Christoph Lehmann, Olaf Kolditz,
Thomas Nagel

Models of Thermochemical Heat Storage

4th October 2017

Springer

About the Authors

Christoph Lehmann obtained his Bachelor's degree in Physics from the Martin Luther University Halle-Wittenberg in 2011 and his Master's in Physics from the University of Leipzig in 2013. He's been working as a software developer at pwp-systems developing tools for traffic data analysis from 2011-2015. After working on scientific visualisation at the Institute of Computer Science at Leipzig University, he joined the UFZ in 2015 to work on the numerical simulation of thermochemical energy storage (TCES), especially systems based on water adsorption to zeolite. His research interests are the numerical assessment of the energy efficiency of those storage systems and the examination of the sortion kinetics of novel composite adsorbents, in particular zeolite granules impregnated with hygroscopic salts for the enhancement of the heat storage density.

Olaf Kolditz is the head of the Department of Environmental Informatics at the Helmholtz Center for Environmental Research (UFZ). He holds a Chair in Applied Environmental System Analysis at the Technische Universität in Dresden. His research interests are related to environmental fluid mechanics, numerical methods and software engineering with applications in geotechnics, hydrology and energy storage. Prof. Kolditz is the lead scientist of the OpenGeoSys project[1], an open-source scientific software platform for the numerical simulation of thermo-hydro-mechanical-chemical processes in porous media, in use worldwide. He studied theoretical mechanics and applied mathematics at the University of Kharkov, got a PhD in natural sciences from the Academy of Science of the GDR (in 1990) and earned his habilitation in engineering sciences from Hannover University (in 1996), where he became group leader at the Institute of Fluid Mechanics. Until 2001 he was full professor for Geohydrology and Hydroinformatics at Tübingen University and director of the international Master course in Applied Environmental Geosciences. Since 2007 he is the speaker of the Helmholtz graduate school

[1] www.opengeosys.org

for environmental research HIGRADE. Prof. Kolditz is Editor-in-Chief of two international journals *Geothermal Energy* (open access) and *Environmental Earth Sciences* (ISI). Prof. Kolditz is a leading scientist in numerous Sino-German research networks.

Thomas Nagel is heading the groups "Computational Energy Systems"[2] at the Helmholtz Centre for Environmental Research—UFZ, as well as "Environmental Geotechnics: Multi-physics simulation for geotechnical system analysis"[3], a joint work group between the UFZ and the Federal Institute for Geosciences and Natural Resources (BGR). He holds an Adjunct Professorship at Trinity College Dublin, Ireland. During his master's studies of Mechanical Engineering/Applied Mechanics at Chemnitz University of Technology he worked on coupled problems in biomechanics and biomedical technology at Rush University Medical Center in Chicago and Eindhoven University of Technology, the Netherlands. In 2012 he was awarded his PhD from Trinity College Dublin for a dissertation in mechanobiology. Since then he has been working at the UFZ as part of a team of scientists developing the numerical simulation framework OpenGeoSys for coupled multiphysical problems. His particular foci are thermochemical and sensible heat storage in porous media as well as geotechnical applications in the context of energy supply, storage and waste management.

[2] http://www.ufz.de/comp-energy-sys

[3] http://www.ufz.de/environmental-geotechnics

Preface

Thermal energy storage is of high strategic relevance for the establishment of a sustainable energy system. The development of next-generation storage systems like thermochemical solutions is accompanied by major scientific and engineering challenges. Due to the complexity of the considered storage systems and the exceptional efforts for the development of storage materials as well as for the implementation of large-scale experiments, modelling and numerical simulation are of outstanding importance for the prediction of the operational behaviour and the optimization of thermochemical heat storage systems. The deployment of the necessary simulation tools is thus one of the key research topics in the area of thermal energy storage systems. These tools are intended to

- satisfy the need to analyse the relevance of various physical and chemical processes for the operation of the considered storage options,
- assess potentials, risks and impacts of sustainable storage options including economical and ecological aspects,
- simulate cyclic processes in sorption storage systems and storage facilities based on reversible chemical bonds within a unified framework,
- support the choice of highly efficient and stable materials for the constituents of the heat storage systems including the mitigation of ageing processes as well as capacity losses,
- aid the design of storage units in order to achieve a high level of efficiency, safety and sustainability.

This book presents an introduction to the computational modelling of the various coupled multi-physical processes relevant to thermochemical heat storage using the open-source software *OpenGeoSys* (*OGS*). The material is mainly based on work done in the framework of the NUMTHECHSTORE[4] project and a collaboration between the Research Programmes “Renewable Energies” and “Storage and Cross-Linked Infrastructures” by the Helmholtz Association

[4] http://www.ufz.de/index.php?en=37528

of German Research Centers. The software development is furthermore the result of a close cooperation within the OGS community (`www.opengeosys.org`). These voluntary contributions are highly acknowledged.

