-17}$ m s^{-1}	$2.5 \cdot 10^{-22}$ m s^{-1}	$9 \cdot 10^{-17}$ m s^{-1}	$3 \cdot 10^{-20}$ m s^{-1}

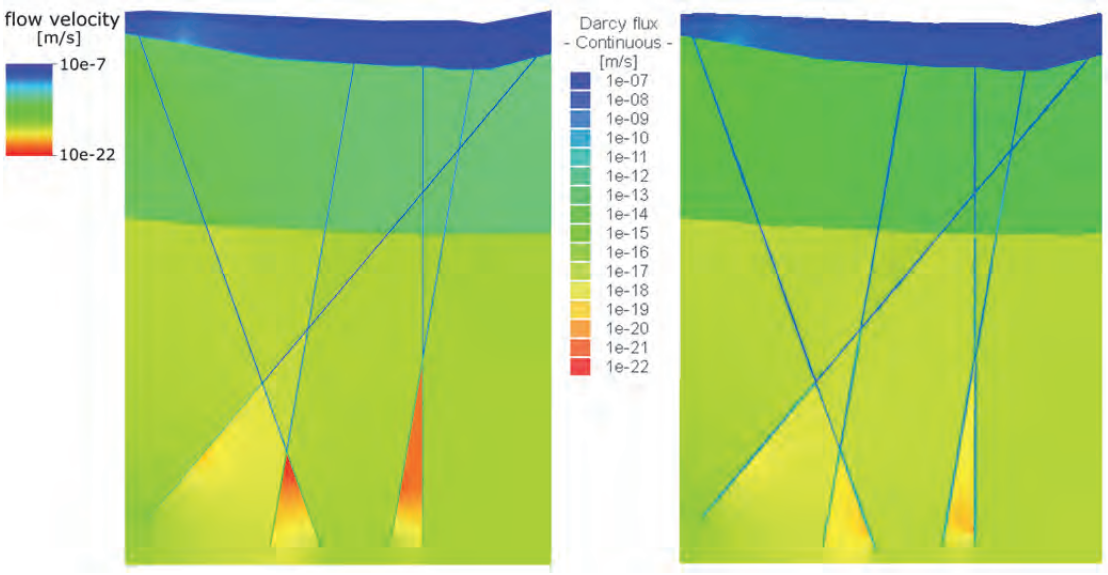


Fig. 11.17 Groundwater flow velocity in the matrix and in the fractures on a logarithmic scale (colours) calculated by d^{3}f (left) and FEFLOW (right)

Groundwater flow directions and velocities in the matrix and the fractures are presented exemplarily from d^{3}f simulations (cf. Fig. 11.18). Due to the groundwater recharge, a downward flow can be observed in the upper two units of the model. The streamlines are partly deflected at the boundary to the subjacent, less permeable layer and run in wide parts parallel to the layer boundary. In the first about 100 m the groundwater leaves to both sides of the model area. Below that depth, the main flow direction is not driven by the groundwater recharge but by the pressure boundary conditions specified

for both sides. This flow field to the right, however, is modified by the influence of the fractures. Especially in the lower permeable units, the velocity vectors shown in Fig. 11.18 get deflected by the higher permeable fractures, e. g. a flow towards fracture F5 occurs in the bottom left corner of the model (nomenclature of the fractures cf. Fig. 11.14).

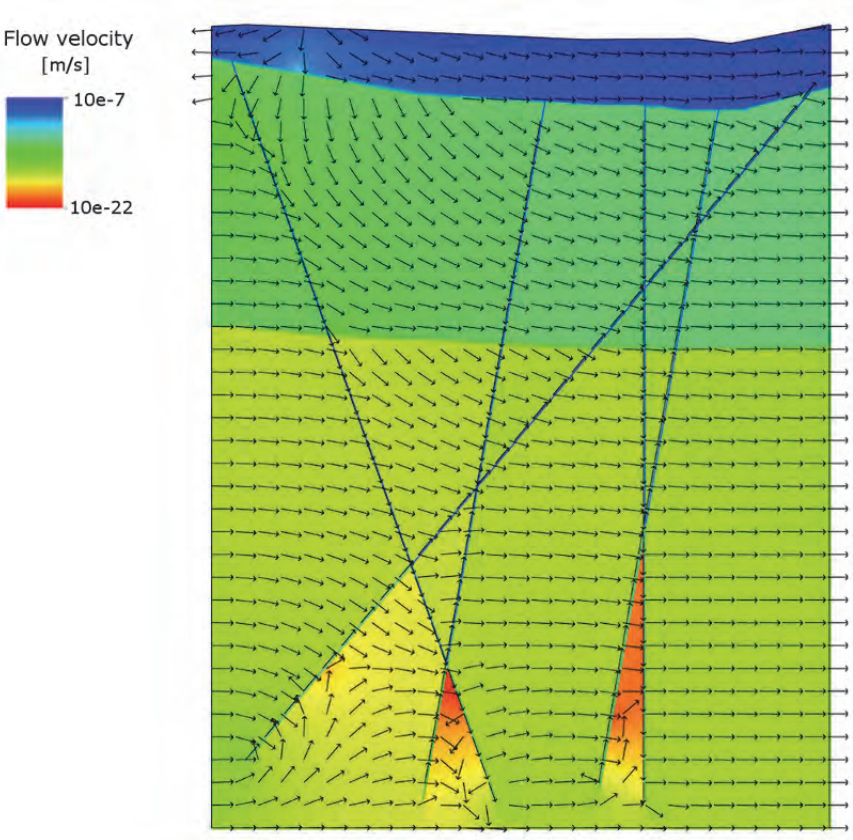


Fig. 11.18 Groundwater flow directions and velocities in the matrix and the fractures calculated with d^{3}f. Velocity vectors are scaled to the same size

A zone of very low flow velocities can be observed in the wedge below the intersection of the fractures F1 and F2. The groundwater coming from the left is led upwards by fracture F2 and only a small proportion of the water crosses the fracture. The area to the right of this fracture is also only slightly fed from the other side because the groundwater flowing downwards through the fracture F1 runs mainly off to the right following the pressure gradient imposed by the boundary conditions. The same situation applies to the wedge below the intersection of the fractures F3 and F4.

The flow field calculated by FEFLOW shows a good agreement with the d^{3}f flow field described above. Beside the same general flow direction the FEFLOW results show

the same prominent characteristics: a) the change between the outflow and inflow on the left boundary occurs approximately at the same location, b) fracture F5 acts as drainage in the bottom left corner of the model area and c) the zones of very low flow velocities below the intersecting fractures F1/F2 and F3/F4 is also represented by the FEFLOW simulation although this effect is less pronounced.

Results from d^3f-calculations for flow directions and velocities in the fracture system are shown in Fig. 11.19. The flow velocities vary between 10^{-15} and 10^{-8} m s^{-1} reaching higher flow velocities in the upper parts of the fractures than in the lower parts. The main flow path through this system is formed by fractures F1 and F5. Water from the relatively highly permeable unit II is transported downwards through fracture F1 until the intersection with fracture F5. This fracture is dipping at a low angle into the direction of the gradient the pressure boundaries impress upon the model area and therefore leads the water back upwards to unit II. Fractures F2, F3, and F4 also draw down water from unit II and supply fracture F5 with it.

The groundwater in the factures partially bypasses parts of F5 by flowing downwards through a fracture dipping to the right (F1 and F3) and then upwards again through an intersecting fracture that dips to the left (F2 and F4). Along these paths the flow velocity stays relatively constant.

In the bottom part of the fractures, i. e. the part below the lowest intersection with another fracture, velocities occur that are three to four orders of magnitude lower than in the remaining fracture system. This is because only a small amount of water can be drawn from or pressed into the low permeable matrix the fractures end in.

FEFLOW simulations produce a flow system in the fractures that is very similar to that in d^3f. The main flow path with comparable flow velocities is the same as in d^3f. Also the drawing and bypassing behaviour of fractures F1 to F4 is equal. Differences are found in the bottom part of the fractures. Here, the flow velocities lie two to three orders above the flow velocities calculated with d^3f.

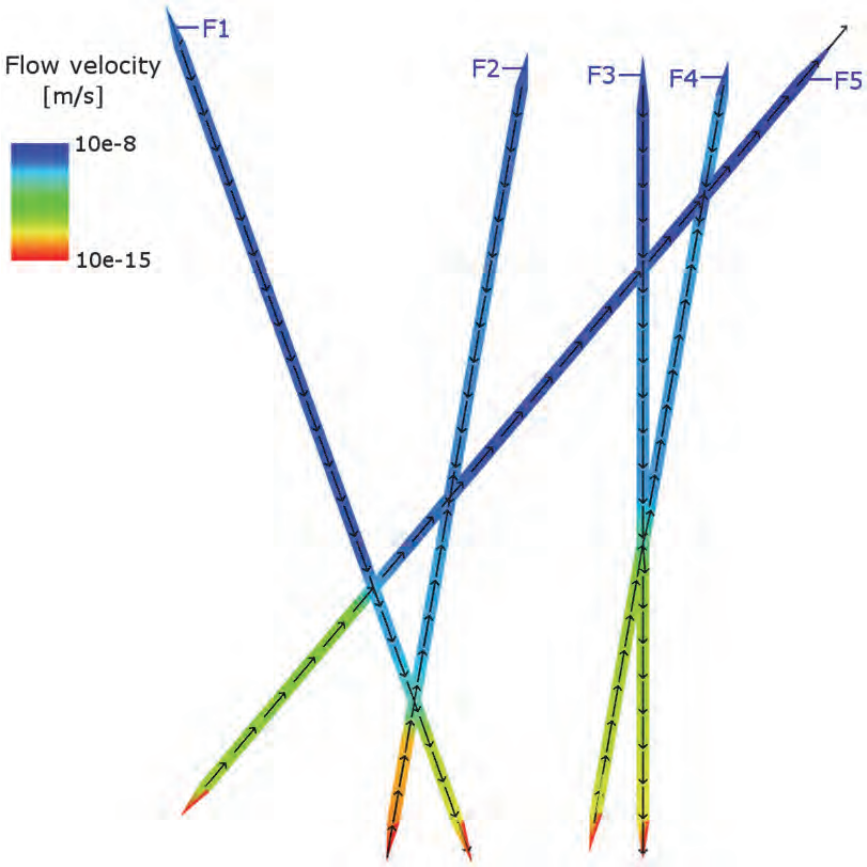


Fig. 11.19 Groundwater flow directions and velocities in the factures calculated with d^3f. Velocity vectors are scaled to the same size

FEFLOW also shows a different behaviour concerning the flow direction in the fractures. In the bottom part of fracture F2 the streamlines point alternatingly upwards and downwards with no clear trend. In the bottom part of fracture F4 the FEFLOW streamlines point in the opposite direction to the d^3f streamlines.

Differences between the results of d^3f and FEFLOW simulations had to be expected due to the different treatment of the fractures. In d^3f jumps of physical quantities from one side of a fracture to the other can be handled because a fracture is represented by three values orthogonal to its plane: one value for each interface with the matrix and one for the middle of the fracture. Nevertheless, the fracture is considered to be (d-1)-dimensional compared to the d-dimensional model. In FEFLOW such discontinuities of physical quantities cannot be represented as only one value is assigned to each point of the fracture plane. Coupling between the processes on both sides of the fracture is thus stronger in the FEFLOW model.

In summary, the following results were obtained concerning the groundwater flow simulation:

- The results of the d^3f- and FEFLOW-simulations show a satisfactory agreement. Some differences had to be expected due to the differences in the implementation of the fractures in the two program codes. Good agreement is obtained concerning the values and the distribution of the flow velocities in the different layers. Only small differences are found in unit IV between fractures F1 and F2 as well as between fractures F3 and F4.
- The main characteristics of the flow field in the matrix calculated with FEFLOW correspond to the results from the d^3f simulations. However, influence of the fractures on the flow in the matrix is less prominent in the FEFLOW model. The results of d^3f show pronounced areas of very low velocities in the wedge between two fractures.
- Results for the flow in the upper part of the fracture system are in good agreement. In the lower third, however, where velocities are low in comparison, FEFLOW calculates higher flow velocities and in two cases also different flow directions.

11.2.2.2.3 Groundwater flow simulations considering heat transport

Heat transport was simulated on the same cross-section as described before. The parameters concerning the heat transport are listed in Tab. 11.8 Permeabilities and conductivities of the different units were set to the same values as in Tab. 11.5. Only the permeability of the fractures was increased to intensify the effect of the flow field on the temperature distribution (cf. Tab. 11.9). The effects of temperature on density and viscosity were neglected to facilitate the comparability of the simulation results.

Tab. 11.8 Hydrogeological parameters for the matrix and the fractures in the groundwater simulation concerning heat transport

Parameter	Matrix	Fracture
Porosity [-]	$4.65 \cdot 10^{-3}$	$5 \cdot 10^{-3}$
Tortuosity [-]	1.0	1.0
Molecular diffusivity [$m^2\ s^{-1}$]	$1.0 \cdot 10^{-9}$	$1.0 \cdot 10^{-9}$
Longitudinal dispersivity [m]	5.0	5.0

Parameter	Matrix	Fracture
Transversal dispersivity [m]	$5.0 \cdot 10^{-1}$	$5.0 \cdot 10^{-1}$
Heat capacity of the fluid [J kg^{-1} K^{-1}]	4.170	4.170
Heat capacity of the solid [J kg^{-1} K^{-1}]	2.199	2.199
Heat conductivity of the fluid [J s^{-1} m^{-1} K^{-1}]	$5.97 \cdot 10^{-1}$	$5.97 \cdot 10^{-1}$
Heat conductivity of the fluid [J s^{-1} m^{-1} K^{-1}]	2.66	2.66
Mass-density of the solid phase (rock)	$3.0 \cdot 10^{3}$	$3.0 \cdot 10^{3}$

Tab. 11.9 Hydrogeological parameters for the five fractures in the groundwater simulation concerning heat transport

Parameter	Fractures
Aperture [m]	$1.0 \cdot 10^{-2}$
Conductivity [m s^{-1}]	$1.0 \cdot 10^{-1}$
Permeability [m^2]	$1.0 \cdot 10^{-8}$

Boundary conditions used in the d^3f model are illustrated in Fig. 11.20. A temperature of 275.15 K was set at the top and of 305.15 K at the bottom surface of the model. To the left boundary, an in-/outflow boundary condition was assigned which means that the inflow temperature reflected the general temperature gradient of about 2.84 K per 100 m and the outflow temperature was allowed to vary. On the right boundary only outflow occurred which was allowed to take variable temperatures. The boundary conditions in FEFLOW were set accordingly. Only the in-/outflow boundary condition used on the left side was not available in FEFLOW so that a Dirichlet´s boundary condition with the same general temperature gradient was employed instead.

In both models a temperature distribution according to the general temperature gradient of about 2.84 K per 100 m was assigned as initial distribution.

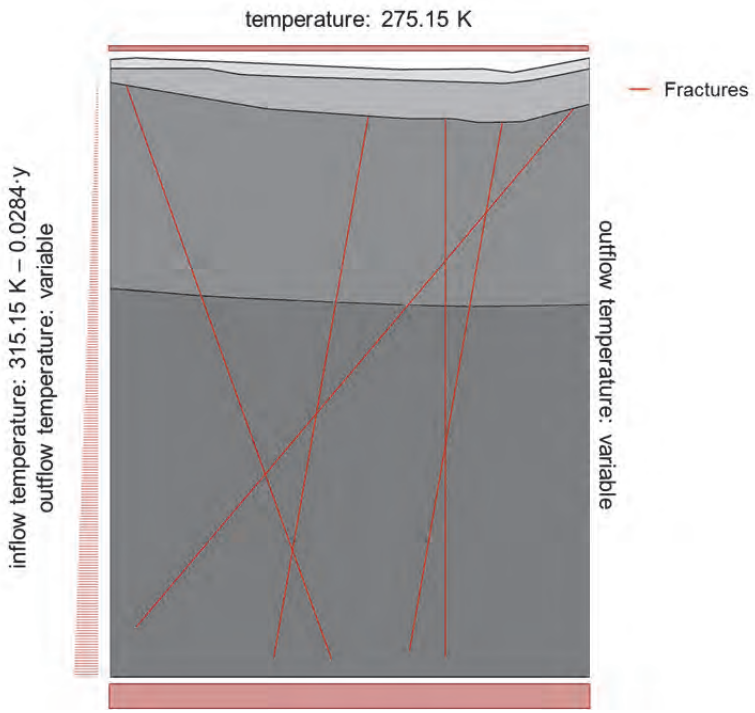


Fig. 11.20 Model geometry and temperature boundary conditions for the groundwater flow simulations with d^{3}f based on the schematic cross section

Both the d^{3}f and FEFLOW model were run for 187 000 days simulation time. Although it would have needed about ten times more time steps to reach the equilibrium state, the temporal dynamic already provided a good basis to compare the models. The velocity field in this scenario differed from the velocity field presented in Chapter 11.2.2.2.2 because of the higher permeability of the fractures. However, only the temperature distribution produced by the two models shall be compared in this chapter.

The overall temperature distribution follows the temperature gradient imposed by the boundary conditions of 275.15 K at the model surface and of 305.15 K at the bottom of the model (cf. Fig. 11.20). This distribution is modified because the fluid and thus heat in the fractures is transported faster than in the adjacent matrix. Due to heat conduction this effect influences also the surrounding matrix. Mainly in the upper two thirds of the model domain the temperature isolines in and near the fractures are shifted with the flow in the fractures. This effect is most prominent in the vicinity of the main flow path where the isolines are moved downwards along fracture F1 and upwards along fracture F5.

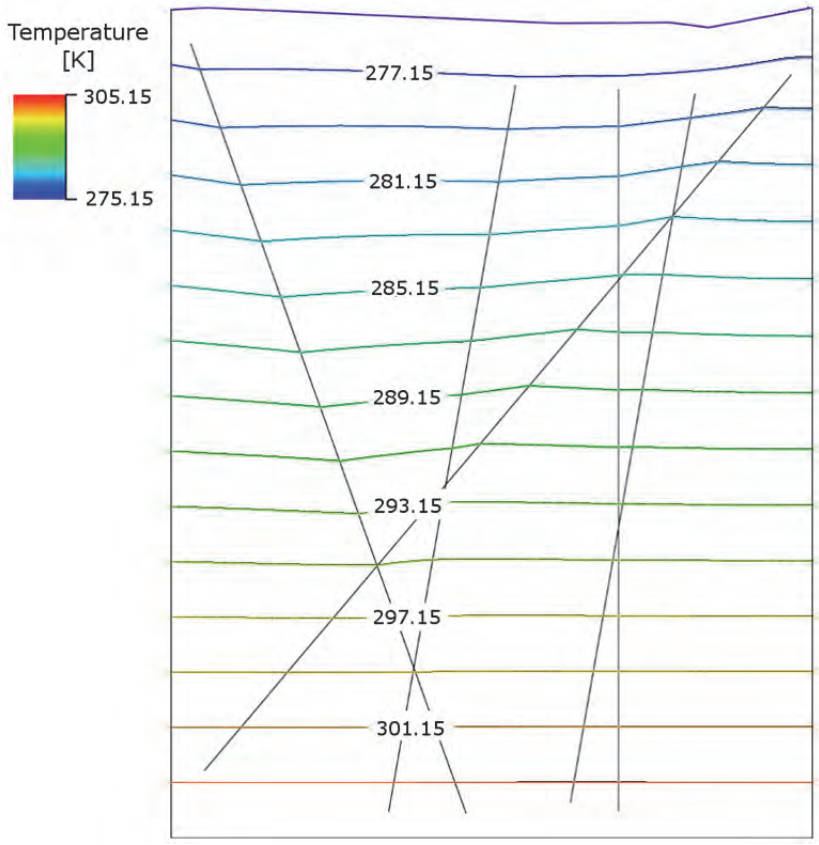


Fig. 11.21 Temperature isolines after 187 000 days calculated with d^3f (coloured isolines) and FEFLOW (black isolines)

The temperature isolines after 187 000 days computed with d^3f and FEFLOW show an excellent agreement (cp. Fig. 11.21). The black isolines obtained from FEFLOW calculations lie almost perfectly beneath the coloured d^3f-isolines.

11.2.2.2.4 Transport simulations

A transport simulation with constant density and viscosity was performed with d^3f and r^3t in order to test the capability of the codes to cope with transport in fractures. The results of these simulations were compared to FEFLOW calculations.

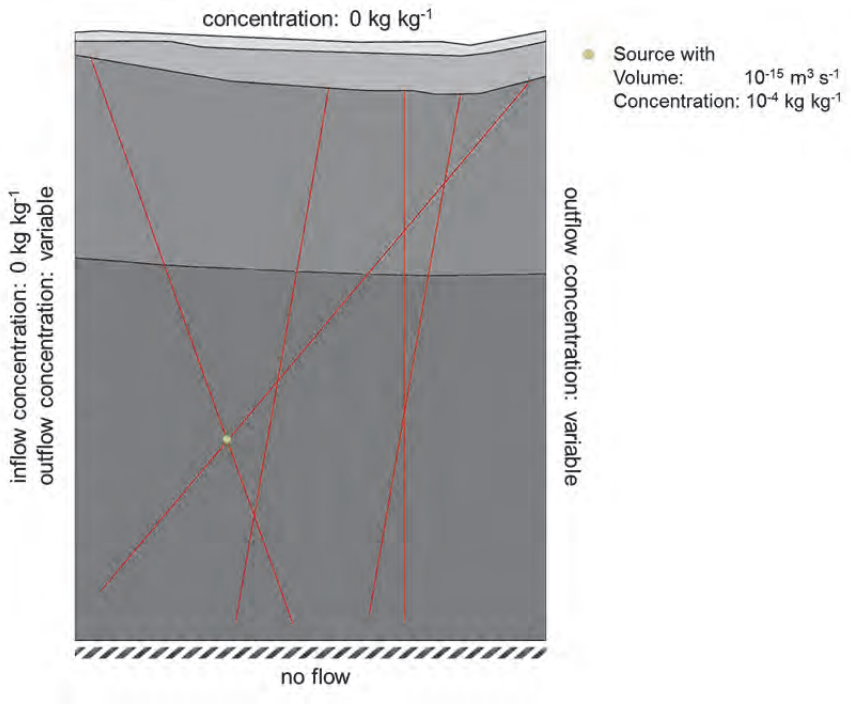


Fig. 11.22 Boundary conditions for the transport model in d^{3}f

The flow parameters listed in Tab. 11.5, Tab. 11.6 and the flow boundary conditions discussed in Chapter 11.2.2 were set for the d^{3}f and FEFLOW simulation. Additionally, a point source with an inflow rate of $1.0 \cdot 10^{-11}$ m^3 s^{-1} was introduced near the intersection of the fractures F1 and F5 in d^{3}f (cf. Fig. 11.22). The resulting steady-state velocity field served as basis for the r^{3}t-simulations.

Only element-wise sources were available in FEFLOW so that the inflow rate had to be related to the element size. Thus a flux of $3.7812 \cdot 10^{-6}$ m d^{-1} was added to an element with the size of 0.23 m^2 at the location of the point source in d^{3}f.