This book is intended primarily for graduate students and applied scientists working in the field of thermochemical heat storage in particular or non-isothermal reactive transport processes in packed beds in general, especially those interested in applying a freely available and customizable open-source simulation framework in their research. As such, this book will be a valuable help in the training of multi-physical modelling and provides an entry point to a modelling tool which is expandable and highly flexible.

This tutorial is the fifth volume in the Springer series *Computational Modeling of Energy Systems*[5] that presents applications of computational modelling in energy sciences. Within this series, a sequel of contributions describes the application of the simulation platform OGS to geotechnical applications in the energy sector based on work performed in close cooperation with the Federal Institute for Geosciences and Natural Resources (BGR)[6]:

- Computational Geotechnics: Storage of Energy Carriers[7] [Nag+17].
- Computational Geotechnics: Deep Geological Repositories, Nagel et al. (2018*).

It was preceded by three tutorials in the series covering the topic geothermal energy

- Geoenergy Modeling I. Geothermal Processes in Fractured Porous Media[8] [Böt+16],
- Geoenergy Modeling II. Shallow Geothermal Systems[9] [Sha+16],
- Geoenergy Modeling III. Enhanced Geothermal Systems[10] [Wat+17].

These contributions are related to a similar publication series in the field of environmental sciences, namely:

- Computational Hydrology I: Groundwater Flow Modeling[11] [Sac+15],
- Computational Hydrology II[12] [Sac+17],
- OGS Data Explorer, Rink et al. (2017*),

(*approximate publication time).

[5] `http://www.springer.com/series/15395`

[6] `http://www.ufz.de/environmental-geotechnics`

[7] `http://www.springer.com/de/book/9783319569604`

[8] `http://www.springer.com/de/book/9783319313337`

[9] `http://www.springer.com/de/book/9783319450551`

[10] `http://www.springer.com/de/book/9783319465791`

[11] `http://www.springer.com/de/book/9783319133348`

[12] `http://www.springer.com/de/book/9783319528083`

This book is exceptional in comparison to the above contributions in at least two ways: firstly, it is the only tutorial covering themes from the material sciences and process engineering rather than a geoscientific topic. Secondly, it is the first tutorial in this series using a completely new version of OpenGeoSys (OGS-6) featuring new models, new numerical methods, state-of-the-art software engineering and code development approaches, and many other innovations waiting to be discovered.

Few books are without errors, and this book is likely no exception. Should you discover errors that should be corrected, we would be grateful if you let us know and help improve this book.

Leipzig, September 2017

Christoph Lehmann
Olaf Kolditz
Thomas Nagel

Acknowledgements

We deeply acknowledge the OpenGeoSys community for their continuous support to the OpenGeoSys development activities. In connection with this book in particular we want to thank Lars Bilke and Tom Fischer for proof reading and Dmitri Naumov for providing background information about the software development. The authors would gratefully like to acknowledge the funding provided by the Helmholtz Association of German Research Centers via the Research Programmes "Renewable Energies" and "Storage and Cross-Linked Infrastructures" as well as the Helmholtz Initiating and Networking Fund through the NUMTHECHSTORE project. Finally, Christoph Lehmann wants to thank Luise Wagner for her patience and for taking over many duties in the last few weeks.

BGR Bundesanstalt für Geowissenschaften und Rohstoffe
GEOZENTRUM HANNOVER
HELMHOLTZ CENTRE FOR ENVIRONMENTAL RESEARCH – UFZ
C A U
TECHNISCHE UNIVERSITÄT DRESDEN
GFZ Helmholtz Centre POTSDAM
PAUL SCHERRER INSTITUT PSI
Leibniz Universität Hannover
BERGAKADEMIE TECHNISCHE UNIVERSITÄT FREIBERG
UNIVERSITÄT HOHENHEIM 1818
Universität Potsdam
THE UNIVERSITY OF EDINBURGH
SIGILL COLL TRIN SANC
CNSC CCSN
UNIVERSITY OF DELAWARE 1743
CHINA UNIVERSITY OF MINING AND TECHNOLOGY
1907 TONGJI UNIVERSITY
USGS science for a changing world
Pacific Northwest NATIONAL LABORATORY
SYDDANSK UNIVERSITET
EBERHARD KARLS UNIVERSITÄT TÜBINGEN
HTWK Leipzig
BERKELEY LAB
中国科学院 CHINESE ACADEMY OF SCIENCES
geo:tools
Schweizerische Eidgenossenschaft Confédération suisse Confederazione Svizzera Confederaziun svizra
Eidgenössisches Nuklearsicherheitsinspektorat ENSI Inspection fédérale de la sécurité nucléaire IFSN
Universität Stuttgart
KIGAM
nagra.
DBETEC DBE TECHNOLOGY GmbH
TUM TECHNISCHE UNIVERSITÄT MÜNCHEN
RWTH AACHEN UNIVERSITY
itwh

Contents

Chapter 1
Introduction

1.1 Thermal Energy Storage

Current energy policies aim at both an increase in energy efficiency and an increasing substitution of fossil fuels by renewable sources of energy. The latter, however, are largely intermittent in nature which increases the need for technologies to decouple energy supply and demand as well as for means to increase energy efficiency. Building climatisation, hot water supply and industrial process heat generation consume a significant share of the primary and final energy supply and offer considerable potentials for efficiency improve-

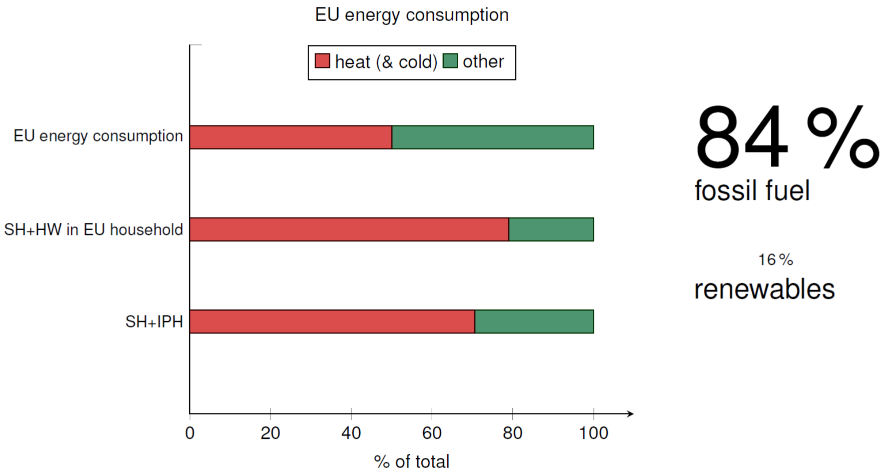


Figure 1.1 The share of heating and cooling in the EU's energy consumption (SH: space heating; HW: hot water; IPH: industrial process heat). Data source: `https://ec.europa.eu/energy/en/topics/energy-efficiency/heating-and-cooling`. Image adapted from [Nag+16b].

ments (Fig. 1.1). Intense research effort is therefore invested into numerous kinds of thermal energy storage systems in order to enhance our technological capabilities and provide a range of technological options in this regard.

Thermal energy[1] can basically be stored in three ways (in descending order of technological maturity):

- As sensible heat in (usually) solid or liquid matter[2] where the amount of energy stored is approximately proportional to the temperature change of the material. The heat capacity of the material determines how much heat is stored per Kelvin temperature change.
- In a phase change of a material by utilizing the latent heat of this phase change. Typically, the solid-liquid transition is used due to the high enthalpy of fusion and the low associated volume change (compared to liquid-gas transitions). Less common yet also possible are solid-solid phase transitions, but they typically have low enthalpies of phase change or insufficient rates. The enthalpy of fusion is absorbed or released under near-isothermal conditions making phase change materials very suitable for temperature buffering around a specified value, e.g. room climatisation around a comfort temperature.
- Via thermochemical reactions thermodynamically exploiting the reversible dissociation of a material compound AB into its constituents A and B. The enthalpy of reaction of suitable reaction systems is often comparably high. This option is at the centre of the present book.

Unique characteristics in terms of storage duration, storage and power density, as well as the possibility for heat transformation make thermochemical reactions an attractive technology for heat storage in a wide temperature range. Among the options considered for heat storage applications are various chemical reaction systems as well as ad- and absorption processes based on a wide range of materials, see Fig. 1.2.

Heat storage technologies are capable of providing multiple services to an energy system [Ker+11; Jeo+10; AR12; CLR13; Won+01]: they can provide storage capacities per se as well as enhance the energy efficiency of industrial processes, e.g., by utilizing waste heat. The possibility of heat transformation inherent to thermochemical heat storage can be exploited for heat upgrading, heat pumps, and air conditioning. Thermal storage can aid in load levelling and peak shaving, be linked to other energy storage technologies (such as in advanced adiabatic compressed air energy storage) as well as to power generation providing efficiency-enhanced co-generation/trigeneration capacities, and be installed to provide district heating and cooling. Thermochemical technologies have been employed for efficiently heating electric vehicles [GG95],

[1] The terms “thermal energy storage” and “heat” are used here without reference to a particular temperature level but of course imply both potential heating and cooling applications (i.e. include “cold storage”).