The transport parameters were the same as for the groundwater simulations considering heat transport (cf. Tab. 11.8). The transport boundary conditions for d^{3}f and r^{3}t are depicted in Fig. 11.22. In FEFLOW the boundary conditions were assigned accordingly. Only the in-/outflow boundary condition on the left side was again replaced by a Dirichlet´s boundary condition and the source was related to the element size. In all three programs the initial concentration was set to 0 mg l^{-1} in the whole model area.

The contaminant was assumed to be a tracer that perfectly follows the fluid motion and neither underlies radioactive decay nor shows any sorption.

11.2.3 Results

The introduction of the volume source causes a change in the flow field compared to the groundwater flow simulations in Chapter 11.2.2. This effect mainly affects the bottom layer left to the fractures F1 and F5 (cf. Fig. 11.23). It can be observed that the fluid flows radially outwards from the source. This pattern is broken where higher flow velocities e. g. in the fractures become dominant. The absolute values of flow velocity produced by FEFLOW are in the same order of magnitude (cf. Fig. 11.23). However, the differences between the bottom part of fracture F1 and F2 and fracture F3 and F4 remain the same as in Chapter 11.2.2 The flow directions in FEFLOW are comparable to d^{3}f but could not be shown due to the lack of an appropriate graphic option.

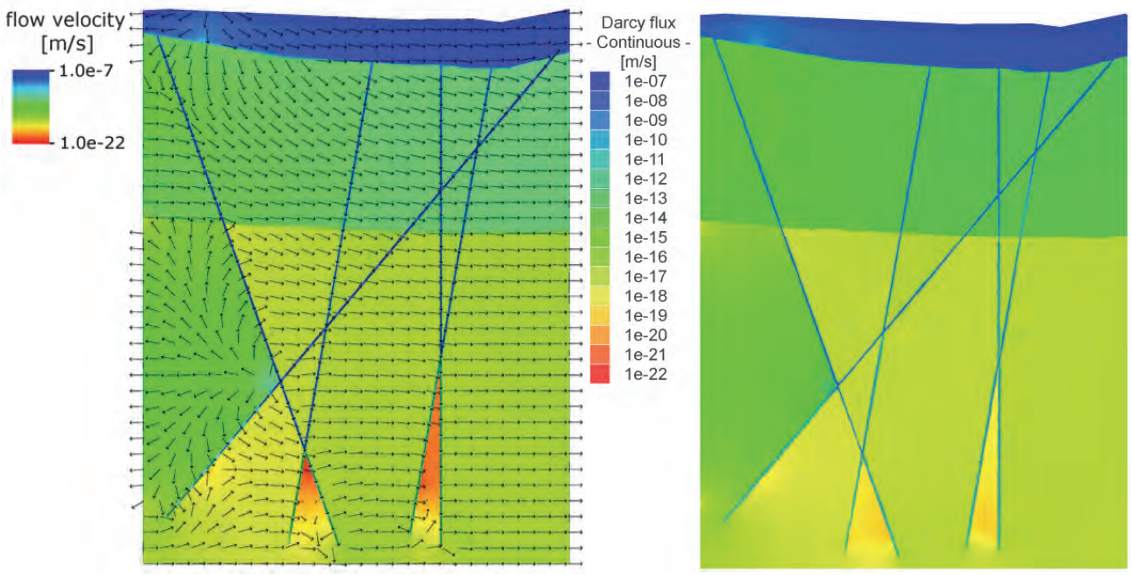


Fig. 11.23 Groundwater flow directions and velocities in d^{3}f (left) and FEFLOW (right) Velocity vectors are scaled to the same size.

The effect of the source on the flow velocities in the fracture differ between d^{3}f and FEFLOW. In d^{3}f the fluid mainly adds to the flow in fractures F5 and the triangle formed by the fractures F1, F2 and F5 (cf. Fig. 11.23). Here, the velocity ranges between $5.0 \cdot 10^{-10}$ and $2.0 \cdot 10^{-9}$ m s^{-1}. The flow velocity in the bottom dead ends of fractures F1 and F2 are very low and amount to 10^{-14} to 10^{-11} m s^{-1}.

In FEFLOW, the downward flow through fracture F1 is twice as high as in d^{3}f. In the dead ends of fractures F1 and F2 the flow velocity is one to three orders of magnitude higher than in d^{3}f and amounts to 10^{-12} to 10^{-10} m s^{-1}. Furthermore, in the bottom part of

fracture F2 the flow is directed upwards in d^3f and downwards in FEFLOW (not shown). The upward flow in fracture F2 between the intersection with the fractures F1 and F5 is reduced compared to d^3f. Thus, the relevance of the upward flow in fracture F2 is reduced and more fluid is transported downwards into the matrix in FEFLOW.

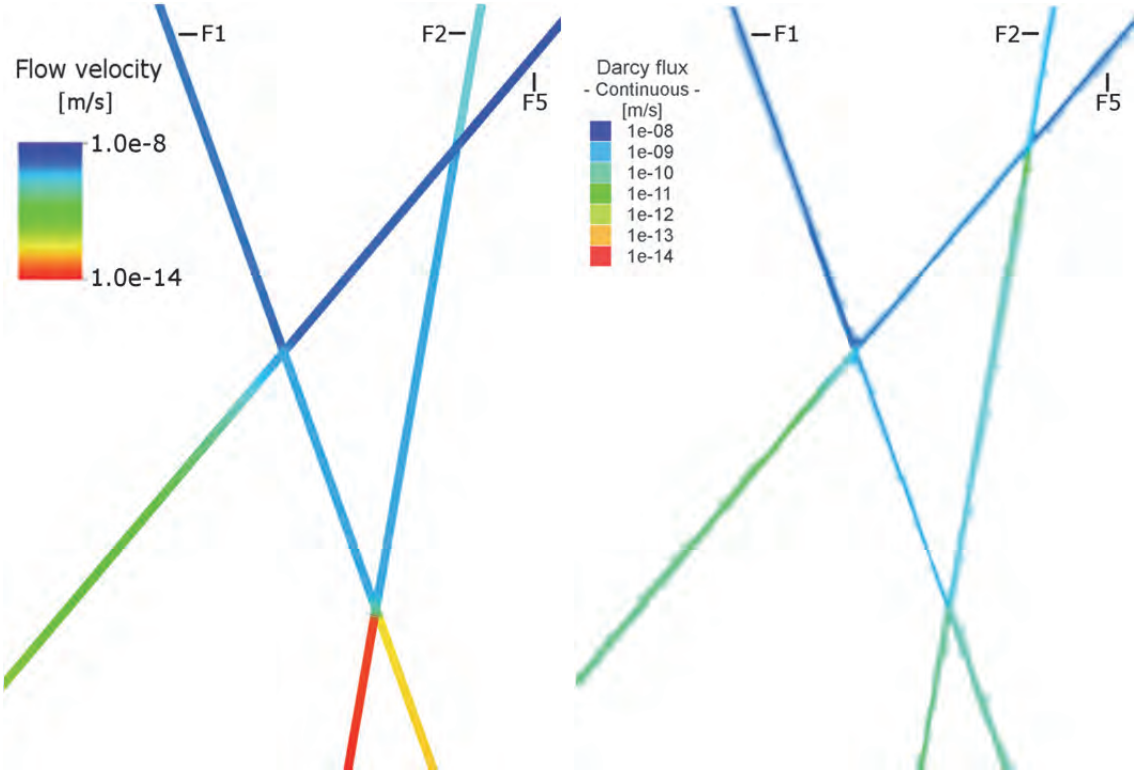


Fig. 11.24 Groundwater flow velocities in the fractures calculated with d^3f (left) and FEFLOW (right)
Velocity vectors are scaled to the same size.

For the sake of clarity, the results of d^3f, r^3t, and FEFLOW are compared pairwise. Thus, Fig. 11.25 shows only the results of d^3f and r^3t after $4.6 \cdot 10^6$ days. Then, d^3f and FEFLOW results are compared in Fig. 11.26.

The concentration isolines of 55, 45, 35, and 25 mg/l form after $4.6 \cdot 10^6$ days concentric circles around the source due to the radial flow direction discussed above (cf. Fig. 11.25). The 15 and 5 mg/l-isolines, however, are distorted because of the higher flow velocities in the fractures. The downward flow in fracture F1, for example, retards the spread upward of the tracer and accelerates its spread downward. Thus, the isolines are drawn in the flow direction. The degree of distortion depends on the flow velocity within the fracture. A low flow velocity causes only a slight change of the spread pat-

tern, e. g. in the bottom third of fracture F5, as opposed to the great change due to high flow velocities in fracture F1.

The isolines produced by d^3f and r^3t show a very good agreement. The agreement is almost perfect in the area left to the fractures (cf. Fig. 11.25). The 15 and 5 mg/l-isolines differ slightly because the transport in the fractures seems to be marginally faster in r^3t than in d^3f. However, the differences are small and both codes produce the same characteristics.

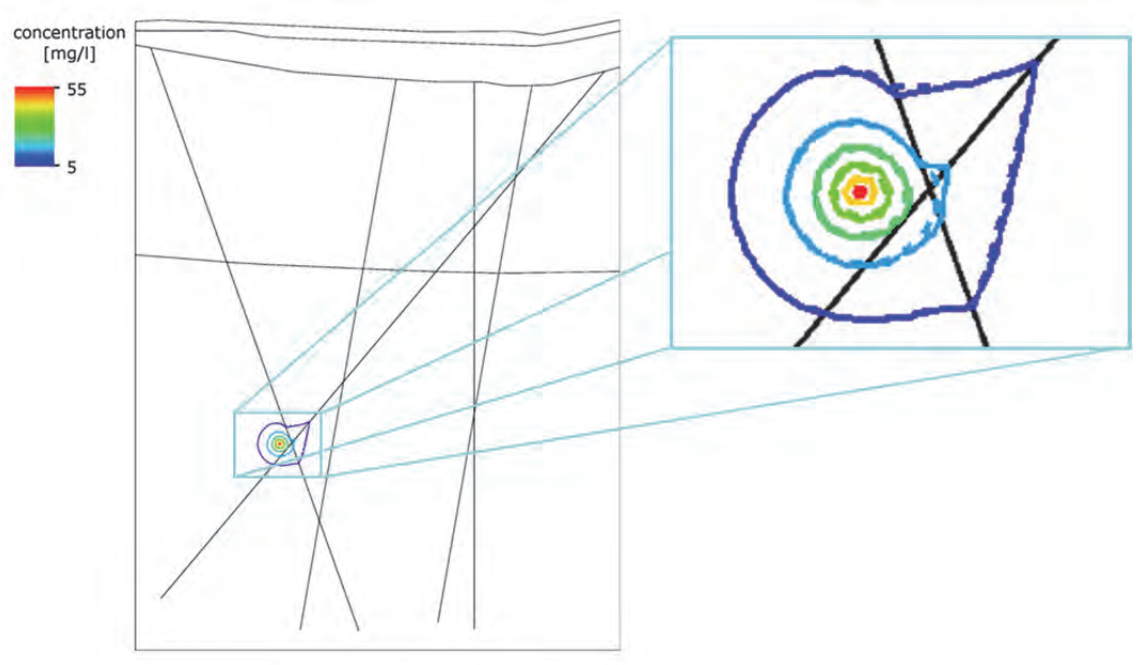


Fig. 11.25 Concentration isolines after $4.6 \cdot 10^6$ days calculated with d^3f (dashed line) and r^3t (solid line)

The distribution of the tracer after $3.5 \cdot 10^7$ days computed with d^3f and FEFLOW are shown in Fig. 11.26. The isolines in the region with low influence of the fractures on the flow field show a very good agreement. The differences between the codes near the fractures can be explained by the flow velocities within the fractures. Although in both models the flow velocities in the fractures near the source are increased, the influence of these fractures is different. In FEFLOW, the downward flow in fracture F1 is more important than in the d^3f model and less fluid is transported upward through fracture F2. Therefore, the tracer is transported further downward in FEFLOW within the simulation time. Therefore less tracer than in d^3f passes across fracture F1 and is transported upwards through fracture F5.

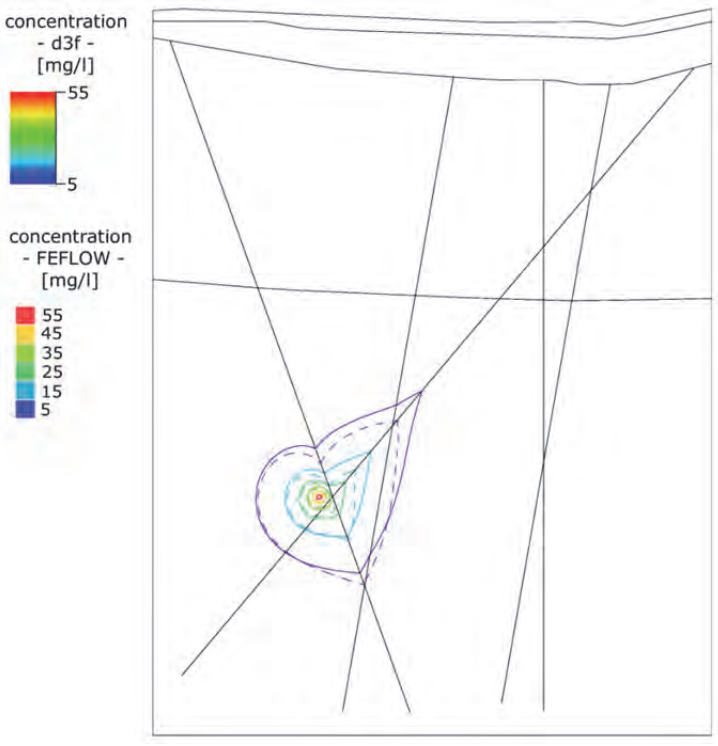


Fig. 11.26 Concentration isolines after $3.5 \cdot 10^7$ days calculated with d^3f (solid line) and FEFLOW (dashed line)

In summary, the d^3f, r^3t, and FEFLOW results of the tracer transport simulation show a good agreement. As expected, the d^3f and r^3t results are very similar. The marginal differences may have numerical reasons e. g. the length of the time steps differed slightly.

Differences between the d^3f and FEFLOW results are mainly caused by differences in the flow velocity in the fractures and thus in the distribution of the tracer. The very good agreement in the area without influence of the fractures demonstrates that the implementation of the point source in d^3f and the element-wise source in FEFLOW match quite well.

11.2.4 3d test case with a single fracture in an inhomogeneous matrix

11.2.4.1 Description of the test case

As a 3d test case an example provided by /ZIE 91/ was used that is based on a simple three-dimensional fracture-matrix system. The set-up described in the following was modelled in /ZIE 91/ with the code ROCKFLOW /KRO 91/ and now also with d^{3}f allowing a performance comparison of the two codes. The models concerned will be called the 'ROCKFLOW-model' and the 'd^{3}f-model' further on.

The model domain is cube-shaped with a side length of 100 m. It has a homogeneous permeability of 10^{-11} m² except for the lowermost 10 metres of the model to which a permeability of 10^{-10} m² was assigned. The model domain contains an almost diagonal planar fracture as shown in Fig. 11.27 with an aperture of 1 m and a permeability of 10^{-9} m, two orders of magnitude higher than in of the surrounding matrix. Water enters the model domain through a 10 m high area at the upper edge of the cube above the location of the fracture. In this area the hydraulic head amounts to 0 m. It leaves the domain again through a similar vertical area of 10 m height at the bottom of the model domain where a head of -100 m is prescribed (see Fig. 11.27). The pressure assigned to the inflow and outflow boundaries has to account for hydrostatic pressure, additionally. This yields values for the pressure at the inflow boundary from 0 MPa to 9.8×10^4 MPa and at the outflow boundary from -9.8×10^4 MPa to 0 MPa, respectively.

At the upper back corner of the cube within the inflow boundary over a tracer also enters the model domain. Tracer inflow into the ROCKFLOW-model is realised as a point source while in the d³f-model tracer inflow is prescribed over an area of 2 m x 2 m. In both cases a relative concentration of 1 is assigned to the inflow location. All other boundaries are closed for flow and transport.

Fig. 11.27 shows the set-up of the problem, and Fig. 11.28 the computational grid. Flow and transport were simulated with d^{3}f using a time step of 1000 s and a multigrid level of 4, resulting in 1.4 millions nodes. For investigating grid convergence the d³f-model was once also run with a level of 3 with 181 000 nodes. The ROCKFLOW-model comprised only 3751 nodes.

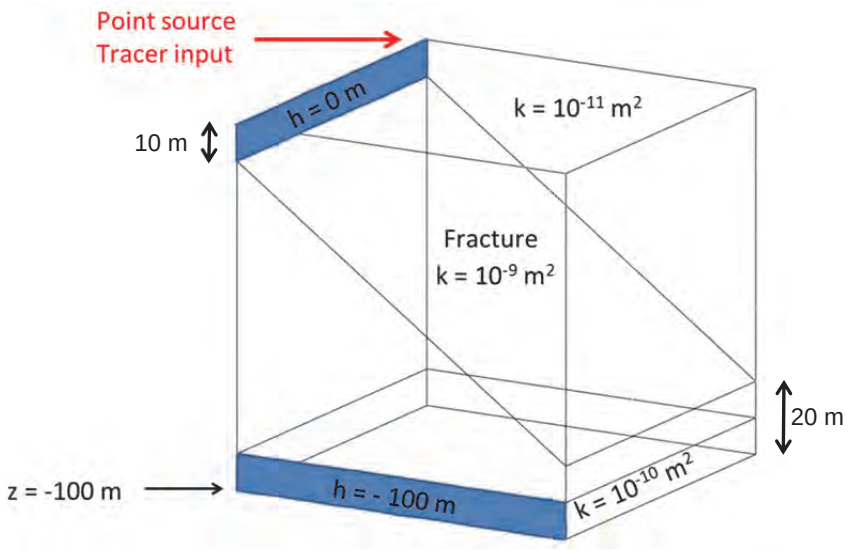


Fig. 11.27 Model and boundary conditions; blue: inflow/outflow boundaries

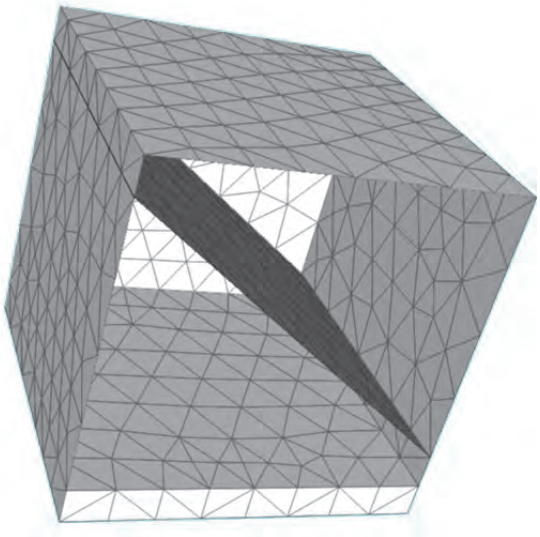


Fig. 11.28 Computational grid, created with ProMesh (see section 9.2); front side and inner elements removed for better view

11.2.4.2 Results

In the following, three models are discussed and compared: the ROCKFLOW-model, the d^{3}f-model and, for reference, a modified d^{3}f-model that contains no fracture. Compared are vertical cross sections through the models showing the velocity and concentration fields. For simulating the system without the fracture the permeability of the fracture was set to the value for the surrounding matrix, i. e., 10^{-11} m^2.

11.2.4.2.1 Velocity

Fig. 11.29 and the left plot in Fig. 11.30 show the velocity field in a vertical section through the model domain from the ROCKFLOW- and the d³f-model, respectively. As may be seen from these figures, the velocities are similar for both models. The right plot in Fig. 11.30 shows the velocity field in the modified d^3f-model without the fracture. A comparison of the d^3f-models in Fig. 11.30 shows the influence of the fracture on the flow field. In the left plot it can clearly be seen that the fracture acting as a hydraulic shortcut deflects the streamlines considerably.

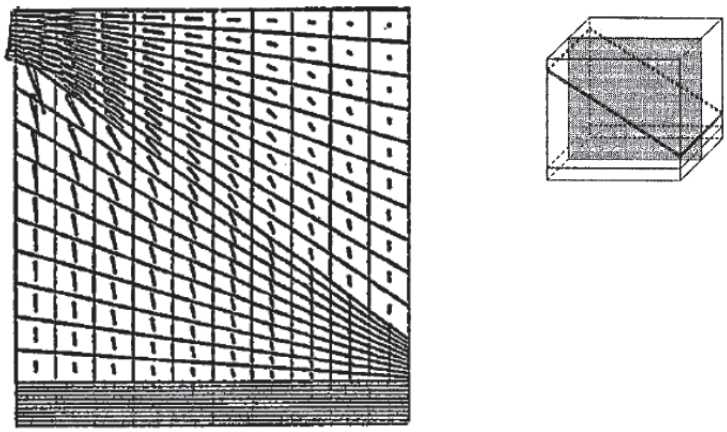


Fig. 11.29 Velocity field in a vertical cross section from the ROCKFLOW-model, source: /ZIE 91/

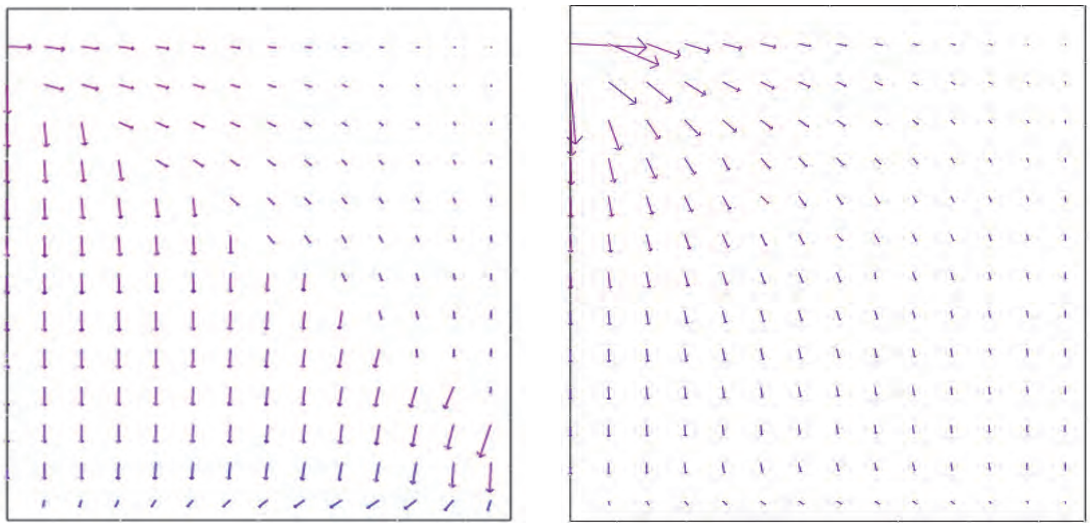


Fig. 11.30 Velocity field in a vertical cross section from the d^3f-model (location as indicated in Fig. 11.29); left: with fracture, right: without fracture

11.2.4.2.2 Concentration

The results from the ROCKFLOW-model in terms of solute concentrations at steady-state are given in /ZIE 91/ as a sequence of eleven vertical concentration profiles, c.f. Fig. 11.31. Related results from d^3f are depicted in Fig. 11.32 in four vertical cross sections whose location and thus relation to the results from the ROCKFLOW-model is indicated in Fig. 11.31. The concentration fields agree rather well. Fig. 11.32a also indicates a discontinuity of the concentration across the fracture. Here, the different handling of fractures in ROCKFLOW and d^3f becomes obvious. In a ROCKFLOW-model the same node that represents the fracture represents also the matrix within the patch of elements that share this node. Under these circumstances the concentration profile across a fracture is continuous in ROCKFLOW. In d^3f, however, such a node is resolved into at least three nodes allowing jumps of the concentration across a facture. It is thus that a complete match of the results cannot be expected.

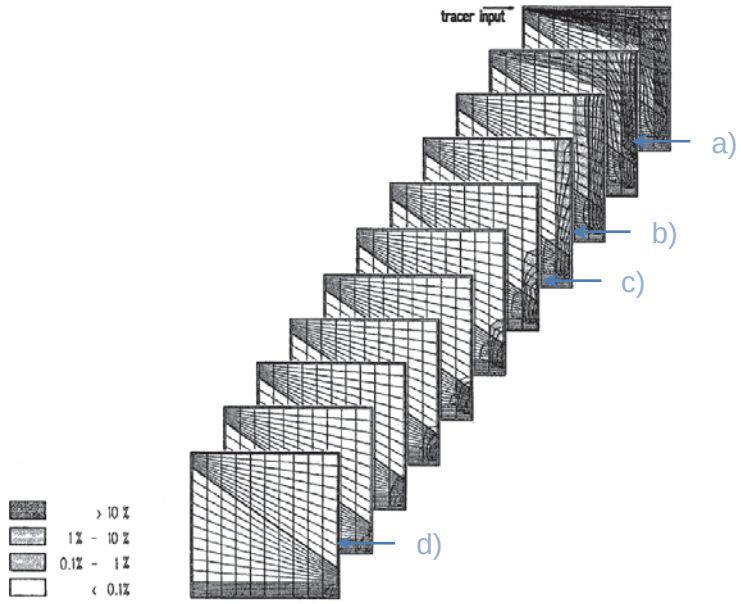


Fig. 11.31 Vertical concentration profiles from the ROCKFLOW-model at steady state; from /ZIE 91/; specified cross sections relate to Fig. 11.32

Exemplarily, a rough check concerning grid convergence is done by reducing spatial resolution using multigrid level 3 instead of 4. The results are given in terms of concentrations in two vertical cross sections as depicted in Fig. 11.29. The plots with multigrid level 3 naturally show more smearing in the concentration distribution where large concentration gradients exist. But they agree qualitatively well with the related results obtained with multigrid level 4.

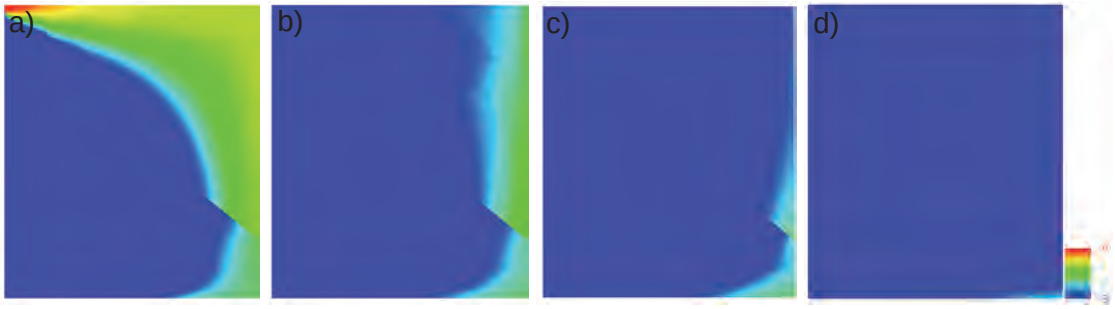


Fig. 11.32 Vertical concentration profiles (logarithmic scale) from the d³f-model at steady state condition at the locations indicated in Fig. 11.31

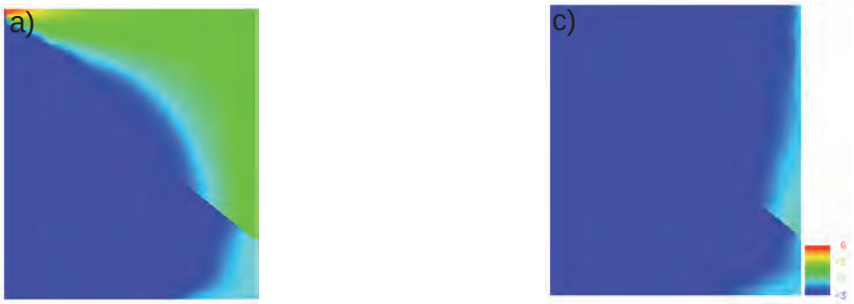


Fig. 11.33 Vertical concentration profiles (logarithmic scale) from the d³f-model at steady state condition at the locations indicated in Fig. 11.31 for a refinement level of 3

11.2.4.3 Summary

The d³f-model was able to reproduce velocity and concentration fields of the three-dimensional fracture-matrix system produced with the ROCKFLOW-model. The influence of the fracture on flow and transport as also discussed based on results from the d³f-model. It was found that the fracture introduces a discontinuity for flow and transport in the matrix. Simulation of solute transport with refinement level 3 generates smearing of the concentration in comparison to results with refinement level 4 but qualitatively no real difference. Thus, grid convergence appears to has been almost reached.

11.2.5 A realistic fracture flow model - SKB's Task8b

11.2.5.1 Introduction

In the past the Swedish SKB has established several Task Forces, two of which are of interest here: the Task Force on groundwater flow and transport of solutes (TF GWFTS) and the Task Force on engineered barrier systems (TF EBS). In collaboration representatives of both Task Forces have come up with the definition of the so-called Task 8, a compilation of several subtasks – called 8a, 8b, etc. – with a view to the hydraulic interaction between the bentonite clay buffer and the granitic host rock.

Presently, the Task is intended to be active from 2010 until 2013. It runs thereby parallel to the related BRIE-project (Bentonite Rock Interaction Experiment) at the Äspö Hard Rock Laboratory (HRL). The BRIE-project is concerned with an in-situ test where one or two boreholes will be drilled from a tunnel floor cutting at least through one large and one minor fracture and will be filled with bentonite. The procedure for finding and characterizing a suitable site for the test is also part of Task 8. The objective of this experiment is to measure a) water uptake of the bentonite via different water flow paths – i. e. fractures and rock matrix – and b) the reaction of the flow system in the rock. Task 8 includes predictive as well as interpretive modelling parallel to the on-going BRIE-experiment.

Task 8b addresses scoping calculations for a possible borehole in a simplified realisation of the actual site at the Äspö HRL. While this Task was originally intended to make the modellers of the bentonite buffer familiar with the influence of a real groundwater flow system it was acknowledged that it could also be used as a platform for testing the fracture flow capabilities of the codes involved. A description of the groundwater flow system underlying Task 8b as well as of the numerical model and its results will be given in the following.

11.2.5.2 Geometry of the groundwater flow system

11.2.5.2.1 Coordinates

The coordinate system used in Task 8 is the Swedish RT90 system. At Äspö this system leads to large numbers if the x- and y-coordinates are given in meters. It is therefore recommended to cut off the leading 4 digits of the x- and y-coordinates.

All coordinates provided in the task description /BOC 11/ and in the supplementing data-files are given as integers meaning that they are resolved only in the meter-scale.

11.2.5.2.2 Model Domain

The suggested model domain is cube-shaped with a side length of 40 m. It is bounded by a set of eight corners. The coordinates given in the task description lead to slightly off-orthogonal angles for the top and bottom quadrilaterals. In order to provide a better approximation to a cube-shaped boundary corrected values with a precision of millimetres instead of meters were calculated.

11.2.5.2.3 Drifts and boreholes

In the model domain two drifts were excavated: the TASD- and the TASO-tunnel. The drifts have a plane floor and plane walls but a domed roof. The TASD-tunnel begins outside the model domain but ends within. The TASO-tunnel branches off from the TASD-tunnel and also ends within the domain. The cross-sections of both drifts are reduced towards the last meters to the end of the respective drift. Contrary to the task description these changes are not considered here.

There are two boreholes from the Temperature Buffer Test (TBT) and the Canister Retrieval Test (CRT) – labelled here: “deposition borehole 2” and “deposition borehole 3” – at the bottom of the TASD-tunnel and one fictional borehole at bottom of the TASO-tunnel, called “deposition borehole 1”. Their size is not directly given. However, from the data it can be assumed that they have a diameter of 1.60 m and a depth of 9 m. A “user-defined” borehole with a diameter of 30 cm and a depth of 3 m is also considered as requested by the task description.

The geometry data is provided in stl-files as ensembles of triangles representing the surface of the drifts. In case of the TASD-tunnel the description covers also the part outside the model domain. The intersection of TASD- and TASO-tunnel is not resolved. Instead, the describing triangles for the TASO-tunnel reach into the TASD-tunnel.

As with the definition of the model boundaries the data provided electronically resolve the coordinates only with an accuracy of 1 m. This leads to strange structures as shown in Fig. 11.34 a). The given structures were therefore replaced by geometrical descriptions that encompass only a minimum of bounding faces as in Fig. 11.34 b). The curvature of the roof is simplified to a polygon with five nodes. Boreholes are represented by hexagonal prisms. Additionally, the part of the TASD-tunnel that lies outside the model boundaries is cut off.

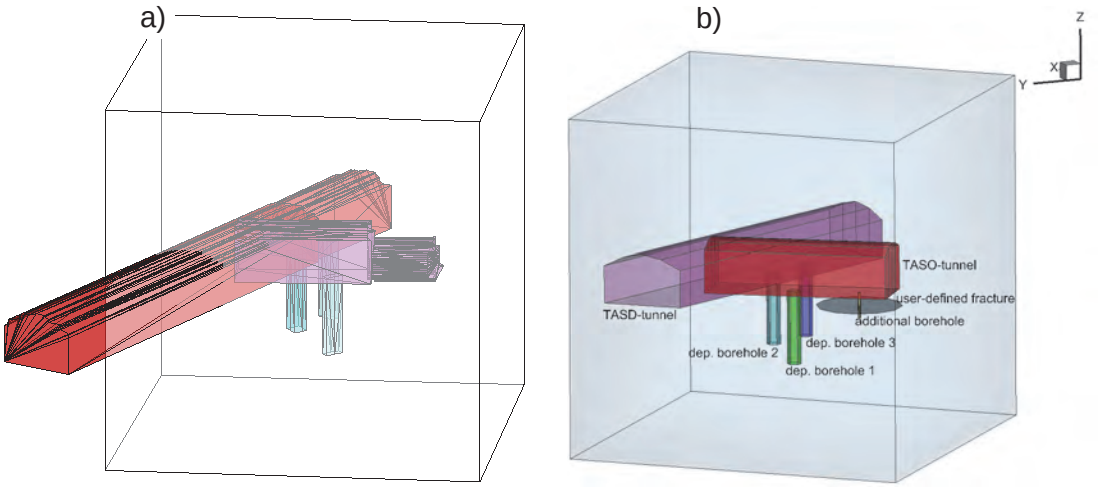


Fig. 11.34 Geometry of the openings; a) original data, b) modified data

The remaining geometrical model contains some inaccuracies:

- The intersection of TASD- and TASO-tunnel is still not resolved.
- There is a little slope in the TASD-tunnel leading to an initial slanting of the floor of the TASO-tunnel that is also not considered in this description.
- The top of the boreholes is only approximately consistent with the floor of the drifts.

11.2.5.2.4 Fractures

In the model domain seven large-scale fractures are located. They are larger than the 40 m model domain. The edges of the fractures are therefore defined by the interception of fractures and the model boundaries.

Data from the task description shows that the fractures are almost but not quite plane features (see Fig. 11.35 a). At a closer look they show actually a polyhedral structure. In the framework of this description, however, the fractures are treated as planes as shown in Fig. 11.35 b). Due to inclination of the fractures and position within the model domain some fractures are represented by pentagons.

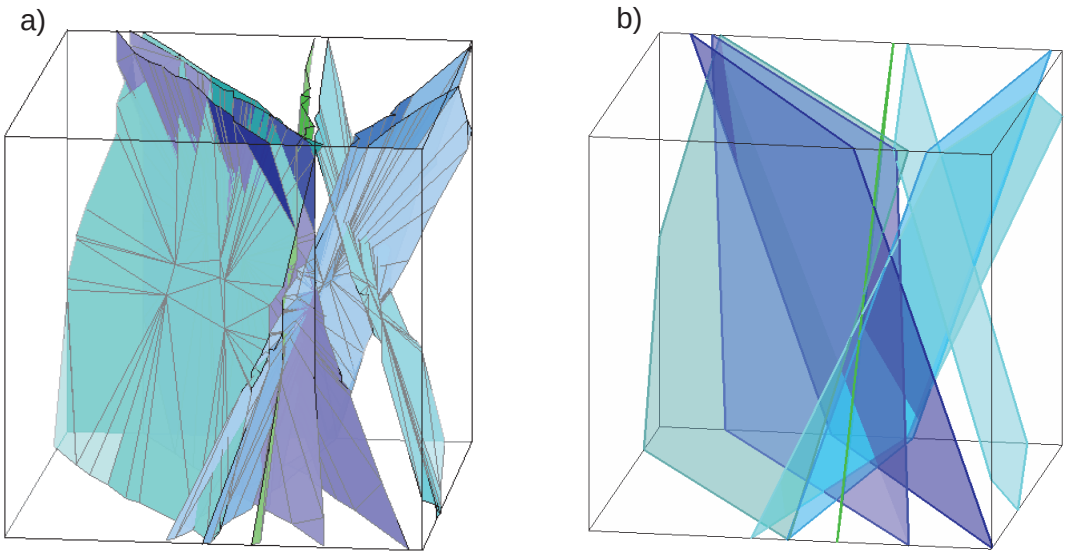


Fig. 11.35 Fracture geometry; a) original data, b) modified data

Additionally, a hypothetical single rock fracture is assigned to the model as a circular (or equivalent) feature of a diameter of 10 metres with its centre along the centre axis of the additional borehole (see section 11.2.5.2.3). As a reference case the single rock fracture lies horizontal and is located 1.5 metres below the drift floor.

Note that some interceptions of different fractures might lead to geometries that provoke difficulties for grid generation as well as for the numerical simulation.

11.2.5.3 Hydraulic properties

Three different hydraulic features have to be characterized: the rock matrix, the large-scale fractures and the user-defined fracture. Drifts and boreholes are assumed to be open to the atmosphere. While the data for the rock is given in terms of hydraulic conductivity and porosity only transmissivity is measured in case of the fractures. An aperture is assigned to the fractures in the task description to enable modelling in discrete fracture networks (DFN). If this transport aperture is also applicable for flow simulations is not clear. Nevertheless, it is used here to derive the fracture permeability. All the data is compiled in Tab. 11.10. Derived values are given in italics.

Tab. 11.10 Hydraulic properties of the hydraulic features

Property	Rock matrix	Large-scale fractures	User-defined fracture
Hydraulic conductivity [m s^{-1}]	$1 \cdot 10^{-12}$		
permeability [m^2]	$1 \cdot 10^{-19}$	$5 \cdot 10^{-7}$	$5 \cdot 10^{-8}$
porosity [-]	$1 \cdot 10^{-5}$	$1 \cdot 10^{-3}$	$1 \cdot 10^{-3}$
transmissivity [m^2 s^{-1}]		$5 \cdot 10^{-8}$	$5 \cdot 10^{-10}$
transport aperture [m]		$1 \cdot 10^{-5}$	$1 \cdot 10^{-6}$

11.2.5.4 Hydraulic boundary conditions

As mentioned above, atmospheric pressure is assigned to the surface of the drifts and boreholes. For the conditions on the outer surface of the model an excel-file with the results of a large-scale flow simulation for the HRL Äspö was provided with the task description. The relevant information about dynamic pressure, flow and salinity for the model could be extracted from this file. Note that dynamic pressure is defined here as the difference between absolute pressure and hydrostatic pressure. The data was given as pointwise information from the nodes of an irregular grid. The position of the nodes in space is depicted in Fig. 11.36.

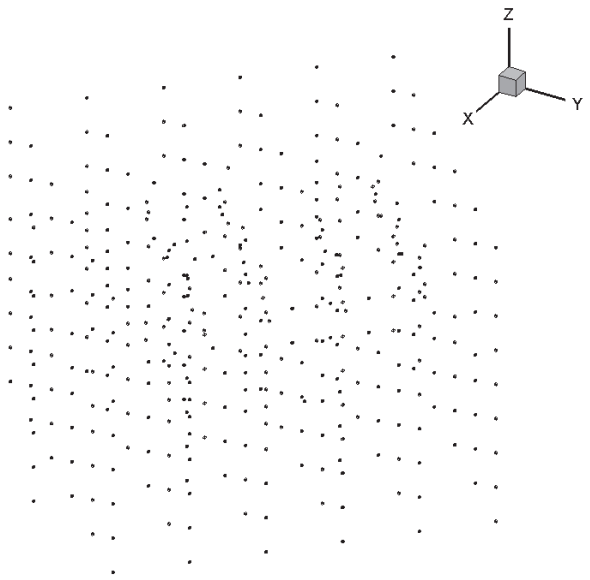


Fig. 11.36 Location of the given nodes in space

Since no information about the connection of the nodes with their neighbouring nodes was given, the data could not simply be interpolated to derive values on the model boundary planes. Instead the following strategy – illustrated in Fig. 11.37 – was applied:

for each of the six boundary planes

- define the mathematical formula for the plane
- find all nodes within a distance of Δl to the plane
- for each of those nodes
  - find other nodes
    - within a distance of 2 Δl
    - on the opposite side of the plane
  - calculate coordinates for the interception of the connecting line with the plane
  - interpolate the data for the interception

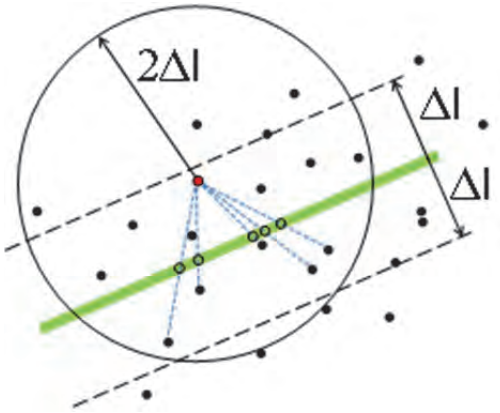


Fig. 11.37 Extraction strategy for the boundary conditions

The extraction process using a value of Δl = 20 m yielded enough data points to construct 2d-data fields in the six bounding planes. They are shown exemplarily for the dynamic pressure in Fig. 11.38. These planes had to be reduced to the boundary faces as indicated in Fig. 11.39. The whole model surface is shown in Fig. 11.40 including the interception of fractures and boundary faces.

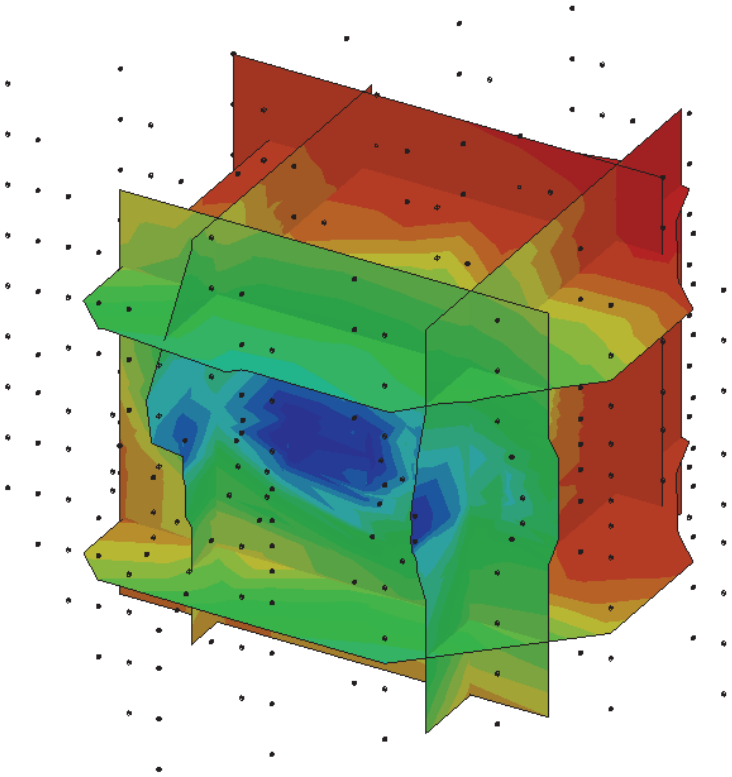


Fig. 11.38 Six planes showing dynamic pressure

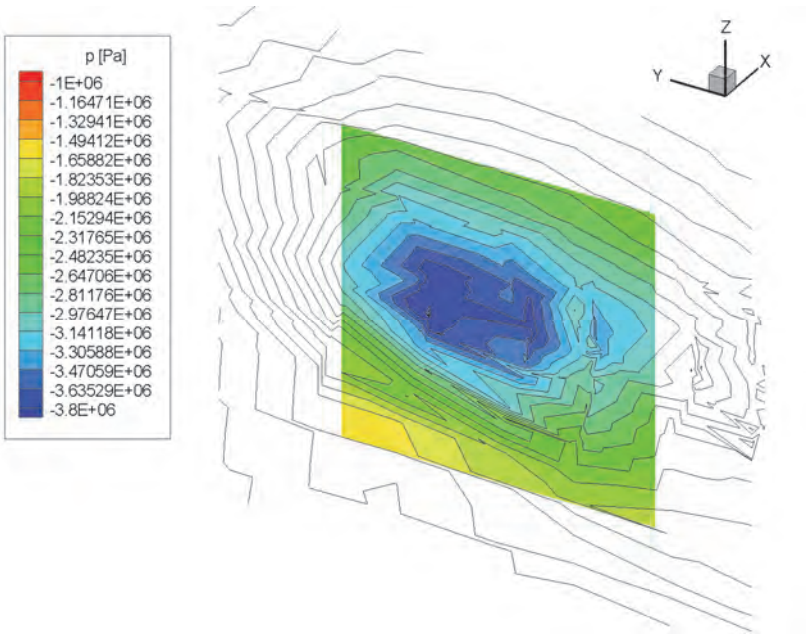


Fig. 11.39 Clipping of a boundary plane

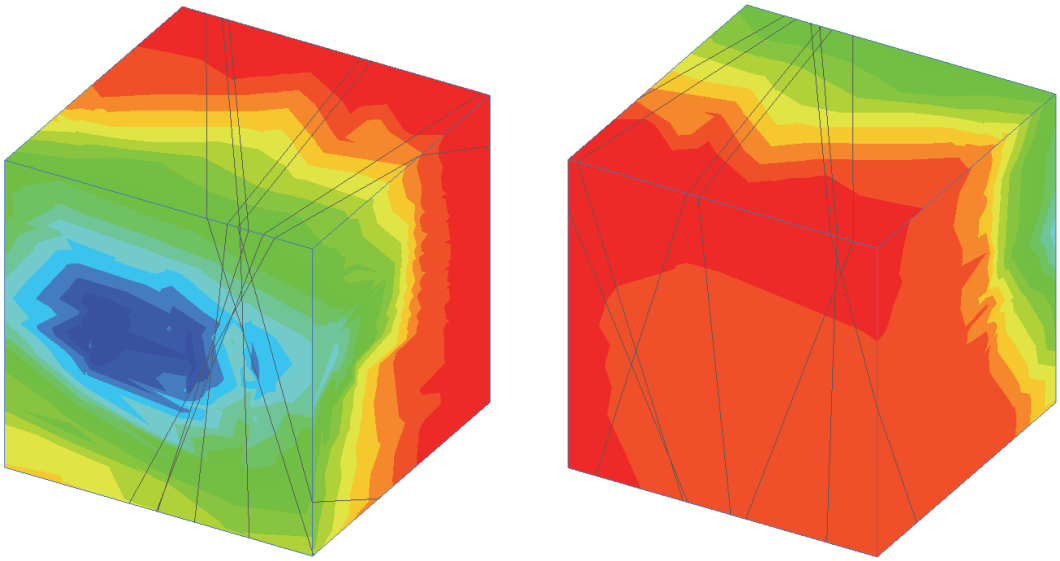


Fig. 11.40 Dynamic pressure on the model boundaries

Fig. 11.40 shows a rather erratic pressure distribution on a small scale that represents the original simulation results on that scale probably quite poorly. Flow simulations based on these boundary conditions can be expected to show numerical difficulties and unphysical results.