[2] For an application of OGS to sensible heat storage in solid-liquid mixtures, see [Mia+16].

Chemical reaction	**Absorption**	**Adsorption**
metallic hydrides carbonate systems hydroxide systems redox systems ammonia systems organic systems	Hygroscopic salts ○ $CaCl_2$ ○ $MgCl_2$ ○ LiCl ○ $MgSO_4$ ○ NaOH ○ LiBr ○ . . .	silica gels metalalumino-phosphates MOFs zeolites

Figure 1.2 Selection of materials suitable for different flavours of thermochemical heat storage.

for space applications [CM90; PMD92], and furthermore for increasing the efficiency of dish washers [Mai+11], and of a paint drying system [LN14].

Numerical simulation has proven to be a valuable tool for a time- and cost-effective design of complex engineered systems by allowing an early identification of crucial parameters which affect system performance and by assessing system behaviour at various scales for different, possibly transient operating conditions. An overview over modelling and simulation efforts undertaken in the case of thermochemical heat storage and heat transformation devices can be found in our review article [Nag+16a].

1.2 Heat Storage using Water Adsorption Processes

Interest in thermal energy storage (TES) using thermochemical reactions and sorption processes first peaked around the time of the oil crisis in the 1970s. Reference [WC76] was among the first to promote the use of chemical reactions without catalysts for storing solar heat in the context of solar power generation. Based on these criteria, a wide range of materials were screened and promising candidates selected for a deeper characterisation [WC76].

The possibility of using adsorption was proposed as a means for storing heat around the same time [CD77] based on the observation of unusually high storage capacities of a gravel bed thermal storage system which could be linked to adsorption phenomena. Silica gel was considered too expensive at the time motivating the authors to call for alternative materials, including salt-impregnated porous carrier materials [CD77].

[SLH79] suggested the use of other adsorbents such as alumina gel or activated alumina, activated carbon, silica gel and crystalline aluminosilicates, such as zeolites, for solar thermochemical heat storage. The water/zeolite working pair compared favourably to the other material pairs particularly in the context of space heating and drying applications [SLH79].

Today's climate, environmental and energy policies aimed at a decarbonisation of the energy sector by a transition to renewable energies[3] on the one hand and at an increased energy efficiency[4] on the other have renewed the interest in space-, cost-, and energy-efficient heat storage technologies.

For further reading, we would like to refer the reader to the following review articles published on various aspects of thermal energy storage taken directly from our review article [Nag+16b]: "[Kay99] discussed the viability of various TES technologies; [Won+01] gave a broad overview over material pairs putting special emphasis on chemical heat pumps for solar thermal cold production and the upgrading/storage of heat; Hasnain [Has98a; Has98b] discussed thermal energy storage materials and technologies in general, with a particular focus on sensible and latent heat storage approaches; a comparison of TES technologies and selection criteria for suitable materials was presented by [AR11]; see reference [AF01] for reactor design; [CLR13] discussed aspects of low-grade (waste) heat utilisation, TES, chemical heat pumps, and thermo-electrical conversion; materials and companies in the field of adsorption-powered heat pumps for the use of low-grade heat were reviewed by [Meu13]; [Ari07] focussed on material aspects, [Par+14] on high temperature heat storage; [CCC12] reviewed materials, operation cycles, and applications of TES; [YWW13] highlighted sorption-based storage for solar energy systems and included a material overview; adsorptive heat transformation with a special focus on novel materials and materials science aspects was at the centre of reference [Ari13]; [AHP12] discussed TES technologies in the context of demand-side management in the electricity grid by coupling them to electrically driven heating and cooling systems, heat pumps, cogeneration systems, etc.; material aspects and projects in the context of long-term solar energy storage using sorption processes were reviewed in reference [NTs+09]; see also a special issue on energy storage in this journal [RTY15]. [YS02] reviewed mathematical models of adsorption heat pumps and chillers, which will not be considered again here. Their review treated not only space- and time-resolved but also steady-state and lumped parameter models, while focussing on adsorption-based systems."

[3] Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee of the Regions - COM(2011) 885 final – *Energy Roadmap 2050* of 15 December 2011.