On a larger scale, however, it shows also a certain pattern. The pressure distribution on the model boundary was therefore approximated by an analytical formulation. The

quality of the approximation is depicted in Fig. 11.41, where the extracted data and the results from the analytical function can be compared.

$$p_{dyn} = \frac{d}{\left[\sqrt{(\xi + a)^2 + (\eta + b)^2 + (\zeta + c)^2}\right]^e} \tag{11.49}$$

p_{dyn}	-	dynamic pressure [Pa]
ξ, η, ζ	-	local coordinates [m] (q. v. (11.50))
a,b,c,d,e	-	constants (q. v. Tab. 11.11)

$$\begin{aligned} \xi &= \frac{1}{\sqrt{2}} x' + \frac{1}{\sqrt{2}} y' \\ \eta &= -\frac{1}{\sqrt{2}} x' + \frac{1}{\sqrt{2}} y' \\ \zeta &= z \end{aligned} \tag{11.50}$$

x' , y'	-	auxiliary coordinates [m] (q. v. (11.51))
z	-	vertical coordinate [m]

$$\begin{aligned} x' &= x - 1551000 \\ y' &= y - 6367000 \end{aligned} \tag{11.51}$$

x,y	-	horizontal RT90-coordinates [m]

Tab. 11.11 Constants for equation (11.49)

a	b	c	d	e
165	-926	413	-14,000,000	0.7

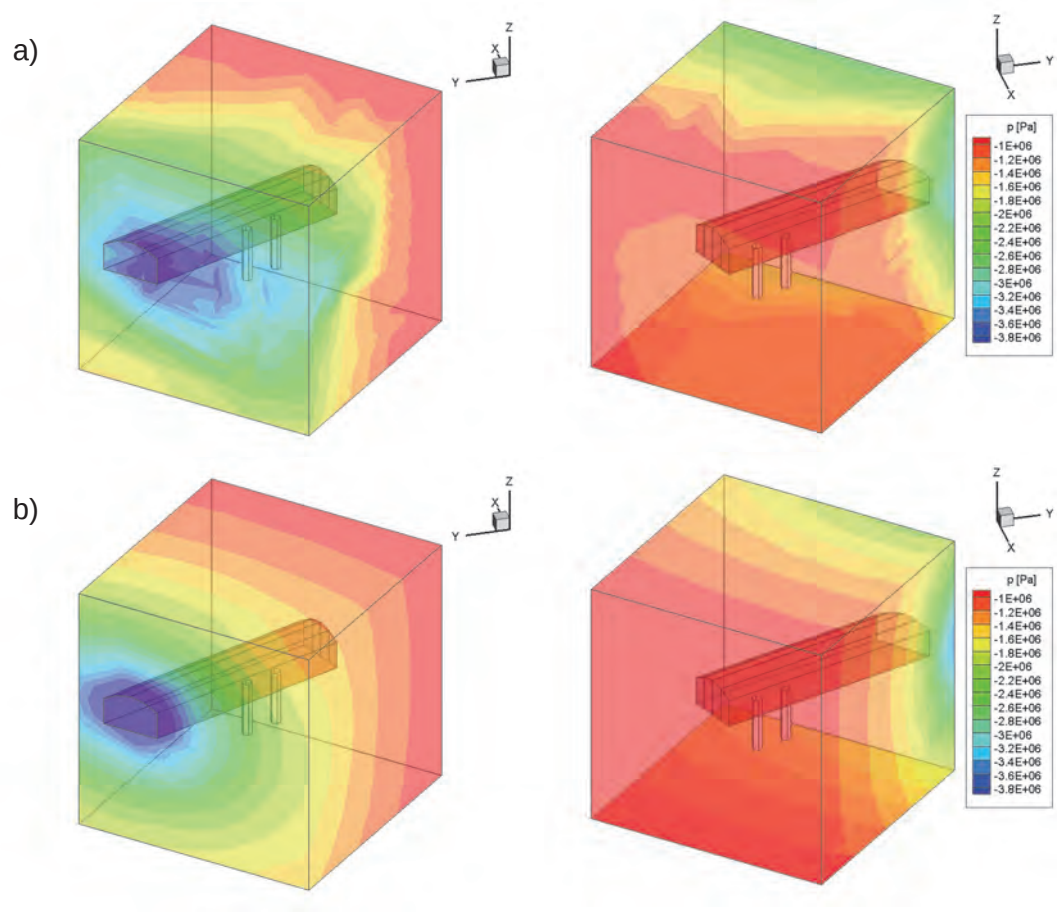


Fig. 11.41 Comparison of extracted data (a)) and results of the analytical function (b))

Note that the pressure distribution of the original simulation does apparently account for the influence of the open TASD-tunnel and the fractures only rather loosely. The boundary conditions with respect to pressure will therefore have to be modified to avoid unrealistic flow rates.

11.2.5.5 Influence of salinity

There is a noticeable trend in the salinity data provided by the TF GWFTS as depicted in Fig. 11.42. However, the maximum difference amounts to less than 0.1 %. In the light of the overall model uncertainties the effect from the varying density can therefore safely be neglected.

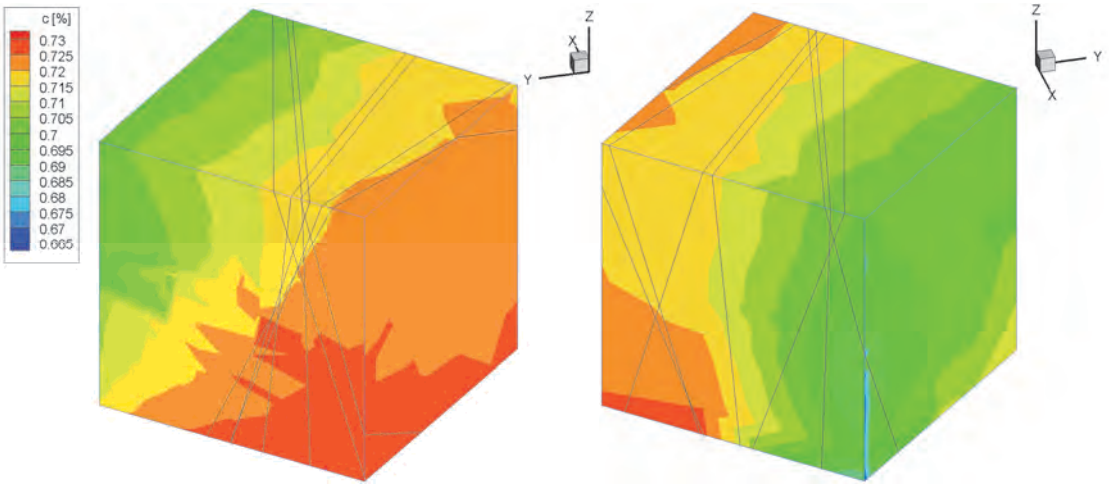


Fig. 11.42 Salt concentration on the model boundaries

11.2.5.6 Numerical grid

The first attempt on the grid for the coarse grid solver was performed with the ProMesh3-Tool (see section 9.2). It consisted of 12 634 nodes and 62 175 elements. Fig. 11.43 a) shows the model surface where the TASD-tunnel cuts through the model surface. The intersections of the fractures with the mode boundary are visible as straight lines on the surface. A vertical cross-section through the model is shown in Fig. 11.43 b). Both drifts can be identified by the characteristic cross-sections as well as the coloured fractures. Fig. 11.43 c) and Fig. 11.43 d) represent horizontal cross-sections in the plane of the drifts, one including the 3D-elements for the rock matrix and one showing only fractures and surfaces.

The reason for this comparatively fine discretization lay in the fracture geometry which included several subparallel fractures intersecting in close vicinity and thus required a rather fine grid resolution. This led to problems with the multigrid solver because the coarse grid solver did not work economically anymore. At a later stage a coarser grid was developed as shown in Fig. 11.44 that consisted only of about 25 000 elements which improved the computational performance considerably. The finest grid used during the calculation contained 140296 nodes and 588776 elements.

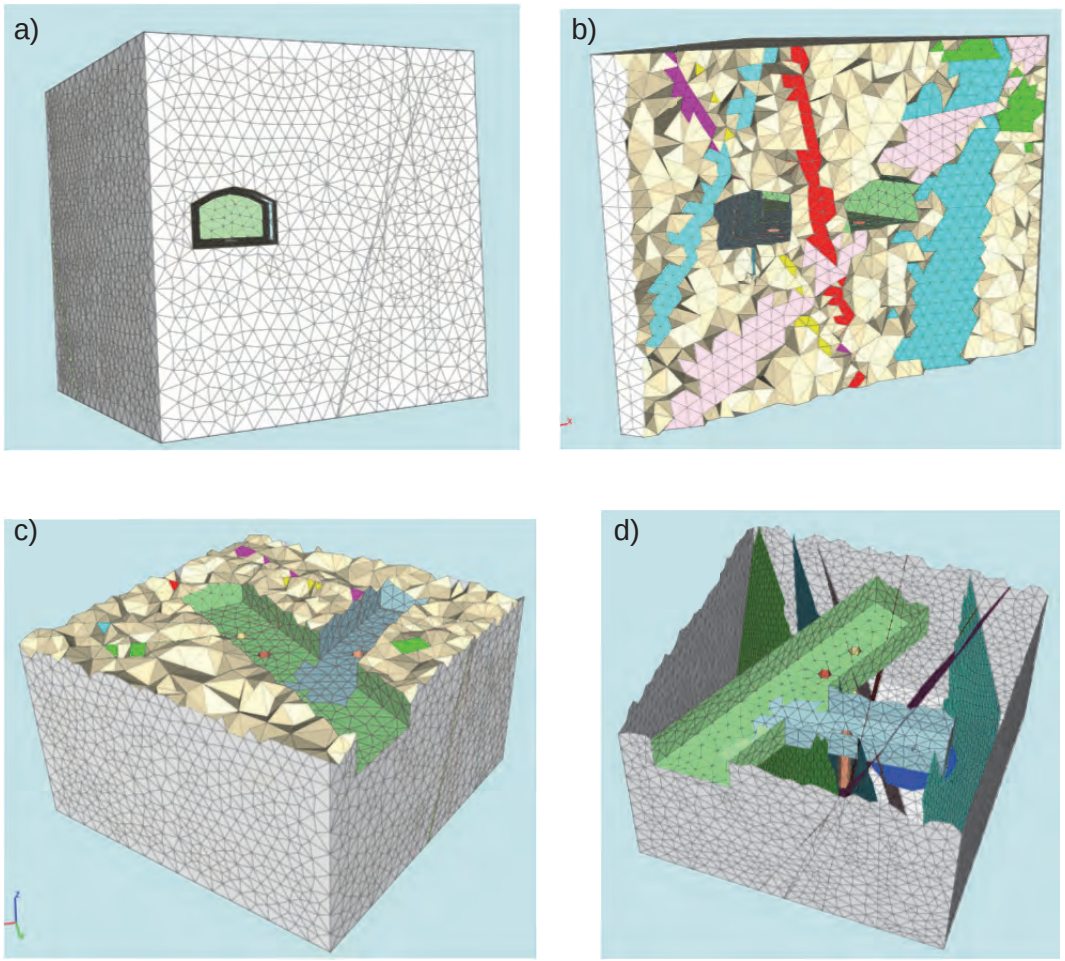


Fig. 11.43 First attempt on the coarse grid; a) view of the model surface, b) vertical cross-section, c) horizontal cross-sections

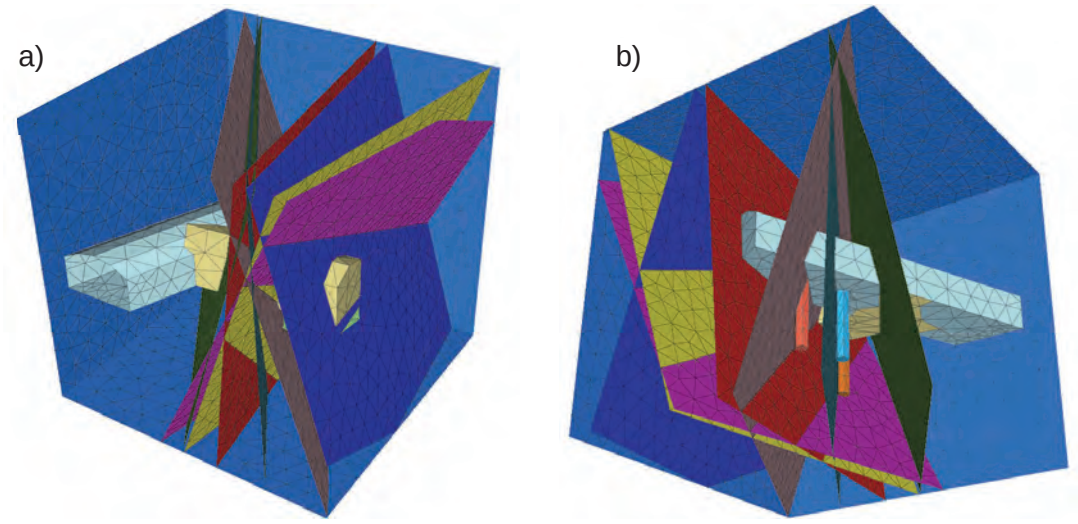


Fig. 11.44 Ultimately used coarse grid for the model; a) view from above, b) view from below

11.2.5.7 Results

11.2.5.7.1 Dynamic pressure

Results are given in terms of dynamic pressure distributions, flow fields, and water outflow at the model boundary. Fig. 11.45 shows isoplanes in a vertical cross-section through the TASD-tunnel representing the dynamic pressures of -3.5, -3.0, -2.5, -2.0, and -1.5 MPa. The pressure decreases from the cube surface in the direction of the openings showing the highest gradient at the end of the TASD-tunnel. The contour plane of lowest pressure (blue) follows loosely the surface of the openings. This is evident at the deposition boreholes 2 and 3.

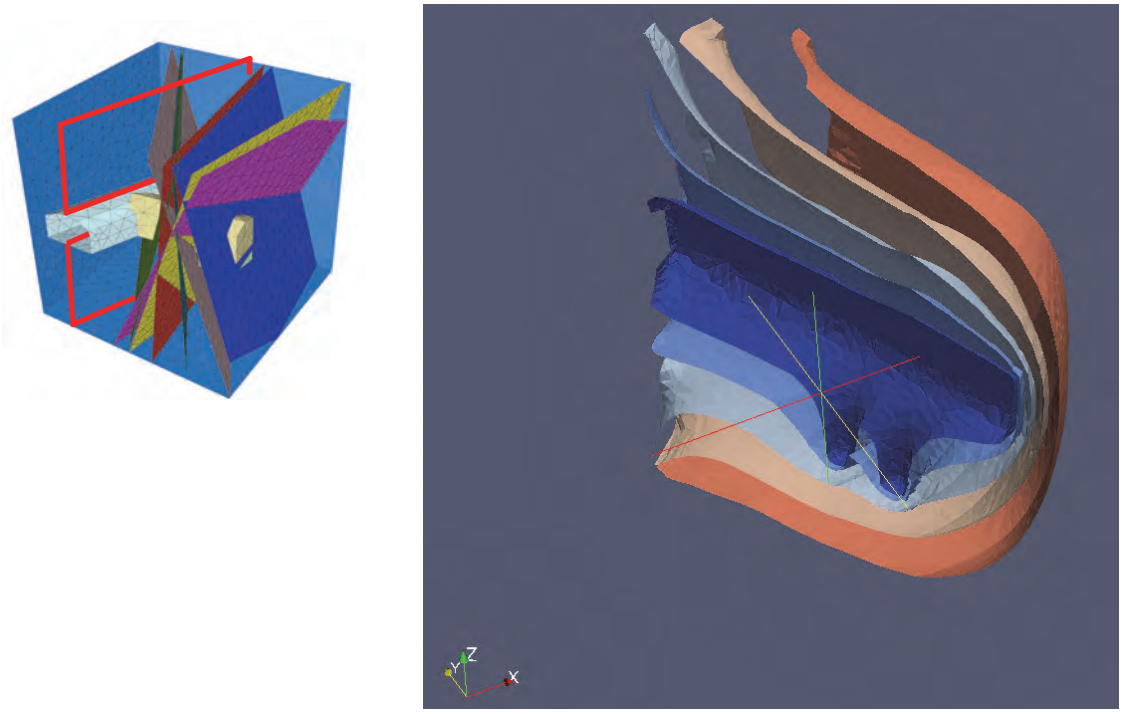


Fig. 11.45 Isoplanes of the dynamic pressure at the TASD-tunnel

The part of the model shown in Fig. 11.45 is only little disturbed by fractures. The isoplanes thus have a rather smooth look. If the vertical cross-section is slightly turned clockwise, though, several fractures are located in the remaining volume of the model which results in wave-like disturbances especially at a distance to the geotechnical openings as seen in Fig. 11.46.

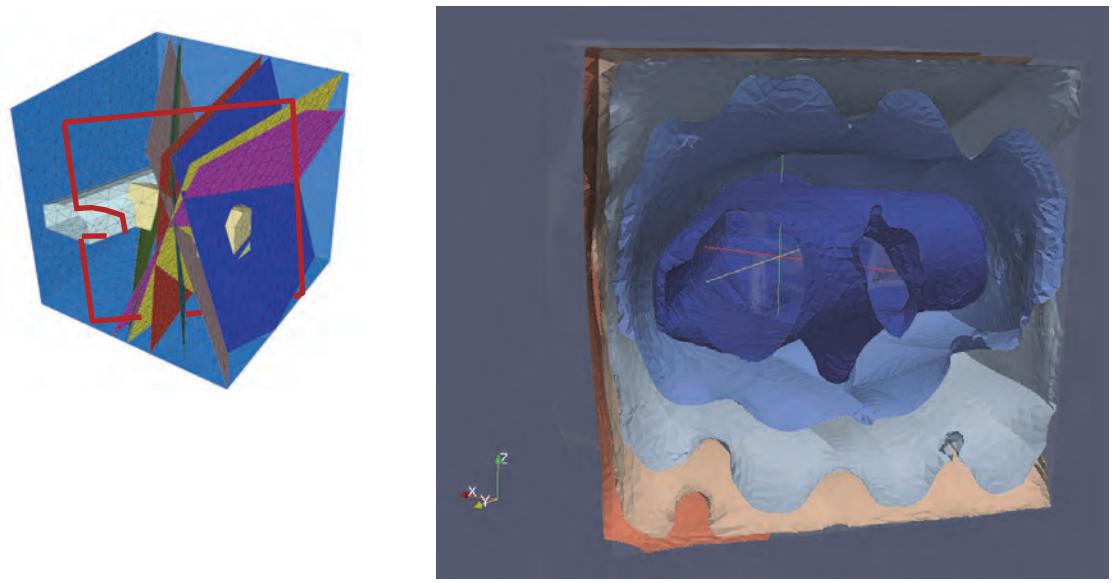


Fig. 11.46 Influence of fractures on the isoplanes of the dynamic pressure

11.2.5.7.2 Flow velocity

Exemplary for the calculated flow field in a fracture Fig. 11.47 depicts a wire plot of the model including a fracture highlighted in red. Direction and flow velocity in the fracture are indicated by equally spaced vectors of varying length. Flow occurs from the cube surface towards the tunnels and boreholes as expected from the pressure plots. A significant influence on the flow field from other fractures is not expected, and in fact cannot be observed, because all fractures are of comparable orientation and assumed to be larger than the model domain. Therefore, all of them simply connect the surfaces of the model with the geotechnical openings resulting in comparable pressure gradients, too.

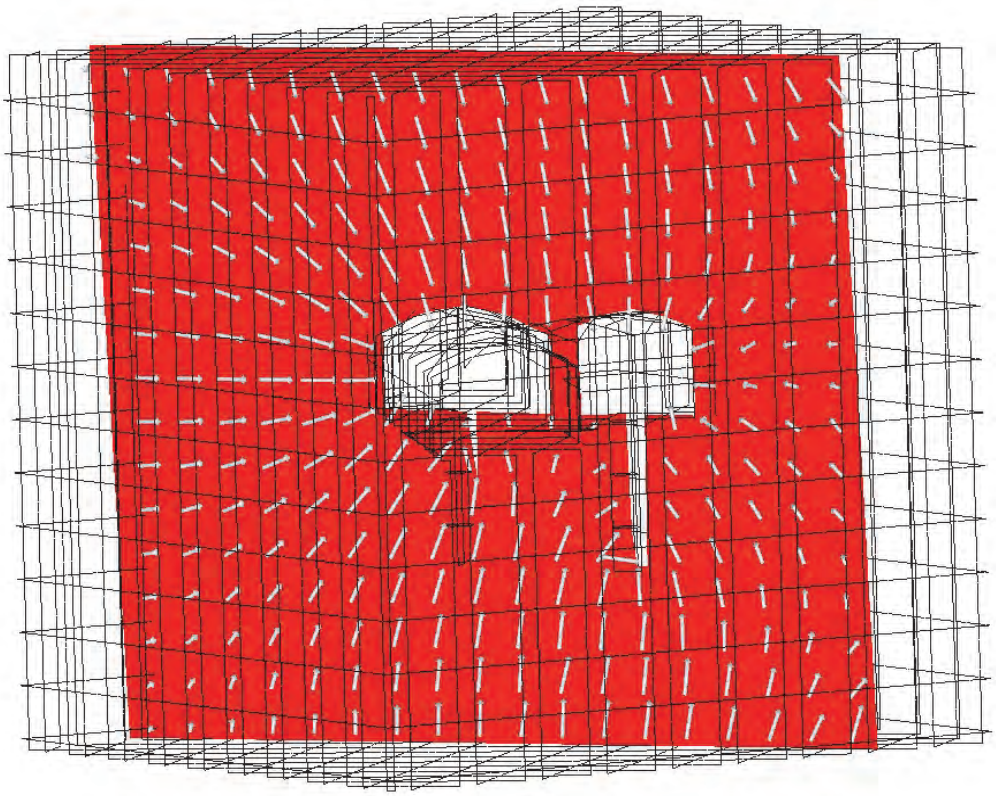


Fig. 11.47 Flow field in a fracture

The situation is different for the flow field in the matrix. Here, the fractures provide hydraulic shortcuts for the water on its way from the cube surface to the openings. The plot of the velocity field in a horizontal cross-section through the matrix provides a meaningful example. In Fig. 11.48 the flow direction is indicated by vectors and the flow rates are visualised by an underlying contour plot. The position of the intersections with the fractures can clearly be determined by the abrupt colour changes in the contour plot. Where this happens the fractures influence the flow field in the matrix by deflecting the stream lines. At some locations the colour changes are accompanied by visible changes in the flow direction as well.

The highest flow velocities can of course be found where the openings are closest to the cube surface, most obvious at the end of the TASD-tunnel. Interestingly, the area showing the highest velocities appears to be more or less symmetrically arranged around the tunnel face despite the fact, that the tunnel face is not parallel to the cube surface. In a homogeneous domain the location with the highest flow velocity would have been expected at the tunnel edge closest to the surface. But apparently, the frac-

ture system lowers the flow resistance to the other edge in such a way that inflow into the tunnel is more or less equally distributed along the tunnel face.

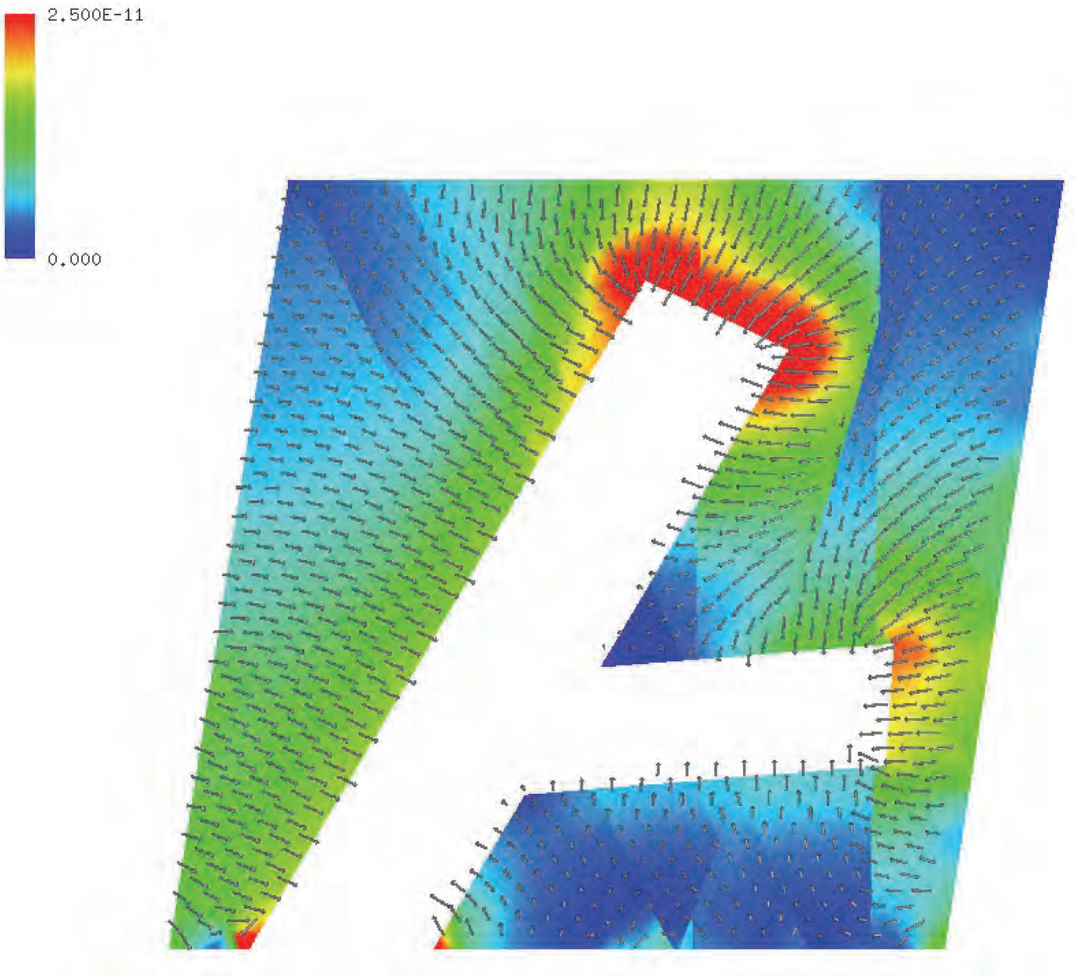


Fig. 11.48 Flow field in the matrix

11.2.5.7.3 Water outflow

Since /BOC 11/ provide no means of comparing the results described above with data from the HRL at least a rough check was devised. Water flow into the openings was calculated to be compared with flow data from a different location in the HRL. The V2-fracture system at niche 2715 in the HRL had been found to be highly permeable and to produce about 50 ml s^{-1} /KUL 02/. This compares nicely to the amount of water flowing out of tunnels and boreholes which amounts to approx. 180 g s^{-1} in the model especially considering that this value comes from the first and uncalibrated model.

11.3 Modelling with free groundwater table

11.3.1 Flow within a dam

For testing the functionality of free surface flow in d³f in principle, the well-known example of flow through a dam is used. Here, only a plausibility test can be provided. A code verification in detail and with realistic applications will be performed in the framework of a subsequent project.

11.3.1.1 Introduction

A schematic diagram of the groundwater flow within a dam is shown in Fig. 11.49. The dam is supposed to consist of a homogeneous material. The dam is supposed to consist of a homogeneous, isotropic material. On the left hand side the water table is defined by the sea level or the surface of a lake which are assumed here to be invariable with time. On the right hand side of the dam another surface water is situated with a lower level than the water on the left hand side. The bottom of the dam is assumed to be impervious. The water density does not vary with space or time. The resulting flow thus represents a potential flow field where the potential is given in terms of the piezometric/hydraulic head.

With a view to potential flow the sketch in Fig. 11.49 shows an interesting variety of boundary conditions. Where the dam is adjoined to a water reservoir the hydraulic head is constant meaning that the boundary is also an an equipotential line. The impermeable bottom represents a streamline since no flow is allowed across this boundary. The same applies to the free groundwater table at steady-state conditions. The seepage section of the boundary where water leaves the dam above the level of the surface water has a varying potential i. e. the hydraulic head equals the height of the dam surface.

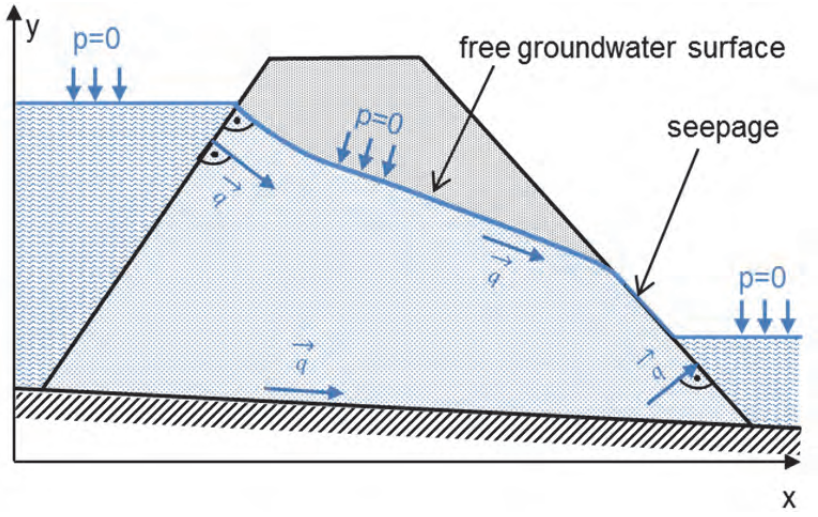


Fig. 11.49 Schematic diagram of groundwater flow within a dam

11.3.1.2 Hydrogeological model

The generic dam model investigated here has a length of 22.5 m and a height of 10.8 m. The dam has a permeability $k = 10^{-10}$ m² and a porosity $\phi = 0.2$. The initial state of the hydrogeological model and the boundary conditions are shown in Fig. 11.50. For the initial level set function $\phi^0(x) = \phi(x, 0) = y - 9.3$ was chosen (see also section 6.) A Dirichlet boundary condition p = 0 Pa is assigned to the free surface boundary and to the right hand side of the dam above water level. To the lateral boundaries below water level hydrostatic pressure is assigned. The bottom boundary is closed to flow. Any influence of salt is neglected: The concentration is set to 0, density and viscosity are constant. Dispersion and diffusion are neglected.

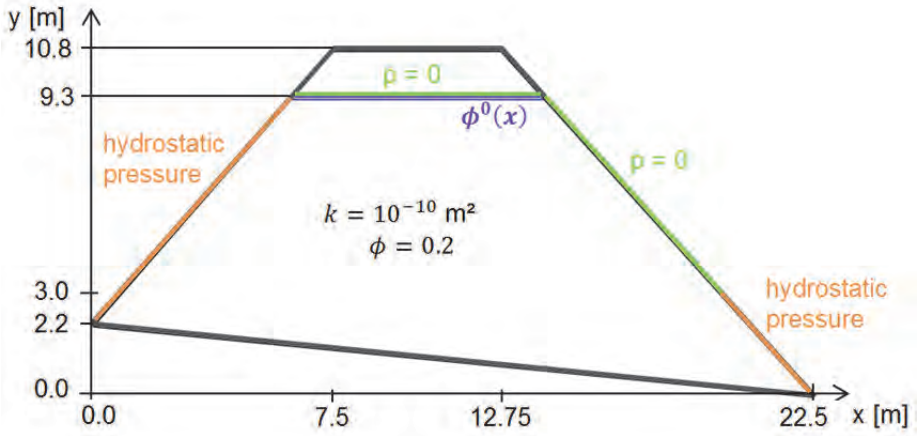


Fig. 11.50 Hydrogeological model for the dam with boundary conditions; blue line: initial level set function $\boldsymbol{\phi^0(x)}$

11.3.1.3 Results

After a model time of about 1 month steady state was reached. The resulting groundwater surface, pressure and velocity field are depicted in Fig. 11.51. The isobars range from 0 (blue) to 70 kPa (red). The velocity reaches absolute values from about 10^{-7} m s^{-1} up to almost 10^{-8} m s^{-1}.

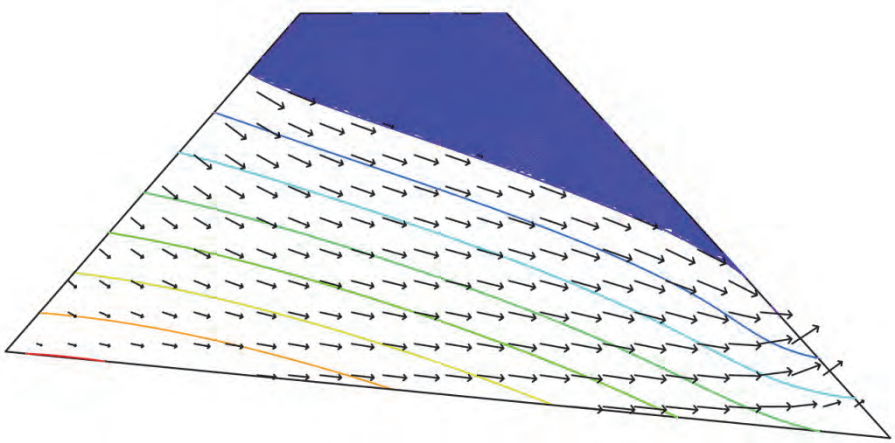


Fig. 11.51 Groundwater surface, velocity vectors and isobars at steady state. Blue area: No values are computed outside the saturated zone

Based on the general statements in the introduction the results are consistent with the expectations:

- The velocity vectors at the groundwater table as well as at the bottom boundary representing two explicitely known streamlines are indeed aligned to these features.
- Since streamlines in a potential field are always orthogonal to the equipotential lines it follows that water enters or leaves the model domain across the boundaries to surface waters at a right angle. This cannot clearly be seen in Fig. 11.51. Here, the geometric resolution or the accuracy of the graphical representation of the surface might not be sufficient.
- For the same reason the groundwater table – representing a streamline – connects to the dam surface on the left hand side at a right angle.
- Above the water table on the right hand side of the dam a seepage region has indeed been formed.

- The emergent angle for flow through the seepage area is not clearly defined because of the varying potential at the boundary. This angle has to be less than 90°.
- Consequently, the angle of the free groundwater surface to the dam surface is also not defined but significantly less than 90°.
- Usually, the curve shape of the free surface has an inflexion point.

Note that the isolines in Fig. 11.51 do neither show stream lines nor equipotential lines but the hydraulic pressure.

11.3.1.4 Conclusion

It was intended to demonstrate that free surface modelling with d³f is viable and that the computed level set function represents the free groundwater surface properly for a simple, typical test case. Using the well-known model of seepage flow through a dam this aim could be met qualitatively. The computed surface as well as the results for pressure and velocity are plausible and meet the expectations.

Further investigations of this feature are necessary. In test cases performed within the framework of subsequent projects results will have to be compared with measured data or results of other codes. Additionally, 3d test cases have to be considered.

12 Conclusion and outlook

12.1 Scaling in heterogeneous media

A stability criterion was derived for density-driven systems by means of homogenization theory. For systems orthogonal to gravity the derived stability number could predict the onset of fingering, dependent on density and viscosity contrasts, flow velocity and concentration gradients. Omitting the Oberbeck-Boussinesq approximation, timely stability predictions could be achieved, and the criterion could be extended to much higher density contrasts. Finally, dispersion could be included. For testing the stability number, an Elder-type system was used. The derived stability number is a function of the perturbation wavelength and the mixing zone width.

The criterion was extended to heterogeneous media, and an expression was determined that predicted stabilising and destabilising effects of variance and correlation length. Medium anisotropy was not included in the expression for the stability number.

12.2 Thermohaline-driven flow

The thermohaline flow problem was described mathematically, and the thermodiffusion effects were introduced. Different thermodynamical concepts were developed, investigated and compared with thermohaline flow and thermodiffusion models known from literature. The influence of the Soret and Dufour effects were investigated and discussed with the result that both are neglectable in the currently envisaged range of applications for d³f.

Finally, three field equations to be solved in d³f were derived, describing the mass balance of the fluid-phase, the mass balance of the solute and the energy balance of the mixture as a whole. Two variants were considered, the Boussinesq approximation and the complete equation system.

Theoretical and experimental analyses were performed to determine the scope of validity of the equations. As a 3d test case the evolution of a brine parcel in the case of negative and positive buoyancy was modelled also demonstrating the capability of d³f to handle models with more than 10^8 nodes.

12.3 Flow and transport in fractured media

Structures of reduced dimension representing fractures in a porous medium were established within d³f and r³t. The system of partial differential equations to be solved for a d-dimensional model changed into a system of both dimensions d and d-1.

The concept is based on the assumption that flow and transport within a fracture are independent of the processes within the surrounding matrix. Interface conditions for the fracture-matrix-interaction had to be formulated. Averaging over the fracture width was a special challange for the development of both the model and the numerical methods. In the former d³f the three differential equations to be solved were based on pressure and brine mass fraction as primary variables. To simplify the averaging process, the equations were re-formulated: Now the salt concentration is used as the second primary variable.

The finite volume discretisation had to be adapted. The fractures are represented by so called degenerated elements that have a thickness of zero. Grid generators and refinement algorithms were adapted.

The transport equations of r³t were treated analogously for every contaminant transported.

Different 2d and 3d test cases were successfully worked on to verify the code and to test its capability.

12.4 Free surface modelling and potential flow

D³f and r³t were enabled to model a free groundwater surface. Here, in accordance with the ug-philosophy, the numerical grid stays fixed, which leaves the task to distinguish between the nodes below and above water level. For this purpose a new level set method was developed where the free groundwater surface is represented in an implicit way as the zero level set of the level set function. These zero level set is computed by solving the advection equation.

This solving algorithm in d³f was tested using different examples. Additionally, a groundwater recharge model was implemented and tested successfully.

For models without any transport effects a feature was created that allows the computing of only potential flow within one step.

12.5 Numerical advances

The solvers within d³f were improved and optimized. A higher order finite volume method was introduced to improve accuracy.

New filtering algebraic multigrid (FAMG) methods were developed and implemented. These FAMG methods proved very suitable for computing problems with large fracture networks.

Additionally, the ug parallelisation concept was advanced to a flexible tool, the parallel communication layer (PCL). These PCL is applicable to many solvers and enables d³f and r³t to use modern parallel computers effectively.

12.6 Adaption of pre- and postprocessors

The postprocessor based on GRAPE (GRAphical Programming Environment) was advanced and extended by new methods to be able to visualise and analyse data effectively. New visualisation concepts had to be developed for data on fractures or the free groundwater surface. New methods for data extraction were studied.

A main focus lay on the development and implementation of visualisation concepts working on lower dimensional structures, another on the robust visualization and analysis of different types of data for implicitely described surfaces. The existing tool set for the interactive local data extraction was extended significantly.

A new PCL based graphical user interface was developed. Additionally, the interactive graphical tool ProMesh was adopted for geometrical model building and grid generation, especially for models containing fractures and internal hollow spaces like drifts or boreholes. With GISLab a new tool was developed that enables to read data for subsurface contour plots from GIS or pointwise data and to preprocess it for the subsequent processing in ProMesh. The existing grid generator ARTE was further developed and adapted to the new fracture discretisation.

12.7 Code verification

The codes were tested by means of various test cases. To verify the implementation of thermohaline flow, a 1d test case was compared with an existing analytical solution known from the literature and a 2d test case with a documented solution calculated with another numerical heat transport model. In both cases the d³f results agreed well with the reference solutions.

Testing fracture flow started with a simple matrix diffusion test case with one single fracture. The results from d³f were compared with an analytical solution and matched this solution very well. A second 2d test case for a large scale domain with 5 fractures was taken from the project ASTER (FKZ 02E9612 und FKZ 02E9622). The FEFLOW-model presented there was reconstructed with d³f and r³t which led also to a satisfying match of the results. As a third test case Task 8b of the Äspö Task Force on groundwater flow and transport of solutes (TF GWFTS) was modelled, a 3d case including 7 large-scale interesting fractures and various geotechnical openings. The ability of d³f to cope with a complex fracture system was tested on the basis of Task 8. During the procedure which led to reasonable results in the end some shortcomings in the pre- and postprocessing tools as well as in the solver became apparent and were resolved ac-cordingly.

Modelling free surface flow with d³f was only checked for plausibility based on the problem of flow within a dam. The results met the criteria derived from theoretical considerations concerning potential flow.

12.8 Outlook

The new features of d³f and r³t are tested now with regard to code verification. Most of the test cases performed were very simple. To enhance the confidence in accuracy and capability it is necessary to apply the codes to field cases and to compare the results with measured data or with the results of other models.

Thinking about data uncertainties and the reliability of forecasts, parameter variations and stochastic modelling will be indispensable in future. Because flow and transport models in strongly heterogeneous media remain complex and time-consuming, modern efficient methods for the handling of uncertainties have to be found.

On the other hand, there is a fast-paced progress in computer science. Software codes have to be adapted continuously to remain competitive even over short periods of time. Modern computers have hybrid structures, cache distribution and handling differ compared to classical parallel computers, and new processors as GPUs or Cell processors are used. To keep the codes d³f and r³t state-of-the-art one has to keep track of these developments and to advanced them if appropriate to be able to exploit these new high-performance technologies.

12.9 Summary

Originally, the codes d³f and r³t were developed to model 2d and 3d density-driven flow and transport of nuclides or contaminants in strongly heterogeneous porous media. By the extensions presented in this report they are now also empowered to model heat transport. Modelling of porous media is complemented with the explicit modelling of fractures. The possibility to model free surface flow provides the ability to take into account pumping wells and groundwater recharge, too. With a view to long-term safety analyses of repositories for radioactive waste the application field is enlarged to other host rocks than rock salt, such as mudstone and crystalline rocks. This leads to a significant fortification of the possibilities and of the reliability of simulations in this field. Moreover, the advances of d³f and r³t described above provide not only more versatile tools for safety analyses, but also represent significant advances in the field of numerics and software engineering.

References

/ADA 99/ Adalsteinsson, D., Sethian, J.: The fast construction of extension velocities in level set methods. J. Comput. Phys. 148 , 2-22, 1999.

/AGA 63/ Agar, J.N.: Adv. Electrochem. Eng., 3(31), 1963.

/ANG 09/ Angot, P., Boyer, F., Hubert, F.: Asymptotic and numerical modelling of flow in fractured porous media. ESAIM: M2AN. Mathematical Modelling and Numerical Analysis 43, 239-275, 2009.

/ASL 03/ Aslam, T. D.: A partial differential equation approach to multidimensional extrapolation. J. Comput. Phys. 152, 349-355, 2003.

/ATT 02/ Attinger, S.; Eberhard, J.; Neuss, N.: Filtering Procedures for Flow in Heterogeneous Porous Media: Numerical Results. Computing and Visualization in Science, Volume 5, No 2, 67-72, 2002.

/BAD 05/ Bader, S., Kooi, J.: Modeling of solute and water transport in semi-permeable clay membranes: Comparison with experiments. Adv. Water Resour. 28, 203-214, 2005.

/BAL 88/ Balmorth, N.J., Cast, A.R.R., Julien, K.A.: Thermohaline convection with nonlinear salt profiles. Physics of Fluids 10, 819-828, 1988.

/BAR 93/ Barrett, R., Berry, M., Chan, T. F., Demmel, J., Donato, J., Dongarra, J., Eijkhout, V., Pozo, R., Romine, C., van der Vorst, H.: Templates for the solution of linear systems: building blocks for iterative methods. SIAM, Philadelphia, PA., 1993.

/BAS 94/ Bastian P., and Wittum G.: Robustness and adaptivity: The UG concept. In: Multigrid Methods IV, proceedings of the fourth european multigrid conference, ed. by Hemker, P., Wesseling, P., 1994.