[4] See the *Energy Efficiency Directive*: `http://ec.europa.eu/energy/efficiency/eed/eed_en.htm`

1.3 Scope and Structure of this Tutorial

The tutorial is geared towards scientists and engineers who want to get a practical introduction into modelling multi-physical processes occurring inside thermochemical heat storage devices: Starting from the underlying theory, to the use of simulation software capable of describing the coupled processes, up to first “numerical experiments”. The material does not cover the numerical implementation itself in detail, nor will specific numerical solver settings be discussed. We’re also not discussing the intricacies of designing a heat storage reactor based on numerical simulations or of obtaining a best-fit parameterization of the numerical models in detail. Wherever necessary, we point to supporting literature in this context.

The tutorial starts with a general introduction to the theoretical background necessary for modelling thermochemical heat storage. The theory of porous media is introduced in Chapter 2 and specified towards the application of a fixed bed of zeolite pellets permeated by a gas mixture.

The subsequent Chapter 3 introduces the current state of OpenGeoSys in its newest version. Concepts are introduced alongside input file examples for simple problems and differences to previous OGS versions are explained.

Now that the groundwork is laid, the tutorial continues in Chapter 4 by analysing a zeolite-based heat storage reactor in detail. Specific boundary conditions, material models and other routines necessary to run this example are explained.

The following Chapter 5 builds on the reactor model build in the previous chapter and extends it by a) relaxing the assumption of a homogeneous porosity and b) simulating the cyclic operation of the heat store in order to establish its steady-state behaviour.

We close this book by some concluding remarks. A complete list of symbols can be found at the end of the book on page Page 93.

Chapter 2
Mathematical Model of the THC Processes in a fixed Adsorbent Bed

The overall goal of every modelling endeavour is to gain insight into and predict the behaviour of a physical system in a resource-efficient[1] way. Of course, experimental techniques are important, and eventually properly conducted experiments decide if our picture of the phyical reality is adequate. But certainly, experiments have their limitations:

- Many measurement techniques only provide spotty data of the system, because probes are positioned at discrete positions, or data are taken only at inlets and outlets.
- The variation of operation conditions can be very time consuming, especially when the used materials have to be regenerated between successive runs of the experiment.
- Changing the geometry (shape or size) of the experimental setup might be practically impossible.
- Catastrophic events, in particular on a larger scale, cannot be examined experimentally.
- There might be regions inaccessible to measurements. Furthermore, one might not be able to observe processes taking a very long time in the real world.
- Finally, in an experiment there might be events that cannot be isolated from others.

In those situations mathematical models play their strengths. In our field of applications that means specifically:

- We get a continuous picture of the heat storage device: E. g., the temperature and humidity at every location in the adsorbent bed. Furthermore, the conversion of the thermochemical material can be monitored in a spatially resolved manner—without additional cost—, which would imply much additional experimental effort.

[1] Time/space/equipment. In the end it always boils down to money. Well, and time: Sometimes you just don't want to wait forever.

- Operating conditions and material parameters can be changed quickly and deliberately. Thereby, limiting factors for the performance of the heat storage device can be identified: E. g., the supply or removal of vapour or heat through the bed might be too slow; or the reaction kinetics might establish the limit.
- Especially in those cases where heat or mass transport are limiting, changing the reactor geometry might be worthwhile in order to increase the performance. That, too, can be easily assessed via mathematical models.

Unfortunately, mathematical models are rather complex entities to deal with. Hence, much care must be taken such that they are constructed correctly, but just the same holds true for every experiment that is to be conducted. Luckily one does not have to reinvent the wheel everytime one wants to simulate something: Either one takes a pre-existing model off the shelf, or one adheres to some modelling best-practices that will ensure physical consistency in the models one derives.

If you prefer the first approach, you might just want to go ahead to Chapter 3 and become familiar with our simulation software OpenGeoSys. Or you might want to check which models for thermochemical heat storage devices other people derived. An overview is given in [Nag+16a]. If, on the other hand, you want to sully your hands, and derive a model by yourself, then this chapter is for you: In the following sections we will explain a method for deriving models for the behaviour of physical systems in general and of thermochemical heat storage devices in particular. In the sections of this chapter we will derive the *governing equations* of our heat storage device from general physical principles and specific physical models of the materials involved. Since the whole derivation is rather lengthy and complex, we will only discuss some important points, such that you get an intuitive understanding of the final equations. The whole derivation in its entire beauty—except for some pages of intermediate calculations—is presented in [Nag+13] and the references therein.[2]

2.1 Modelling—General Workflow

In order to make reliable predictions of the behaviour of a physical system, certain requirements must be met. What those requirements are, will be explained in this section based on a general modelling workflow, which is sketched in Fig. 2.1.

[2] In the cited paper the derivation is done for thermal non-equilibrium. In the following we will consider only the simpler case of local thermal equilibrium, which is sufficient for most sorption heat storage devices.