/BAS 97/ Bastian, P., Birken, K., Johannsen, K., Lang, S., Neuß, N. Rentz-Reichert, H., Wieners, C.: UG – A flexible software tool for solving partial differential equations. Computing and Visualization in Science 1, 27-40, 1997.

/BAS 99/ Bastian, P., Chen, Z., Ewing, R. E., Helmig, R., Jakobs, H., Reichenberger, V.: Numerical simulation of multiphase flow in fractured porous media. In: Chen, Z., et al. (Eds.), Numerical treatment of multiphase flows in porous media. Proceedings of the international workshop, Beijing, China, August 2–6, 1999. Vol. 552 of Lect. Notes Phys. Springer, Berlin, 50-68, 2000.

/BAS 00/ Bastian, P.; Johannsen, K.; Lang, S.; Nägele, S.; Reichenberger, V.; Wieners, C.; Wittum, G.; Wrobel, C.: Parallel solution of partial differential equations with adaptive multigrid methods on unstructured grids. In: Jäger, W.; Krause, E. (eds.): High performance computing in science and engineering. Springer, Berlin, 506-519, 2000.

/BAS 05/ Bastian, P.; Droske, M.; Engwer, C.; Klöfkorn R.; Neubauer, T.; Ohlberger, M.; Rumpf, M.:.In*:* Kornhuber, R. (ed.) et al.: Domain decomposition methods in science and engineering. Selected papers of the 15th international conference on domain decomposition, Berlin, Germany, July 21-25, 2003. Berlin: Springer. Lecture Notes in Computational Science and Engineering 40, 167-174, 2005.

/BEA 72/ Bear, J., Dynamics of fluid in Porous Media. Dover Publications, INC. New York, 1972.

/BEA 77/ Bear, J., On the aquifer's integrated balance equations. Adv. Water Resour. 1 (1), 15–23, 1977.

/BEA 79/ Bear, J., 1979. Hydraulics of Groundwater. Dover Publications. Inc., Mineola, 1979.

/BEA 88/ Bear, J.: Dynamics of Fluids in Porous Media. Dover Publication, Inc., 1988

/BEA 90/ Bear, J., Bachmat, Y.: Introduction to Modeling of Transport Phenomena in Porous Media. Kluwer Academic Publishers, Dordrecht, Boston, London, 1990.

/BEA 91/ Bear, J., Bachmat, Y.: Introduction to Modeling of Transport Phenomena in Porous Media of Theory and applications of transport in porous media 4. Kluwer Academic, Dordrecht, 1991.

/BEA 93/ Bear, J., Tsang, C.-F., deMarsily, G.: Flow and contaminant transport in fractured rocks. Academic Press, Inc., New York, 1993.

/BEH 82/ Behie, A., Vinsome, P.K.W.: Block iterative methods for fully implicit reservoir simulation, Soc. Petroleum Eng. J, 22, 658-668, 1982.

/BEN 78/ Bensoussan, A., Lions, J. L., Papanicolau, G.:. Asymptotic Analysis for Periodic Structures. North-Holland, Amsterdam, 1978.

/BEN 00/ Bennethum, L.S., Murad, M.A., Cushman, J.H.: Macroscale thermodynamics and the chemical potential for swelling porous media. Transport in porous media, 39, 187 – 225, 2000.

/BEN 01/ Benano-Melly, L.B., Caltagirone, J.-P., Faissat, B., Montel, F. Costeséques, P.: Modeling Soret coefficient measurement experiments in porous media considering thermal and solutal convection. Int. J. Heat and Mass Transfer, 44, 1285 – 1297, 2001.

/BEN 07/ Bennethum, L.S.: Theory of flow and deformation of swelling porous materials at the macroscale. Computer and Geotechnics, 34, 267 – 278, 2007.

/BER 00/ De Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry (2nd revised ed.), Chapter 3: Polygon Triangulation, 45-61, 2000.

/BIR 94/ Birken, K. and Bastian, P.: Dynamic Distributed Data (DDD) in a parallel programming environment, specification and functionality. Citeseer, 1994.

/BIR 98/ Birken, K.: Ein Modell zur effizienten Parallelisierung von Algorithmen auf komplexen, dynamischen Datenstrukturen. PhD Thesis, Universität Stuttgart, 1998.

/BIR 00/ Birthler, H.; Fein, E.; Schneider, A.: Validierung von Einzeleffekten in Grundwassermodellen. FKZ-02 E 8865 0 Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-150, Braunschweig, 2000.

/BOC 11/ Bockgård, N., Vidstrand, P., Åkesson, M.: Modelling the interaction between engineered and natural barriers – An assessment of a fractured bedrock description in the wetting process of bentonite at deposition tunnel scale. Technical Committee of Task 8, SKB, Rev. 2011-02-23, 2011.

/BOU 03/ Boussinesq, J.: Theorie analytique de la chaleur, Vol. 2. Gauthier-Villlars, Paris, 1903.

/BRA 11/ Brandt, A., Brannick, J., Kahl, K., Livshits, I.: Bootstrap AMG. SIAM J. on Scientific Computing, 33(2):612-632, 2011.

/BRE 62/ Brenner, H.: The Diffusion Model of Longitudinal Mixing in Beds of Finite Length: Numerical Values. Chem. Eng. Sci. 17, 229–243, 1962.

/BRE 01/ Brezina, M., Cleary, A.J., Falgout, R. D., Henson, V. E., Jones, J. E., Manteuffel, T. A., McCormick, S. F., Ruge, J. W.: Algebraic Multigrid Based on Element Interpolation (AMGe). SIAM Journal on Scientific Computing, 22(5): 1570-1592, 2001.

/BUE 91/ Bues, M.A. and Aachib, M.: Effect of the heterogeneity of the solutions on the parameters of miscible displacement in saturated porous medium, Experiments in Fluids 11, 25-32, 1991.

/CAI 90/ Cai, Z.: On the finite volume element method. Numer. Math. 58 (1), 713-735, 1990.

/CAR 59/ Carslaw, H.S. and Jaeger, J.C.: Conduction of Heat in Solids, Oxford 1959.

/CEL 05/ Celestino, A., Leonardi, E.: The effect of thermodiffusion on the stability of a salinity gradient solar pond. Int. J. of Heat and Mass Transfer 48, 4633-4639, 2005.

/CEL 06/ Celestino, A., Leonardi, E., Maciocco, L.: A computational study of salt diffusion and heat extraction in solar pond plants. Solar Energy 80, 1498-1508, 2006.

/CER 05/ Cermelli, P., Fried, E., Gurtin, M. E.: Transport relations for surface integrals arising in the formulation of balance laws for evolving fluid interfaces. J. Fluid Mech. 544, 339–351, 2005.

/CHA 88/ Chang, S., Slattery, J.C.: A linear stability analysis for miscible displacements, Transport in porous media 1, 179-199, 1988.

/CIO 99/ Cioranescu, D., Donato, P.: An introduction to Homogenization, Oxford University Press, 1999.

/CLE 07/ Clees, T. Ganzer, L.: An Efficient Algebraic Multigrid Solver Strategy for Adaptive Implicit Methods in Oil Reservoir Simulation. paper SPE 105789 presented at the 2007 SPE Reservoir Simulation Symposium, Houston, TX, Feb 28-30, 2007.

/COD 09/ Codehaus Foundation, Groovy 1.7, http://groovy.codehaus.org, 2009

/COS 90/ Coskuner, G., Bentsen, R.G.: An extended theory to predict the onset of viscous instabilities for miscible displacements in porous media, Transport in porous media 5, 473-490, 1990.

/COS 02/ Costeséques, P., Fargue, D., Jameth, Ph.: Thermodiffusion in porous media and its consequences. In Thermal Nonequilibrium Phenomena in Fluid Mixtures. Köhler, W., and Wiegand, S. (Ed.). Springer-Verlag, 2002.

/DAG 88/ Dagan, G.: Time-dependent macrodispersion for solute transport in anisotropic heterogeneous aquifers, Water Resources Research 24, 1491-1500, 1988.

/DAV 94/ Davison, C.C., Chan, T., Brown, A.: The disposal of Canada´s nuclear fuel waster: The geosphere model for postclosure assessment. Atomic Energy of Canada Limited (AECL) Research, AECL-10719, 1994.

/DEA 03/ http://www.dealii.org/, 2003

/DEG 69/ De Groot, S., Mazur, P. : Non-Equilibrium Thermodynamics, 2nd reprint. North-Holland Publishing Company, 1969.

/DEN 68/ Denbigh, K.: The Principles of Chemical Equilibrium, 4th Edition. Cambridge University Press, 1968.

/DHI 10/ DHI-WASY GmbH: FEFLOW Classic. Finite Element Subsurface Flow & Transport Simulation System. User Manual. Berlin (DHI-WASY GmbH), 172 p, 2010.

/DIE 81/ Diersch, H.-J.: Primitive variables finite element solutions of free convection flows in porous media. Zschr. Angew. Math. Mech., 61, 325-337, 1981.

/DIE 02/ Diersch, H.J.G. and Kolditz, O.: Variable-density flow and transport in porous media: approaches and challenges, Advances in Water Resources, 25, 899-944, 2002.

/DIE 04/ Diersch, H-J. G.: DHI-WASY GmbH: FEFLOW 5.1. Finite Element Subsurface Flow & Transport Simulation System. Use'r Manual. Berlin (WASY GmbH), 168 p, 2004.

/DIE 05/ Diersch, H.-J.G., Kolditz, O.: Variable-density flow and transport in porous media: approaches and challenges. In Wasy Software FEFLOW – Finite Element Subsurface flow and Transport Simulation System, Vol. 2, WASY GmbH, 2005.

/ECL 11/ Eclipse Foundation, Eclipse 3.6.1, http://www.eclipse.org/, 2011.

/ELD 67/ Elder, J .W.: Transient convection in porous media. Journal of Fluid Mechanics., 27, S. 609-623, 1967.

/FEI 91/ Fein, E.: Statusreport: Grundwasserprogramme mit variabler Dichte. GSF – Forschungszentrum für Umwelt und Gesundheit, GmbH. GSF-31/91, Braunschweig, 1991.

/FEI 99/ Fein, E.; Schneider, A. (eds.): d^3f - ein Programmpaket zur Modellierung von Dichteströmungen. Final report. FKZ-02 C 0465 0. Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-139, Braunschweig 1999.

/FEI 04/ Fein, E. (ed.): Software Package r^3t. Model for Transport and Retention in Porous Media. Final report. FKZ-02 E 9148/2. Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-192, Braunschweig 2004.

/FEI 08/ Fein, E.; Kröhn, K.-P.; Noseck, U.; Schneider, A.: Modelling of field-scale pollutant transport. Final report. FKZ-02 E 9934. Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-231, Braunschweig 2008.

/FEU 03/ Feuchter D., Heppner I., Sauter S., Wittum G.: Bridging the gap between geometric and algebraic multi-grid methods, Computing and Visualization in Science, 6(1):1-13, 2003.

/FLU 61/ Flügge, S.: Lehrbuch der theoretischen Physik. Springer-Verlag, 1961.

/FLU 09/ Flügge, J.: Radionuclide Transport in the Overburden of a Salt Dome – The Impact of Extreme Climate States. Dr. Hut, München, 2009

/FRE 98/ Frey, P. J., Borouchaki, H.,: Geometric surface mesh optimization. Computing and Visualization in Science 1, 113-121, 1998.

/FRO 96/ Frolkovič, P.: Finite Volume Discretization of Density Driven Flows in Porous Media. In: Benkhaldoun, F., Vilsmeier, R. (eds.) Finite Volumes for Complex Applications, Hermes, Paris S. 433 – 440, 1996.

/FRO 96a/ Frolkovič, P., Knabner, P.,: Consistent velocity approximations in finite element or volume discretizations of density driven flow. In: Aldama, A. A., et al. (Eds.), Computational Methods in Water Resources XI. Computational Mechanics Publication, Southhampten, pp. 93–100, 1996.

/FRO 97/ Frolkovič, P., Knabner, P., Tapp, C., Thiele, K.: Adaptive finite volume discretization of density driven flows in porous media. Preprint 220, Institut für Angewandte Mathematik, Universität Erlangen-Nürnberg, FR Germany, 1997. Lecture Notes to INRIA Rocquencourt: Transport de contaminants multiespeces en milieux poreux, Jun 2-6, 1997.

/FRO 98/ Frolkovič, P.,: Consistent velocity approximation for density driven flow and transport. In: Van Keer, R., at al. (Eds.), Advanced Computational Methods in Engineering, Part 2: Contributed papers. Shaker Publishing, Maastricht, pp. 603–611, 1998.

/FRO 98a/ Frolkovič, P.,: Maximum principle and local mass balance for numerical solutions of transport equation coupled with variable density flow. Acta Mathematica Universitatis Comenianae 1 (68), 137–157, 1998.

/FRO 98b/ Frolkovič, P.: Discretization in d^3f - A Simulator for Density-Driven Flow, 39 pages; in User's Manual (compiled by E. Fein); Gesellschaft fuer Anlagen- und Reaktorsicherheit (mbH), Braunschweig, 1998.

/FRO 01/ Frolkovič, P., De Schepper, H.: Numerical modeling of convection-dominated transport coupled with density-driven flow in porous media. Advances in Water Resources 24, 63-72, 2001.

/FRO 07a/ Frolkovič, P., Mikula, K.: Flux-based level set method: A finite volume method for evolving interfaces. Applied Numerical Mathematics 57, 436 – 454, 2007.

/FRO 07b/ Frolkovič, P. & Mikula, K.: High-resolution flux-based level set method. SIAM J. Sci. Comp. 29, 579-597, 2007.

/FRO 09/ Frolkovič, P. & Wehner, C.: Flux-based level set method on rectangular grids and computations of first arrival time functions. Computing and Visualization in Science 12, 297-306, 2009.

/FRO 10a/ P. Frolkovič, P. Zacharovská: Numerical Modelling of Dynamic Groundwater Table using Level Set Formulation. In: XVIII Conference on Computational Methods in Water Resources, Barcelona 2010.

/FRO 10b/ Frolkovič, P.: Flux-based level set method for extrapolation along characteristics using immersed interface formulation. MAGIA 2010, Publishing house of STU, 15-26, 2010.

/FUC 01/ Fuchs, A.: Almost Regular Triangulations of Trimmed NURBS-Solids. Engineering with Computers 17, 55-65, 2001.

/GAJ 03/ Gajo, A., Loret, B.: Finite element simulations of chemo-mechanical coupling in elastic-plastic homoionic expansive clays. Comput. Methods Appl. Mech. Engrg., 192, 3489-3530, 2003.

/GAM 95/ Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995.

/GAR 87/ Gärtner, S.: Zur diskreten Approximation kontinuumsmechanischer Bilanzgleichungen. Bericht Nr. 24/1987, Institut für Strömungsmechanik und Elek-tronisches Rechnen im Bauwesen, Universität Hannover, 1987.

/GEI 03/ Geiser, J.: Diskretisierungsverfahren für Systeme von Konvektions-Diffusions-Dispersions-Reaktions-Gleichungen und Anwendungen. PhD thesis, Universität Heidelberg, 2003.

/GEL 83/ Gelhar, W.L. and Axness C.L.: 'Three-dimensional stochastic analysis of macrodispersiion in aquifers', Water Resources Research 19, 161-180, 1983.

/GEL 93/ Gelhar, W.L.: Stochastic Subsurface Hydrology, Prentice Hall, New Jersey, 1993.

/GRA 82/ Gray, W. G.: Derivation of vertically averaged equations describing multiphase flow in porous media. Water Resources Research 18, 1705-1712, 1982.

/GRA 05/ Graf, T.: Modeling Coupled Thermohaline Flow and Reactive Solute Transport in Descretely-Fractured Porous Media. Ph.D. thesis, Laval University, Quebec City, Canada, 2005

/GRA 05a/ Graf, T., Therrien: Variable-density Groundwater Flow and Solute Transport in Porous Media Containing Nonuniform Discrete Fractures. Adv. Water Resour. 28, 1351-1367, 2005.

/GRA 07/ Graf, T., Therrien, R.: Thermohaline groundwater flow and single-species reactive solute transport in fractured porous media. Adv. Water Resour., 30, S. 742-771, 2007.

/GRA 83/ Gray, W. G.: Constitutive Theory for Vertically Averaged Equations Describing Steam-Water Flow in Porous Media. Water Resources Research 19, 1501–1510, 1983.

/GRA 99/ GRAPE Manual. Version 5.3. SFB 256, University of Bonn. Download at: http://numod.ins.uni-bonn.de/grape/ (date of access: October 2008).

/GRI 09a/ Grillo, A., Wittum, G., Giaquinta, G., Micunonic, M.V.: A multiscale analysis of growth and diffusion dynamics in biological materials. Int. J. Eng. Sci., 47, S. 261-283, 2009.

/GRI 09b/ Grillo, A., Federico, S., Wittum, G., Imatani, S. Giaquinta, G., Micunovic, M.V.: Evolution of a fibre-reinforced growing mixture. Il Nuovo Cimento C, 32, 97-119, 2009.

/GRI 10/ Grillo, A., Wittum, G.: Growth and mass transfer in multi-constituent biological materials. American Institute of Physics, CP1281, 355 – 359, 2010.

/GRI 10a/ Grillo, A., Logashenko, D., Stichel, S., Wittum, G.: Simulation of density-driven flow in fractured porous medis, Advances in Water Resources 33(12), 1494-1507, 2010.

/GRI 10b/ Grillo, A., Lampe, M., Wittum, G.: Three-dimensional simulation of the thermohaline-driven buoyancy of a brine parcel. Computing and Visualization in Science, 13, 287-297, 2010.

/GRI 11/ Grillo, A., Lampe, M., Wittum, G.: Modelling and Simulation of Temperature-Density-Driven Flow and Thermodiffusion in Porous Media. Journal of Porous Media, v14.i8.20, 671-690, 2011.

/GRO 54/ De Groot, S., Mazur, P.: Non-Equilibrium Thermodynamics, 2nd reprint. North-Holland Publishing Company, 1954.

/GRO 99/ Gropp, W., Lusk, E., Skjellum, A.: Using mpi: portable parallel programming with the message passing interface, 1999.

/HAC 85/ Hackbusch, W.: Multi-Grid Methods and Applications. Springer-Verlag, 1985.

/HAG 75/ Hageman, L. A., Porsching, T. A.: Aspects of Nonlinear Block Successive Overrelaxation. SIAM Journal on Numerical Analysis, 12, 316-335, 1975.

/HAL 02/ Halloway, S. D.: Component Development for the Java Platform, 2002.

/HAR 00/ Harbaugh, A. W., Banta, E. R., Hill, M.C., McDonald, M.G.: MODFLOW-2000, The U.S. Geological Survey modular ground-water model–User Guide to modularization concepts and the ground-water flow process: U.S. Geological Survey, Reston, VA, 2000.

/HAS 86/ Hassanizadeh, S. M.: Derivation of basic equations of mass transport in porous media, Part 1. Generalized Darcy's and Fick's laws. Adv. Water Resour. 9, 196–206, 1986.

/HAS 88/ Hassanizadeh, S.M., Gray, W.G.: General conservation equations for multi-phase systems: 3. Constitutive Theory for porous media flow. Adv. Water Resour., 3, 25-40, 1988.

/HAS 89/ Hassanizadeh, S. M., Gray, W. G.: Boundary and Interface Conditions in Porous Media. Water Resources Research 25 (7), 1705-1715, 1989.

/HAS 89a/ Hassanizadeh, S. M., Gray, W. G.: Derivation of Conditions Describing Transport Across Zones of Reduced Dynamics Within Multiphase Systems. Water Resources Research 25, 529–539, 1989.

/HAS 90/ Hassanizadeh, S. M., Gray, W. G.: Mechanics and thermodynamics of multiphase flow in porous media including interphase boundaries. Adv. Water Resources 13, 169-186, 1990.

/HEL 05/ Held, R.; Attinger, S.; Kinzelbach; W.: Homogenization of the Henry Problem in Heterogeneous Formations. Water Resources Research, 41 (11), W11420, 2005.

/HEN 64/ Henry, H. R.: Interfaces between salt water and fresh water in coastal aquifers, US Geological Survey Water- Supply Paper 1613-C, Sea Water in Coastal Aquifers: C35-C70, 1964.

/HEN 64a/ Henry, H. R., 1964. Effects of dispersion on salt encroachment in coastal aquifers. In: Sea water in coastal aquifers. USGS Water Supply Paper 1613-c, 70–84, 1964.

/HEN 01/ Henson, V., Vassilevski, P. S.: Element-free AMGe: General Algorithms for Computing Interpolation Weights in AMG. SIAM J. Sci. Comput., 23:629--650, 2001.

/HOL 98/ Holzbecher, E. O.: Modeling density-driven flow in porous media. Springer, Berlin, Heidelberg, 1998.

/HOL 01/ Holstad, A.: Temperature-driven flow in porous media using a Mixed Finite Element Method and Finite Volume Method. Adv. Water Resour., 24, 843-862, 2001.

/HYA 83/ Hyakorn, P.S., Pinder, G.F.: Computational Methods in Subsurface Flow. Academic Press, 1983.

/IBM 91/ IBM, Visualization Data Explorer, http://www.opendx.org, 1991

/ING 73/ Ingle, S.E., Horne, F.H.: The Dufour Effect. J. Chem. Phys., 59(11), 5882 – 5894, 1973.

/JOH 97/ Johannsen, K.: An aligned 3D-Finite-Volume Method for Convection-Diffusion Problems. Notes on Num. Fluid Mech., 59, 227-243, 1997.

/JOH 02/ Johannsen, K.; Kinzelbach, W.; Oswald, S.; Wittum, G.: The saltpool benchmark problem - numerical simulation of saltwater upconing in a porous medium. Advances in Water Resources, 25 (3), 335-348, 2002.

/JOH 02a/ Johannsen, K.: The Elder Problem - Bifurcations and steady state solutions, in: Computational Methods in Water Resources. S.M. Hassanizadeh et. al. (eds), 47(1), 485-492, 2002.

/JOH 03/ Johannsen, K.: On the validity of the Boussinesq approximation for the Elder problem, Computational Geosciences, 7 (3), 169-182, 2003.

/JOH 04/ Johannsen, K.: Numerische Aspekte dichtegetriebener Strömung in porösen Medien. professorial dissertation (habilitation), 2004.

/JOH 05/ Johannsen, K.: A Symmetric Smoother for the Nonsymmetric Interior Penalty Discontinuous Galerkin Discretization. ICES Report 05-23, University of Texas at Austin, 2005.

/JOH 05a/ Johannsen, K.: A robust 9-point ILU smoother for anisotropic problems. IWR Preprint, University of Heidelberg, 2005/10, 2005.

/JOH 05b/ Johannsen K. Multigrid methods for nonsymmetric interior penalty discontinuous Galerkin methods. ICES Report 05-23, University of Texas at Austin, 2005.

/JOH 06/ Johannsen, K., Numerical Aspects of Density Driven Flow in Porous Media, Proceedings of the CMWR XVI, Copenhagen, Denmark, June 19-22, 2006.

/JOH 06a/ Johannsen, K., Oswald, S., Held, R., Kinzelbach, W.: Numerical simulation of three-dimensional saltwater-freshwater fingering instabilities observed in a porous medium. Advances in Water Resources, 29(11): 1690-1704; 2006.

/JOH 07/ Johannsen, K.: Multilevel Methods for Nonsymmetric Discontinuous Galerkin Methods. Proceedings of the 20th Nordic Seminar on Computational Mechanics NSCM20, Gothenburg, December 2007.

/KAH 09/ Kahl, K.:Adaptive Algebraic Multigrid for Lattice QCD Computations. PhD thesis, University of Wuppertal, 2009.

/KAR 95/ Karimian, S. M., Schneider, G. E.: Pressure-based control-volume finite element method for flow at all speeds. AIAA Journal 33 (9), 1611-1618, 1995.

/KEM 94/ Kempers, L.J.T.M. and Haas, H.: 'The dispersion zone between fluids with different density and viscosity in a heterogeneous porous medium', J. Fluid Mech. 267, 299-324, 1994.

/KES 05/ Keesmann, S.; Noseck, U.; Buhmann, D.; Fein, E., Schneider, A.: Modellrechnungen zur Langzeitsicherheit von Endlagern in Salz- und Granitformationen. FKZ-02 E 9239. Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-206, Braunschweig, 2005.

/KNA 96/ Knabner, P., Frolkovič, P.: Consistent velocity approximations in finite element and volume discretizations of density driven flow in porous media. In Aldama A. A. et al., editor, Computational Methods in Water Resources XI: 93-100. Computational Mechanics Publications, Southampton, Boston, 1996.

/KNA 98/ Knabner, P., Tapp, C., Thiele, K.: Adaptive finite volume discretization of density driven flows in porous media. Acta Mathematica Universitatis Comenianae, 67(1), 1998.

/KOH 09/ Kohlmeier, M., Maßmann, J., Wulkau, M., Ziefle, G.: RockFlow 5 User's Manual, Version 5.1. Institute of Fluid Mechanics and Environmental Physics in Civil Engineering, Leibniz-Universität Hannover, 2009.

/KOL 03/ Kölling, M., Quig, B., Patterson, A., Rosenberg, J.: The BlueJ system and its pedagogy, Journal of Computer Science Education 13, 1-12, 2003.

/KRA 06/ Kraus, J., Schicho, J.: Algebraic multigrid based on computational molecules, 1: Scalar elliptic problems. Computing, 77, 57-75, 2006.

/KRA 07/ Kraus, J.: Algebraic multigrid based on computational molecules, 2: Linear elasticity problems. SIAM J. Sci. Comput., 30, 505-524, 2007.

/KRE 03/ Kretz, V., Berest, P., Hulin, J.P. and Salin, D.: An experimental study of the effects of density and viscosity contrasts on macrodispersion in porous media, Water Resources Research 39(2), 1032, 2003.

/KRO 91/ Kröhn, K.-P.: Simulation von Transportvorgängen im klüftigen Gestein mit der Methode der Finiten Elemente. Bericht 29/1991, Institut für Strömungsmechanik und Elektronisches Rechnen im Bauwesen, Universität Hannover, 1991.

/KRO 10/ Kröhn, K.-P.: State Variables for Modelling Thermohaline Flow in Rocks. Status report, FKZ-02 E 10336, Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-268, Braunschweig, Dezember 2010.

/KUL 02/ Kull, H. (ed.), Helmig, R., Jacobs, H., Jockwer, N., Kröhn, K.-P., Zimmer, U.: Two-Phase-Flow Experiment in the Fractured Crystalline Rock of the Äspö Hard Rock Laboratory. Final Report. FKZ-02 E 9027 8. Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-183, Braunschweig, 2002.

/LAC 03/ Lacroix, S. Vassilevski, Y., Wheeler, J., Wheeler, M.: Iterative solution methods for modeling multiphase flow in porous media fully implicitly. SIAM J. Sci. Comput., 25, 905-926, 2003.

/LAN 84/ Landau, L.D., Liftschitz, E.M.: Fluid Mechanics – Course of Theoretical Physics. 2nd edition, Pergamon, UK, 1984.

/LAN 05/ Lang, S., Wittum, G.: Large-scale density-driven flow simulations using parallel unstructured Grid adaptation and local multigrid methods. Concurrency Computat.: Pract. Exper. 17, 1415-1440, 2005.

/LIU 72/ Liu, I.-S.: Method of Lagrange Multipliers for exploitation of the entropy principle, Archive Rational Mech. Anal., 46, 131-148, 1972.

/LIZ 06/ Li, Z.: The immersed interface method: Numerical solutions of PDEs involving interfaces and irregular domains. Cambridge University Press, 2006.

/LOY 02/ Loy, M., Eckstein, R., Wood, D., Elliot, J., Cole, B.: Java Swing, O'Reilly, 2002.

/LUB 09/ Lu, B., Wheeler, M. F.: Iterative coupling reservoir simulation on high performance computers. Petroleum Science, 6, 43-50, 2009.

/MAN 99/ Mandel, J., Brezina, M., Vanek, P.: Energy Optimization of Algebraic Multigrid Bases. Computing, 62, 205-228, 1999.

/MAR 81/ Marle, C. M.: Multiphase Flow in Porous Media, EditionsTechnip, 217-236, 1981.

/MAR 06/ Martinez-Landa, L., Carrera. J.: A methodology to interpret crosshole tests in a granite block. Journal of Hydrogeology 325, 222-240, 2006.

/MUR 79/ Murphy, H. D.: Convective Instabilities in Vertical Fractures and Faults. Journal of Geophysical Research 84, 6121-6130, 1979.

/MUR 99/ Murdoch, A. I., Soliman, A.: On the slip-boundary condition for liquid flow over planar porous boundaries. Proc. R. Soc. Lond. A 455, 1315-1340, 1999.

/MUR 05/ Murdoch, A. I.: Some Fundamental Aspects of Surface Modelling. Journal of Elasticity 80, 33-52, 2005.

/MUS 09/ Musuuza, J.L., Attinger, S., Radu, F.A.: An extended stability criterion for density-driven flows in homogeneous porous media, Advances in Water Resources 32, 796-808, 2009.

/MUS 10/ Musuuza, J.L, Radu, F.A., Attinger, S.: The effect of dispersion on the stability of density-driven flows in saturated homogeneous porous media, 417-432, 2010.

/NAE 08/ Nägel, A., Falgout, R. D., Wittum, G.: Filtering algebraic multigrid and adaptive strategies. Computing and Visualization in Science, 11(3):159–167, 2008.

/NEU 05/ Neumann, S.: Trends, prospects and challenges in quantifying flow and transport through fractured rocks. Hydrogeology J. 13, 124-147, 2005.

/NOK 09/ Nokia, Qt Development Frameworks, Qt 4.6, http://qt.nokia.com, 2009

/ODE 98/ Oden, T.; Baumann, C. E.; Babuska, I.: A discontinous hp finite element method for diffusion problems. J. Comp. Phys., 146, 491-519, 1998.

/OGA 61/ Ogata, A., Banks, R.B.: A solution of the differential equation of longitudinal dispersion in porous media. U.S. Geol. Surv. Prof. Paper 11-A, 1961.

/OLD 95/ Oldenburg, C.M., Pruess, K.: Dispersive transport dynamics in a strongly coupled groundwater-brine flow system, Water Resources Reaseach 31, 289-302, 1995.

/OLD 98/ Oldenburg, C.M., Pruess, K.: Layered thermohaline convection in hypersaline geothermal systems. Transport in porous media, 33, 29 – 63, 1998.

/OLD 99/ Oldenburg, C.M., Pruess, K.: Plume separation by transient thermohaline convection in porous media. Geophysical research letters, 26, 2997-3000, 1999.

/OLD 00/ Oldenburg, C.M., Pruess, K.: Thermohaline convective mixing at a brine interface. Proceedings of the 25th Workshop on Geothermal Reservoir Engineering. Stanford University, Stanford, California, USA, January 24th-26th, 2000.

/ORA 96/ Oracle, formally Sun Microsystems, http://www.oracle.com/us/technologies/java/index.html, 1996.

/ORA 10/ Oracle, formally Sun Microsystems, JavaFX 1.3, http://javafx.com, 2010.

/ORA 10a/ Oracle, formally Sun Microsystems, Netbeans IDE 6.9.1, http://www.netbeans.com, 2010

/ORT 66/ Ortega, J., Rockoff, M.: Nonlinear difference equations and Gauss Seidel type iterative methods. J. SIAM Numer. Anal., 3, 497-513, 1966.

/OSW 98/ Oswald, S.: Dichteströmungen in porösen Medien: Experimente und Modellierung. Dissertation, ETH Zürich, Nr. 12812, 1998.

/PAR 88/ Parvathy, C.P., Prabhamani, R .P.: Thermohaline instability with thermal diffusion and horizontal gradients. Applied Scientific Research, 45, 163 – 178, 1988.

/PER 66/ Perrine, R.L., Gay, G. M., Unstable miscible flow in heterogeneous systems, Society of Petroleum Engineering Journal 6, 228, 1966.

/PET 86/ Petit, C.J., Hwang, M .-H., Lin, J.-L.: The Soret effect in dilute aqueous alkaline Earth and nickel chloride solutions at 25 °C. Int. J. of Thermophysics, 7(3), 687-697, 1986.

/RAU 06/ Rauch, J.: Diffusion and thermal diffusion in polymer solutions. PhD Thesis, Universität Bayreuth, 2006.

/REM 01/ Rempel, A.W., Wettlaufer, J.-S., Worster, M.-G.: Interfacial premelting and thermo-molecular force: Thermodynamic buoyancy. Phys. Rev. Letters, 87(8), S. 088501-1, 2001.

/ROW 80/ Rowley, R.L., Horne, F .H.: The Dufour Effect. III. Direct experimental determination of the heat of transport of carbon tetrachloride-cyclohexane liquid mixtures, J. Chem. Phys., 72(1), 131-139, 1980.

/RUG 07/ Ruggeri, T., Simic, S.: On the hyperbolic system of a mixture of Eulerian Fluids: a comparison between single- and multi-temperature models, Math. Mech. Appl. Sci., 30, 335-849, 2007.

/RUG 87/ Ruge, J. W., Stüben, K.: Algebraic multigrid (AMG) in McCormick, S. F. (Ed.): Multigrid Methods, Frontiers in Applied Mathematics 5, SIAM, Philadelphia, 1986.

/RUM 92/ Rumpf, M., Wierse, A.: GRAPE, Eine Interaktive Umgebung für Visualisierung und Numerik. Informatik, Forschung und Entwicklung, 7,145-151, 1992.

/SAN 88/ Sanyal, S.K., Mukherjee, A.K.: Heat of transport and heat capacity of transport of some aqueous electrolytes, J. Chem., 66, 435-438, 1988.

/SCH 74/ Scheidegger, A. E.: The Physics of Flow through Porous Media, 3rd Edition. University of Toronto Press, Toronto, 1974.

/SCH 90/ Schincariol, R.A., Schwartz, F.W.: An experimental investigation of variable-density flow and mixing in homogeneous and heterogeneous media, Water Resources Research 26(10), 2317-2329, 1990.

/SCH 97/ Schincariol, R.A., Schwartz, F.W., Mendoza, C.A.: Instabilities in variable-density flow and mixing in homogeneous and heterogeneous media, Water Resources Research 33(1), 31-41, 1997.

/SCH 03/ Scheichl, R., Masson, R., Wendebourg, J. Decoupling and Block Preconditioning for Sedimentary Basin Simulations. Computational Geosciences, 7:295-318, 2003.

/SCH 04/ Schneider, A.; Birthler, H.: Modellrechnungen zur großräumigen dichteabhängigen Grundwasserbewegung. FKZ-02 C 0628 4. Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-191, Braunschweig 2004.

/SER 03/ Serco Assurance: NAMMU Release 7.2 Technical Summary Document. Serco Reference SA/ENV-0626, Oxfordshire, UK, 2003.

/SET 99/ Sethian, J.: Level Set Methods and Fast Marching Methods. Cambridge University Press, 1999.

/SET 77/ Settari, A., Price, H.S., Dupont, T.: Development and application of variational methods for simulation of miscible displacements in porous media, Society of Petroleum Engineering Journal 17, 228-246, 1977.

/SHA 09/ Sharp, J. M. J., Shi, M.: Heterogeneity effects on possible salinity-driven free convection in low-permeability strata. Geofluids 34, 263–274, 2009.

/SHI 98/ Shikaze, S. G., Sudicky, E. A., Schwartz, F. W.: Density-dependent solute transport in discretely-fractured geologic media: is prediction possible? Journal of Contaminant Hydrogeology 34, 273-291, 1998.

/SIM 04/ Simpson, M., Clement, T.: Improving the worthiness of the Henry problem as a benchmark for density-dependent groundwater flow models. Water Resources Research 40, 2004.

/SOR 01/ Sorek, S., Borisov, V., Yakirevich, A., 2001. A two-dimensional Areal Model for Density Dependent Flow Regime. Transport in Porous Media 43, 87–105, 2001.

/STI 11/ Stichel, S., Logashenko, D., Grillo, A., Reiter, S., Lampe, M., Wittum, G.: Numerical Methods for Flow in Fractured Porous Media. Accepted for publication in J. M. P. Q. Delgado (ed.), Heat and Mass Transfer in Porous Media, Advanced Structured Materials 13, Springer-Verlag, 2011.

/STU 01/ Stüben, K.: An Introduction to Algebraic Multigrid, chapter Appendix A, 413-532. Academic Press, 2001.

/STU 07/ Stüben, K., Clees, T., Klie, H., Lu, B., Wheeler, M.F.: Algebraic Multigrid Methods (AMG) for the Efficient Solution of Fully Implicit Formulations in Reservoir Simulation, Paper SPE 105832 presented at the 2007 SPE Reservoir Simulation Symposium, Houston, TX, Feb. 28-30, 2007.

/SUG 75/ Sugisaki, M.: Soret coefficients and heat of transport of polyvalent electrolytes in an aqueous solution, Bulletin of the chemical society of Japan, 40(10), 2751 – 2754, 1975.

/TAN 81/ Tang, D.H., Frind, E.O., Sudicky, E.A.: Contaminant Transport in Fractured Porous Media: Analytical Solution for a Single Fracture. Water Resources Research, Vol. 17, No. 3, 555-564, 1981.

/THE 10/ Therrien, R., McLaren, R.G., Sudicky, E.A., Panday, S.M.: HydroGeoSphere - A Three-dimensional Numerical Model Describing Fully-integrated Subsurface and Surface Flow and Solute Transport. User´s Manual for HydroGeoSphere, Université Laval, 2010 (Draft).

/THI 98/ Thiele, K.: An error estimation for density driven flow problems in porous media. PhD thesis, Institut für Angewandte Mathematik, Universität Erlangen-Nürnberg, 1998.

/TUR 82/ Turcotte, D.L., Schubert, G.: Geodynamics.John Wiley & Sons, 1982.

/TYR 56/ Tyrrel, H.J.V.: The calculation of diffusion coefficients and Soret coefficients from optical measurements on pure Soret effect cells, Trans. Faraday Society, 52, S. 940 – 948, 1956.

/VAN 01/ Vanek, P., Brezina, M., Mandel, J.: Convergence of algebraic multigrid based on smoothed aggregation. Numerische Mathematik, 88, 559--579, 2001.

/VAN 96/ Vanek, P., Mandel, J., Brezina, M.: Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. Computing, 56, 179-196, 1996.

/VOG 10/ Vogel, A.; Xu, J.; Wittum, G.: A generalization of the vertex-centered finite volume scheme to arbitrary high order. Computing and Visualization in Science, 13, 221-228, 2010.

/VOS 87/ Voss, C.I., Souza, W.R.: Variable density flow and solute transport simulation of regional aquifers containing a narrow freshwater-saltwater transition zone. Water Resour Res, 26, 2097-2106, 1987.

/WAG 00/ Wagner, C.: On the algebraic construction of multilevel transfer operators. Computing, 65:73-95, 2000.

/WAL 85/ Wallis, J. R., Kendall, R. P., Little, T. E.: Constrained Residual Acceleration of Conjugate Residual Methods. SPE 13536, 415--426, 1985.

/WAL 05/ Wallner, M., Mrugalla, S.; Hammer, J.; Brewitz, W.; Fahrenholz, C.; Fein, E.; Filbert, W.; Haverkamp, B.; Jobmann, M.; Krone, J., Lerch, C., Ward, P., Weiß, E., Ziegenhagen, J., Gupalo, T., Kamnev, E., Konovalov, V., Lopatin, V., Milovidov, V., Prokopova, O.: Anforderungen an die Standorterkundung für HAW-Endlager im Hartgestein (ASTER), Final report, FKZ 02E9612 and FKZ 02E9622, 386 p, 2005.

/WAN 09/ Wang, W., Kosakowski, G., Kolditz, O.: A parallel finite element scheme for thermo-hydro-mechanical (THM) coupled problems in porous media. Computers & Geosciences 35, 1631–1641, 2009.

/WAR 84/ Ward, D.S., Reeves, M., Duda, L.E.: Verification and Field Comparison of the Sandia Waste-Isolation Flow and Transport Model (SWIFT), NUREG/CR-3316, SAND83-1154, Sandia National Laboratories, Albuquerque, New Mexico, 1984.

/WAS 10/ FEFLOW 6 User Manual. www.feflow.info DHI-WASY, Berlin, 2010.

/WEL 91/ Welty, C. and Gelhar, W.L.: Stochastic analysis of the effects of fluid density and viscosity variability onmacrodispersion in heterogeneous porous media, Water Resources Research 27, 2061-2075, 1991.

/WHI 99/ Whitaker, S.: The Method of Volume Averaging - Theory and Applications of Transport in Porous Media, 1st Edition. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999.

/WIE 99/ Wieners, C.: Parallel linear algebra and the application to multigrid methods. Notes on Numerical Fluid Mechanics, 56, 1999.

/WIL 03/ Wilmanski, K.: On Thermodynamics of Nonlinear Poroelastic Materials, Journal of Elasticity, 71, S. 247 – 261, 2003.

/WIT 89/ Wittum, G.: Multi-grid methods for Stokes and Navier-Stokes equations. Transforming smoothers algorithms and numerical results. Numerische Mathematik. 54, 543-563, 1989.

/WIT 92/ Wittum, G.: Filternde Zerlegungen : Schnelle Löser für große Gleichungssysteme. (Habilitationsschrift) Skripten zur Numerik Bd 1, Teubner, 1992.

/WOO 62/ Wooding, R.A.: Stability of an interface between miscible fluids in a porous medium, Zeitschrift für angewandte Mathematik und Physik 13, 255-266, 1962.

/WOR 99/ WorldWide Web Consortium, MathML, http://www.w3.org/1999/07/REC-MathML-19990707, 1999.

/YAN 00/ Yang, J., Edwards, R.N.: Predicted groundwater circulation in fractured and unfractured anisotropic porous media driven by nuclear fuel waste heat generation. Canadian Journal of Earth Science 37, 1301-1308, 2000.

/ZIE 91/ Zielke, W., Helmig, R., Kröhn, K.-P., Shao, H., Wollrath, J.: Discrete Modelling of Transport Processes in Fractured Porous Rock, in “International Congress on Rock Mechanics. Wittke (ed.), 57-60, International Society for Rock Mechanics, Aachen, A.A.Balkema Publishers, Rotterdam, 1991.

Table of figures

Fig. 3.1 The Schincariol results 26

Fig. 3.2 Stable or unstable? 27

Fig. 3.3 Homogeneous Medium: Density Effects 27

Fig. 3.4 Homogeneous Medium: Viscosity Effects 28

Fig. 3.5 Fitting the mixing zone width 29

Fig. 3.6 The fitted longitudinal dispersivities 30

Fig. 3.7 Density effects: onset of convection 31

Fig. 3.8 Density effects: onset of unstable convection 31

Fig. 3.9 Effects of longitudinal dispersivity 32

Fig. 3.10 The transverse dispersivity effects 33

Fig. 3.11 The critical correlation length 34

Fig. 3.12 The stabilised range of densities 35

Fig. 3.13 Stabilising effect of variance 36

Fig. 3.14 Effect of longitudinal dispersivity 37

Fig. 3.15 Effect of the medium anisotropy 37

Fig. 3.16 Passive tracers 38

Fig. 3.17 Favourable density contrasts 39

Fig. 3.18 Unstable density contrasts 40

Fig. 3.19 Effect of the correlation length 41

Fig. 3.20 Effect of the heterogeneity variance 41

Fig. 3.21 Effect of medium anisotropy 42

Fig. 3.22 Asymmetry caused by perturbing inflow region 43

Fig. 3.23 Summary of homogeneous and heterogeneous media 44

Fig. 4.1 Scheme of an Elder problem 67

Fig. 4.2 Thermohaline Elder problem 68

Fig. 4.3 Initial configuration of negative buoyancy 70
Fig. 4.4 Evolution of the parcel after 10 yrs..... 70
Fig. 4.5 Evolution of the parcel after 20 yrs..... 70
Fig. 4.6 Initial configuration of the parcel for positive buoyancy..... 71
Fig. 4.7 Evolution of the parcel after 10 yrs..... 71
Fig. 4.8 Evolution of the parcel after 20 yrs..... 71
Fig. 4.9 Initial configuration of the mixing problem..... 72
Fig. 4.10 Distribution of mass fraction and temperature after t = 150 yrs 72
Fig. 4.11 Mass fraction vs. depth 73
Fig. 4.12 Soret cell 74
Fig. 4.13 Numerically computed profile of mass fraction in a Soret cell..... 75
Fig. 4.14 Analytically computed profile of mass fraction in a Soret cell..... 75
Fig. 4.15 Computation domain and parcel “immersed” in it..... 81
Fig. 4.16 Exemplified two-dimensional grid and dual mesh 84
Fig. 4.17 Evolution of the parcel in the case of negative buoyancy 88
Fig. 4.18 Evolution of the parcel in the case of positive buoyancy..... 89
Fig. 4.19 Isosurfaces of brine mass fraction in case of negative buoyancy 90
Fig. 4.20 Isosurfaces of brine mass fraction in case of positive buoyancy..... 91
Fig. 5.1 Scheme of a planar *d*-dimensional fracture in the averaging process 103
Fig. 5.2 Enumeration of the grid nodes in a piece of a grid with *N*=100..... 119
Fig. 5.3 Geometry and boundary conditions for the modified Henry problem featuring a fracture 125
Fig. 5.4 Isolines of the mass fraction and velocity directions of the $\boldsymbol{d}$- and $(\boldsymbol{d}-\mathbf{1})$-dimensional simulation of Henry's problem with a fracture 127
Fig. 5.5 Comparisons of full- and low-dimensional simulations at *x*=1.5 m..... 128
Fig. 5.6 Isolines of the mass fraction and velocity directions from the simulation with an intersection of fractures 128
Fig. 5.7 Location of the fractures, boundary grid and decomposition into the boundary surfaces in the 3d test 129