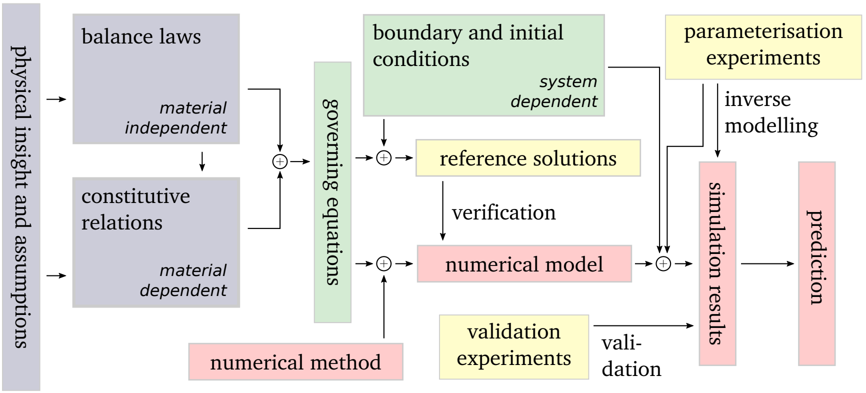


Figure 2.1 General modelling workflow: Starting with physical insight into and assumptions about the modelled system, heading for the final goal of predicting the system's behaviour. Adapted from [Nag+16a].

Even in the beginning, one already has some basic knowledge or working hypotheses about the system under examination. This knowledge, possibly together with some simplifying assumptions, first of all enables us to choose the right physical theory to describe the system. In the example of our packed beds, we know that we don't want to treat every single adsorbent bead individually, but rather in a homogenized way. Furthermore, we know that classical physics is appropriate to treat our system, and that we need to consider neither general nor special relativity. That leads us to choosing the *theory of porous media* (TPM; more on that later) to describe the system.

As every physical theory, also the TPM comes with a set *balance laws* for conserved quantities—mass, momentum, angular momentum, energy—and for the entropy. Those balance laws are valid for any system covered by the theory, independent of the specific materials present in the system.

The material-specific counterparts of the balance laws are the *constitutive relations* defining individual physical properties, e. g., the heat capacity, thermal conductivity, reaction equilibria and kinetics. Inserting them into a selection of balance laws one obtains the *governing equations* of the specific physical system: A system of partial differential equations (PDEs) for some (previously chosen) *primary variables*. For any realistic application, those PDEs are usually too complicated to be solved analytically. Therefore one has to pick a suitable *numerical method*[3], which gives rise to a *numerical model* of the system that, in turn, can be solved by computers. Although *analytical solutions* mostly are of little practical relevance, they are still important as a tool to *verify* the numerical model: Verifying that the analytical solution to the governing equations—which are generally obtained on simple geometries,

[3] We love the finite element method!

possibly using simplified material models—is conforming with the solution of the numerical model confirms that the numerical model has been correctly implemented in the simulation software.[4]

That concludes the purely theoretical part of the modelling workflow. What remains is (i) to *parametrize* the model, in particular the parameters of the constitutive models used. That can be done either by taking literature values, measuring properties, or even by running entire simulations, using the parameters of the constitutive relations as free values in order to fit some measurement data (*inverse modelling*). But parametrization experiments are not enough to ensure that the model provides reliable *predictions*, because if you have enough parameters in you model, you will be able to fit almost any parametrization experiment. Therefore we (ii) need to *validate*[5] the parametrized model against some additional measurements. Those *validation experiments* must be different from the parametrization experiments, e. g., they could use sufficiently different materials, (reactor) geometries or working conditions. For the validation one takes the previously parametrized model, applies the (possibly changed) boundary conditions of the validation experiments to it and tries to predict the outcome of the validation experiment.

If that succeeds, one can be confident that the model will make reliable predictions of the system behaviour within the range of conditions it was designed for and validated with. If not, one has to adapt the model, such that it matches the physical reality more closely.

2.2 The Physical Setting

In this chapter the steps taken to derive the governing equations for the thermal (T), hydraulic (H) and chemical (C) processes inside a thermochemical heat storage device are described. We start with the setting: We are interested in modelling the THC processes inside a packed bed of adsorbent pellets, which is located in an open sorption chamber, during both adsorption and desorption, cf. Fig. 2.2. As exemplary adsorptive/adsorbent working pair we'll consider water/zeolite throughout this book. From that we can infer the following properties, as is common for sorption heat storage devices, cf. e. g., the references in [Nag+16a]:

[4] Of course, that is no rigorous proof of a correct implementation, since only those parts of the model are checked that actually play a role for the analytical solution. E. g., if you have a model comprising both heat (T) and fluid (H) transport, but in the (successful) verification the heat transport did not play any role—e. g., because the problem was isothermal—you can't be sure that the couplet TH process is correct.