Fig. 5.8 Isolines for c=0.1 and c=0.3 at time 100 min. in the 3d test 130

Fig. 6.1 Illustration of the notation 132

Fig. 6.2 Several contour lines of the initial level set function, zero contour line that defines the initial position of horizontal groundwater table 136

Fig. 6.3 Groundwater table, pressure, and groundwater flow 139

Fig. 6.4 Signed distance function, its gradient and the extrapolated speed 143

Fig. 6.5 Position of groundwater table after one advection step, pressure and groundwater velocity field for the new position 145

Fig. 6.6 Signed distance function, normal directions and extrapolated speed for the new position 146

Fig. 6.7 Domain Ω inside of D,two vector fields, and the corresponding signed distance function Φ 149

Fig. 6.8 Signed distance function for the implicitly given interface in the shape of a square 150

Fig. 6.9 Resulting signed distance function for the implicitly given interface of some complex shape 151

Fig. 6.10 Circular interface Γ, related signed distance function Φ, and constant extrapolation along characteristics generated by $\nabla\Phi$ 153

Fig. 8.1 A finite-element mesh with the degrees of freedoms for linear Ansatz functions and quadratic Ansatz functions 158

Fig. 8.2 Subdivision of triangles 159

Fig. 8.3 Difference to a reference solution of the Henry problem for the first three orders of Ansatz spaces measures in H^1-norm 161

Fig. 8.4 Visualisation of the operator Ai and its local support $\boldsymbol{Li}$ for the model problem 177

Fig. 8.5 Distribution of the eigenvalues λk of the generalized eigenvalue problem (8.47) for model problem (8.49) 179

Fig. 8.6 Solution at one particular point in time 180

Fig. 8.7 Smooth Error after 3 SGS relaxation sweeps after an initialisation with a random vector $(\mathrm{r}p, 0)T$ 181

Fig. 8.8 Smooth Error after 3 SGS relaxation sweeps after an initialization with a random vector $(0, \mathrm{r}\omega)T$ 181

Fig. 8.9 Coarse grid and different hydro-geological areas and boundary conditions for the layered aquifer 187

Fig. 8.10 Horizontal interfaces IAB for the vertices of a distributed grid on three processes $P0, P1, P2$ 193

Fig. 8.11 Exemplary distribution of a 1d grid onto 2 processes..... 194

Fig. 8.12 1d Algebra hierarchy 195

Fig. 9.1 ProMesh: A cube with two volume-subsets, divided by a fracture 200

Fig. 9.2 Edge operations: Edge swap, edge split and edge collapse..... 204

Fig. 9.3 The optimization algorithm applied to a cube with an internal fracture ... 204

Fig. 9.4 Cuts of a cubic fractured geometry consisting of tetrahedrons 205

Fig. 9.5 Topological scheme of the degenerated elements and additional vertices in the fracture 207

Fig. 9.6 VRL Component Types 210

Fig. 9.7 Visualization of a simple Java object 210

Fig. 9.8 Code used to create the visualization shown in Fig. 9.7 211

Fig. 9.9 Compact implementation of a Swing application that is able to add two numbers..... 211

Fig. 9.10 Illustration of parameter annotations 213

Fig. 9.11 Visualization of a simple Java object using parameter annotations 213

Fig. 9.12 Type representations for java.lang.Integer and java.awt.Color 214

Fig. 9.13 Circle Class 214

Fig. 9.14 Type representation for the Circle class..... 215

Fig. 9.15 Class that uses the Circle class in its public interface, listing..... 216

Fig. 9.16 Class that uses the `Circle` class in its public interface 216

Fig. 9.17 Data dependency between two objects 217

Fig. 9.18 Custom sequence of method calls via codeblocks..... 217

Fig. 9.19 Parameter groups (selection) 219

Fig. 9.20 Parameter groups (result) 220
Fig. 9.21 Function plotter 220
Fig. 9.22 Reduced interface for the function plotter 221
Fig. 9.23 BasicMath Sample Session 222
Fig. 9.24 UG input with three BasicMath elements 223
Fig. 9.25 SolutionPlotter component visualizing a calculated geometry 226
Fig. 9.27 Interactive selection of physical domain 228
Fig. 9.28 Selection of physical parameters 229
Fig. 9.29 d³f sample application (a) 230
Fig. 9.30 d³f sample application (b) 231
Fig. 10.1 Salt concentration at three time steps of the Elder problem with a diagonal fracture 234
Fig. 10.2 Topological structure of two elements in a fracture. 235
Fig. 10.3 Geometrical representation of the two elements from Fig. 10.2 235
Fig. 10.4 Fracture element with normal pointing outwards 236
Fig. 10.5 The outer vertices are displaced in normal direction 236
Fig. 10.6 A vector field in a fracture 237
Fig. 10.7 A scalar field in a 2D fracture in three dimensions 237
Fig. 10.8 Cross section through a two dimensional fracture in 3D 238
Fig. 10.9 Salt concentration on the level set of an artificial level set function 239
Fig. 10.10 Salt concentration on parallel clipping planes 239
Fig. 10.11 Sources and sinks in a model geometry 240
Fig. 10.12 Several probe input methods 241
Fig. 10.13 Probe output and grid sampling 242
Fig. 10.14 Plot from probing at three different points 242
Fig. 10.15 Stochastic data mapped to a cross section of the model geometry 245
Fig. 10.16 Visualization of sub domains with manually selected colouring 245

Fig. 11.1 Model geometry, initial and boundary conditions for the two-dimensional conductive heat transfer problem 256

Fig. 11.2 Analytical solution (OBs) and the referring numerical results from d³f for the test of heat conduction 257

Fig. 11.3 Geometry, initial and boundary conditions for the Ogata-Banks model 258

Fig. 11.4 Analytical solution and referring numerical results from d³f..... 259

Fig. 11.5 Model geometry and boundary conditions for heat transfer in anisotropic porous media 259

Fig. 11.6 Temperature profiles in the vertical symmetry axis of the model 261

Fig. 11.7 Comparison of the temperature fields 262

Fig. 11.8 Flow fields calculated with d^3f 263

Fig. 11.9 Physical system underlying the analytical solutions of /TAN 81/..... 264

Fig. 11.10 Geometry and boundary conditions for the numerical model..... 271

Fig. 11.11 Concentration in the fracture: analytical and numerical solution 271

Fig. 11.12 Concentration in the matrix after 1, 3, and 5 years 272

Fig. 11.13 Schematic cross section through the investigation area 274

Fig. 11.14 Model geometry based on the schematic cross-section with identifiers of the units and fractures 275

Fig. 11.15 Boundary conditions for the groundwater flow simulations with d^3f 276

Fig. 11.16 Pressure distribution in the model area..... 277

Fig. 11.17 Groundwater flow velocity in the matrix and in the fractures on a logarithmic scale 278

Fig. 11.18 Groundwater flow directions and velocities in the matrix and the fractures calculated with d^3f. 279

Fig. 11.19 Groundwater flow directions and velocities in the factures calculated with d^3f..... 281

Fig. 11.20 Model geometry and temperature boundary conditions for the groundwater flow simulations with d^3f 284

Fig. 11.21 Temperature isolines after 187 000 days calculated with d^3f and FEFLOW 285

Fig. 11.22 Boundary conditions for the transport model in d^3f 286

Fig. 11.23 Groundwater flow directions and velocities in d^3f and FEFLOW 287

Fig. 11.24 Groundwater flow velocities in the fractures calculated with d^3f and FEFLOW 288

Fig. 11.25 Concentration isolines after $4.6 \cdot 10^6$ days calculated with d^3f and r^3t 289

Fig. 11.26 Concentration isolines after $3.5 \cdot 10^7$ days calculated with d^3f and FEFLOW 290

Fig. 11.27 Model and boundary conditions 292

Fig. 11.28 Computational grid, created with ProMesh 292

Fig. 11.29 Velocity field in a vertical cross section from the ROCKFLOW-model 293

Fig. 11.30 Velocity field in a vertical cross section from the d^3f-model 293

Fig. 11.31 Vertical concentration profiles from the ROCKFLOW-model at steady state; 294

Fig. 11.32 Vertical concentration profiles from the d^3f-model at steady state condition 295

Fig. 11.33 Vertical concentration profiles from the d^3f-model at steady state condition for a refinement level of 3 295

Fig. 11.35 Fracture geometry; a) original data, b) modified data 299

Fig. 11.37 Extraction strategy for the boundary conditions 302

Fig. 11.38 Six planes showing dynamic pressure 302

Fig. 11.41 Comparison of extracted data and results of the analytical function 305

Fig. 11.43 First attempt on the coarse grid 307

Fig. 11.44 Ultimately used coarse grid for the model 307

Fig. 11.45 Isoplanes of the dynamic pressure at the TASD-tunnel 308

Fig. 11.49 Schematic diagram of groundwater flow within a dam 313

Fig. 11.50 Hydrogeological model for the dam with boundary conditions 313

Fig. 11.51 Groundwater surface, velocity vectors and isobars at steady state 314

List of tables

Tab. 2.1 Modelling with d3f and r3t 6

Tab. 3.1 The Effect of Density 27

Tab. 3.2 Homogeneous Medium: Viscosity Effects 28

Tab. 3.3 Dispersion extension: density 300

Tab. 3.4 Dispersion extension: longitudinal dispersivity 32

Tab. 3.5 Effects of transverse dispersivity 322

Tab. 3.6 The effect of the correlation length 344

Tab. 3.7 The stabilising effect of heterogeneities 35

Tab. 3.8 Effect of variance 36

Tab. 3.9 Effect of longitudinal dispersivity 366

Tab. 4.1 Thermodynamics quantities and related fluxes to substitute in eq. (4.1) 51

Tab. 4.2 Thermodynamics quantities and related fluxes to substitute in eq. (4.2) 52

Tab. 4.3 Parameters used for the numerical simulations 76

Tab. 4.4 Initial and boundary values of the variables featuring in the model 866

Tab. 5.1 Parameters used for the computations 126

Tab. 8.1 Measurements of the approximation rate for different orders of Ansatz spaces used to discretize the density driven flow equations 161

Tab. 8.2 Convergence results for the salt pool problem. 186

Tab. 8.3 Convergence results for the aquifer problem 188

Tab. 8.4 Convergence results for the Norderney problem 188

Tab. 8.5 Strong Scaling of the laplace problem 197

Tab. 8.6 Weak Scaling of the laplace problem 197

Tab. 11.1 Parameter values for the Ogata-Banks model after /THE 10/ 254

Tab. 11.2 Parameter values for the Ogata-Banks model for purely convective heat transfer, modified after /THE 10/ 256

Tab. 11.3 Parameter values for the Yang-Edwards model after /THE 10/ 260

Tab. 11.4 Parameters for the matrix diffusion model according to /TAN 81/ 270

Tab. 11.5 Hydrogeological parameters for the four geological units 275

Tab. 11.6 Hydrogeological parameters for the five fractures 276

Tab. 11.7 Velocity ranges for the different units from d3f and FEFLOW simulations 277

Tab. 11.8 Hydrogeological parameters for the matrix and the fractures in the groundwater simulation concerning heat transport 282

Tab. 11.9 Hydrogeological parameters for the five fractures in the groundwater simulation concerning heat transport 283

Tab. 11.10 Hydraulic properties of the hydraulic features 300

Tab. 11.11 Constants for equation (11.48) 304

A Appendix A: Notation

The most general notations used in this report are given below. Where different authors used different symbols for the same quantity the referring chapter is also cited.

- q - flow velocity vector [m s^{-1}] (chapter 4, 5, 6, 8)
- $\mathbf{u}$ - flow velocity vector [m s^{-1}] (chapter 3)
- v - flow velocity vector [m s^{-1}] (chapter 4, 11)
- V - flow velocity vector [m s^{-1}] (chapter 6)
- $\mathbf{K}$ - permeability tensor [m^2] (chapter 4, 5, 6, 8, 11)
- $\mathbf{k}$ - permeability tensor [m^2] (chapter 3,11)
- Λ - thermal conductivity [W m^{-1} K^{-1}] (chapter 4)
- λ - thermal conductivity [W m^{-1} K^{-1}] (chapter 11)
- C - specific heat capacity [J kg^{-1} K^{-1}] (chapter 4)
- c - specific heat capacity [J kg^{-1} K^{-1})] (chapter 11)
- θ - temperature [K] (chapter 4)
- T - temperature [K] (chapter 11)
- ω - solute mass fraction [-] (chapter 3, 4, 5, 8)
- χ - solute mass fraction [-] (chapter 8)
- c - volumetric solute concentration [kg m^{-3}] (chapter 5)
- c - specific solute concentration [kg m^{-3}] (chapter 11)
- ϕ - porosity [-]
- ρ - density [kg m^{-3}]
- μ - viscosity [Pa s]
- τ - tortuosity [-]
- p - hydraulic pressure [Pa]
- $\mathbf{g}$ - gravitational acceleration [ms^{-2}]
- $\mathbf{D}$ - dispersion tensor [m^2 s^{-1}]
- D_m - molecular diffusion coefficient [m^2 s^{-1}]
- $\mathbf{I}$ - identity matrix [-]
- α_l - longitudinal dispersion length [m] (chapter 4, 5, 6, 8, 11)
- $\alpha_{\|}$ - longitudinal dispersion length [m] (chapter 3)
- α_t - transverse dispersion length [m] (chapter 4, 5, 6, 8, 11)

$\alpha_{\perp}$ - transverse dispersion length [m] (chapter 3)

$\boldsymbol{J}_T$ - heat flux vector

$\boldsymbol{J}_d$ - diffusive flux

B Appendix B: Publications

The following publications and master as well as PhD theses have been written in the framework of the project:

GRS-report

Kröhn, K.-P.: State Variables for Modelling Thermohaline Flow in Rocks. FKZ 02 E 10336 (BMWi), Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH, GRS-268, Köln, 2010.

Papers

Frolkovič, P., Logashenko, D., Wittum, G.: Flux-based level set method for two-phase flows. In R. Eymard and J.-M. Herard, editors, Finite Volumes for Complex Applications, pages 425–422. ISTE and Wiley, 2008.

Frolkovič, P., Lampe, M., Wittum, G.: Numerical simulation of contaminant transport in groundwater using the software tool r³t, submitted 2011.

Frolkovič, P., Logashenko, D., Wehner, C., Wittum, G.: Flux-based level set method and applications on unstructured grids, submitted 2011.

Grillo, A., Lampe, M., Wittum, G.: Modelling and Simulation of Temperature-Density-Driven Flow and Thermodiffusion in Porous Media. Journal of Porous Media, v14.i8.20, pages 671-690, 2011.

Grillo, A., Logashenko, D., Stichel, S., Wittum, G.: Simulation of Density-Driven Flow in Fractured Porous Media. Advances in Water Resources, Volume 33, Issue 12, Pages 1494-1507, 2010.

Grillo, A., Lampe, M., Wittum, G.: Three-dimensional simulation of the thermohaline-driven buoyancy of a brine parcel. Comput Vis Sci, Comput Vis Sci Volume 13, Number 6, 287-297, 2010.

Grillo, A., Lampe, M., Logashenko, D., Stichel, S., Wittum, G.: Simulation of salinity- and thermohaline-driven flow in fractured porous media. Journal Porous Media, to appear 2011.

Grillo, A., Federico, G., Wittum, G.: Growth, mass transfer, and remodeling in fiber-reinforced multi-constituent materials. International Journal of Non-Linear Mechanics, 2011 to appear.

Grillo, A., Logashenko, D., Wittum, G.: Study of a transport problem in a two-dimensional porous medium. Cosserrat +100, International Conference on the legacy of Théorie des Corps Déformables by Eugene and Francois Cosserat in the centenary of its publication, 15-17 July 2009.

Hoffer, M., Poliwoda, C., Wittum, G.: Visual Reflection Library. A Framework for Declarative GUI Programming on the JAVA Platform, submitted 2011.

Micunovic, M. V., Grillo, A., Muha, I., Wittum, G.: Two dimensional plastic waves in quasi rate independent viscoplastic materials. 7th Euromesh Solid Mechanics Conference, J. Ambrosio et.al. (eds.), Lisbon, Portugal, 7–11 September 2009.

Muha, I., Stichel, S., Attinger, S., Wittum, G.: Coarse graining on arbitrary grids. Multiscale Model. Simul. Volume 8, Issue 4, pp. 1368-1382, 2010.

Musuuza J., Radu F., Attinger, S.: The stability of density-driven flows in saturated heterogeneous porous media, Advances in Water Res. 34, 1464–1482, 2011

Musuuza J., Radu F., Attinger, S.: The effect of dispersion on the stability of density-driven flows in homogeneous porous media, Advances in Water Res., 34 (3): 417-432, 2011.

Musuuza, J. L., Attinger, S., Radu, F. A.: An extended stability criterion for density-driven flows in homogeneous porous media, Adv.Water Resour. 32 (6), 796-808, 2009.

Nägel A., Falgout R., Wittum G.: Filtering algebraic multigrid and adaptive strategies. Computing and Visualization in Science, 11(3):159–167, 2008.

Vogel, A., Wittum, G., Xu, J.: A generalization of the vertex-centered Finite Volume scheme to arbitrary order. Comput. Vis. Sci. Volume 13 Issue 5, June 2010.

Posters and presentations

Attinger, S.: Mixing processes in heterogeneous porous media without separated scales, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Berkels, B., Linkmann, G., Rumpf, M.: An SL(2) invariant shape median. Journal of Mathematical Imaging and Vision, 37(2):85-97, June 2010.

Frolkovic, P.: Some advances in numerical modelling of groundwater flow and contaminant transport, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Grillo, A., Lampe, M., Logashenko, D., Reiter, S., Stichel, S., Wittum, G.: An Overview of Models on Variable-Density Flow in Fractured Porous Media, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Hoffer, M.: A software for the automated mapping of program functionality to intuitive user interfaces in the context of technical simulation, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Kröhn, K.-P.: Qualifying a code for simulating fracture flow, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Lenz, M., Nemadjieu, S. F., Rumpf, M.: A convergent finite volume scheme for diffusion on evolving surfaces. SIAM Journal on Numerical Analysis, 49(1):15-37, 2011.

Musuuza, J. L: A Stability Criterion for Homogeneous Density-driven Flows. Poster EGU Konferenz, Wien, Apr. 2008.

Musuuza, J. L: A Stability Criterion for Density-driven Flows in Heterogeneous Media. Poster. NUPUS Konferenz, Stuttgart, Okt. 2009.

Musuuza, J. L: The Stability of Density-driven Flows in Heterogeneous Porous Media. Poster. DBG Jahrestagung, Bonn, Sept. 2009.

Musuuza, J. L: The Effect of Dispersion on Flow Stability. Poster Mai 2009 SIAM Konferenz, Leipzig. A Stability Criterion for Heterogeneous Density-driven Flows. Gastvortrag. EGU Konferenz, Wien, Apr. 2009.

Musuuza, J. L: The Stability of Density-driven Flows in Heterogeneous Media. Vortrag. EGU Konferenz, Wien, Mai 2010.

Musuuza, J. L.: The Application of Homogenization Theory to Study Instabilities in Saturated Heterogeneous Media. Gastvortrag. SIAM Konferenz, Long Beach, USA, März 2011.

Musuuza, J. L., Radu, F., Attinger, S.: The application of homogenization theory to predict the onset of thermal convection in thermohaline systems, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Reiter, S., Logashenko, D.: Topological Expansion of Low Dimensional Fractures in Porous Media, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Rumpf, M.: Variational methods in image matching and motion extraction. In M. Burger and S. Osher, editors, Level Set and PDE based Reconstruction Methods: Applications to Inverse Problems and Image Processing, Lecture Notes in Mathematics. to appear as CIME course notes, Springer, 2009.

Rupp, M.: On Algebraic Multigrid for Density Driven Flow in Porous Media, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Schneider A.: Developing a modelling tool for density-driven flow in complex hydrogeological structures. Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Stichel, S., Logashenko, D., Grillo, A., Wittum, G.: Numerical methods for flow in fractured porous media, Conference on Modelling Storage in Deep Layers (MSDL), Schwetzingen, 2011.

Master theses

Föhner, M.: Geometry Visualization & Modification Toolkit (GVMT), ein Werkzeug zur Ein-/Ausgabe und Modifikation von UG-Geometrien und UG-Gittern mit der 3D-Grafiksoftware Blender, 2008.

Hoffer, M.: Methoden der visuellen Programmierung, 2010.

Muha, I.: Coarse Graining auf beliebigen Gitterhierarchien, 2008.

Poliwoda, C.: Erstellung von Bedienoberflächen zur Steuerung von ausgewählten UG-Komponenten mit VRL, 2011.

Rupp, M.: Filternde Algebraische Mehrgitterverfahren zur Berechnung großer Eigenwertprobleme, 2008.

Stepniewski, M.: Approximation glatter Ränder und Volumengittergenerierung zur alternativen Erzeugung von 3D-Gitterhierarchien für Mehrgitterverfahren, 2011.

Stichel, S.: Numerisches Coarse-Graining in UG, 2008.

Vogel, A.: Ein Finite-Volumen-Verfahren höherer Ordnung, 2008.

Wehner, C.: Numerische Verfahren für Transportgleichungen unter Verwendung von Level-Set-Verfahren, 2008.

PhD theses

Feuchter, D.: Geometrie- und Gittererzeugung für anisotrope Schichtengebiete, Heidelberg, 2008.

Hauser, A.: Large Eddy Simulation auf uniform und adaptiv verfeinerten Gittern, Heidelberg 2009.

Musuuza, J. L.: Scaling Haline Flows in Heterogeneous Formations, PhD Thesis, Friedrich-Schiller-Universität Jena, 2010.

Nägel, A.: Schnelle Löser für große Gleichungssysteme mit Anwendungen, Heidelberg, 2010.

C Appendix C: Meetings

Regular project meeting	Date	Location	Organizer
1.	February 13, 2007	Heidelberg	Prof. Wittum
2.	October 22, 2007	Braunschweig	GRS
3.	April 21-22, 2008	Leipzig	Prof. Attinger
4.	October 27, 2008	Frankfurt	Prof. Wittum
5.	April 20-21, 2009	Maulbronn	Prof. Wittum
6.	October 19, 2009	Frankfurt	Prof. Wittum
7.	April 19-20, 2010	Leipzig	Prof. Attinger
8.	November 08-09, 2010	Braunschweig	GRS
9.	May 05, 2011	Frankfurt	Prof. Wittum

Gesellschaft für Anlagen- und Reaktorsicherheit (GRS) mbH

Schwertnergasse 1
50667 Köln
Telefon +49 221 2068-0
Telefax +49 221 2068-888

Forschungszentrum
85748 Garching b. München
Telefon +49 89 32004-0
Telefax +49 89 32004-300

Kurfürstendamm 200
10719 Berlin
Telefon +49 30 88589-0
Telefax +49 30 88589-111

Theodor-Heuss-Straße 4
38122 Braunschweig
Telefon +49 531 8012-0
Telefax +49 531 8012-200

www.grs.de

ISBN 978-3-939355-68-7