[5] N. b., there is a difference between *verification* and *validation*, although the two words are very similar: The former is a purely theoretical test that the model is self-consistent. The latter is a test that the model actually describes the physical reality to the desired degree.

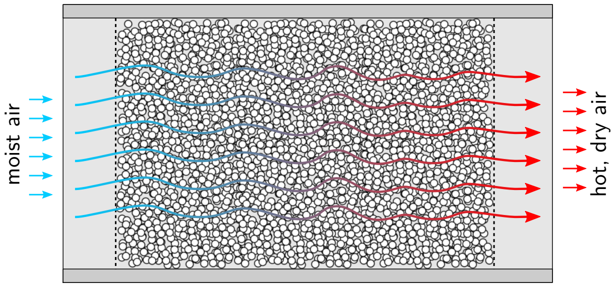


Figure 2.2 Schematic view of an open sorption chamber containing a packed bed of adsorbent pellets during an adsorption phase.

- There are two phases present in the sorption chamber: A solid and a gas phase.
- The gas phase comprises two components: The carrier gas (e. g., air) and the adsorptive (water vapour).
- There is interphase mass exchange (i. e., the ad-/desorption process), as well as interphase heat exchange.
- The typically rapid sorption kinetics depends both on the thermophysical (i. e., temperature and humidity) and chemical state (i. e., adsorbent loading) at the specific position inside the bed.
- Heat is transported both by conduction, and by advection via the flowing gas stream.
- We assume that the solid and gas phases are in local thermal equilibrium. I. e., at every single location in the bed the gas and solid temperature are equal, but the temperature is allowed to change over time and vary with varying position.
- The gas flow is taken to be compressible.
- Since zeolite is the chosen adsorbent, we can neglect volume changes during ad- and desorption. Therefore, we also don't have to bother with mechanical effects due to deforming solids or changing bed void fraction.

Furthermore, in this chapter we concentrate on the processes inside the packed bed. In particular the heat conduction inside the wall of the adsorption chamber will not be explicitly discussed. Anyway, those additional processes are typically much simpler than the coupled THC process discussed here.

2.3 Homogenized Treatment of the Packed Bed

We are confronted with an utterly complex, disordered system: A packed bed of adsorbent pellets. Modelling that system in detail would require us to know the position of every single bead in the bed—knowledge that we usually lack—, or to run an enormous number of simulations with randomly generated packings. Furthermore, both attempts mean that we have to provide

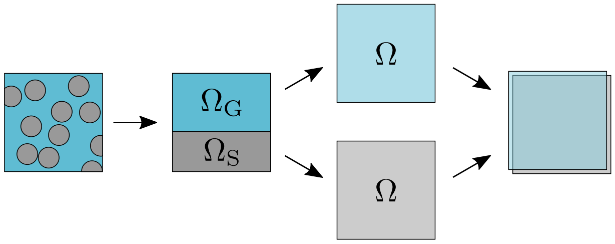


Figure 2.3
Concept of the homogenization procedure. Grey: Solid. Blue: Gas. By the homogenization the individual solid and gas subvolumes are smeared out over the entire volume Ω.

the computer with a mesh that reqresents the packed bed very precisely. That is simply not feasible, and if it was, the arising equation systems would be huge, needing a very long time to be solved.

Therefore we have to get rid of the convoluted structure of the bed and transform it to something that is easier to describe. Luckily, there is a standard procedure for treating porous media in an uncomplicated way: Homogenization. It proceeds as follows: In each subvolume Ω of the fixed bed there are parts filled with gas (amounting to the subvolume Ω_G, and parts occupied by solid material (subvolume Ω_S), such that in total $\Omega = \Omega_G \cup \Omega_S$ and $|\Omega| = |\Omega_G| + |\Omega_S|$, cf. Fig. 2.3. The simplifying assumption now is, that the physics inside the bed will not change, no matter how the solid and gas parts are distributed within Ω.[6] Hence, the only physically relevant quantities to describe the spatial solid and gas distribution inside Ω are $|\Omega_S|$ and $|\Omega_G|$. Furthermore, we can even imagine that both that solid and the gas part are smeared out over the whole volume Ω, i. e., in every single point, there exists both a fraction of gas and a fraction of solid; and the size of that fraction is determined by the respective *volume fraction* ϕ_S and ϕ_G.

For a more general system than in the example, e. g., one constisting of n phases α, each occupying a separate subvolume Ω_α, we get the following formulae:

$$\phi_\alpha := \frac{\mathrm{d}\Omega_\alpha}{\mathrm{d}\Omega} \qquad \sum_{\alpha=1}^{n} \phi_\alpha = 1\,, \tag{2.1}$$

where the right equation is the saturation constraint: Every location in Ω is occupied by something—solid or fluid, but not vacuum.

In the “smearing procedure” we only increased the volume that each phase occupies to the total volume Ω, but we kept the mass of the phase contained inside Ω constant. Therefore, the mass density apparently decreases so that we now distinguish between two density measures: The *intrinsic/real density*

[6] That requires that Ω is representative of the system under consideration (*representative elementary volume*), i. e., that it is much bigger than the structure to be averaged out (i. e., it has to contain a number of pellets in the present case). But at the same time the subvolume Ω must be much smaller than the entire system, because it must remain sensible to treat the quantities at the macroscale in a spatially resolved way.

$\varrho_{\alpha\mathrm{R}}$ and the *apparent/partial density* ϱ_α:

$$\varrho_{\alpha\mathrm{R}} = \frac{\mathrm{d}m_\alpha}{\mathrm{d}\Omega_\alpha} \qquad \varrho_\alpha = \frac{\mathrm{d}m_\alpha}{\mathrm{d}\Omega} \qquad \varrho_\alpha = \phi_\alpha \varrho_{\alpha\mathrm{R}} \tag{2.2}$$

In fact, we use such two density measures for other densities, e. g., the internal energy density, too. And the intrinsic/real densities always carry the additional index “R”.[7]

With that said, the homogenization procedure for porous media is essentially established. However, we don’t want to conclude this section without having stated the axioms of mixture theory—Truesdell’s “metaphysical” principles—explicitly, because they go beyond what we already said and will be required in the next section [Tru84, p. 221]:

1. All properties of the mixture must be mathematical consequences of properties of the constituents.
2. So as to describe the motion of a constituent, we may in imagination isolate it from the rest of the mixture, provided we allow properly for the actions of the other constituents upon it.
3. The motion of the mixture is governed by the same equations as is a single body.

2.4 Balance Laws

Now we can start the derivation of the differential equations governing the homogenized adsorbent bed. As we want to end up with a physically consistent description of the system, we begin with equations that are generally valid for any continuous system: Balance laws. As widely known, such laws can be formulated for the “usual suspect” quantities: mass, momentum, angular momentum, energy, and entropy. And we can write the balance for those quantities both for individual constituents and for the porous medium as a whole. But in any case, for a physical quantity $\mathcal{G}$ inside a volume Ω the balance law is

$$\frac{\mathrm{d}\mathcal{G}}{\mathrm{d}t} = \mathcal{F} + \mathcal{S} + \hat{\mathcal{P}}\,, \tag{2.3}$$

where $\mathcal{F}$ is the flux across the surface $\Gamma = \partial\Omega$ of Ω, $\mathcal{S}$ are sources or sinks inside Ω,[8] and $\hat{\mathcal{P}}$ is the production inside Ω. I. e., the change of $\mathcal{G}$ inside Ω amounts to the sum of (i) what flows in and out through the surface, (ii) what is supplied to or withdrawn from Ω, and (iii) what is produced or consumed inside Ω. The difference between $\mathcal{S}$ and $\hat{\mathcal{P}}$ is, that the former is caused by

[7] Well, not always: The velocity, which is an intrinsic volume flux density, is just $\boldsymbol{v}$. Instead, the apparent volume flux density is denoted as $\boldsymbol{v}_{\mathrm{Darcy}}$. See also Eq. (2.19).

[8] $\mathcal{S}$ is also called supply term.

actions at distance, and the latter is caused by interactions with other phases. E. g., if the balanced quantity is the momentum, $\mathcal{S}$ could be gravity acting on the body (i. e., an action at distance), and $\hat{\mathcal{P}}$ could be due to friction between the solid and gas phases (i. e., an interaction with other phases). In this book, all production terms carry a circumflex accent $\hat{\circ}$, and later on source/sink terms will be neglected since they are not relevant for our application.

Usually, it is more convenient to do derivations not with the integral law (2.3), but to express the quantities $\mathcal{G}$, $\mathcal{F}$, $\mathcal{S}$, and $\hat{\mathcal{P}}$ by integrals of the respective densities:

$$\mathcal{G} = \int_{\mathit{\Omega}} \gamma \, \mathrm{d}\mathit{\Omega} \quad \mathcal{F} = - \int_{\mathit{\Gamma}} \boldsymbol{\varphi} \cdot \boldsymbol{n} \, \mathrm{d}\mathit{\Gamma} \quad \mathcal{S} = \int_{\mathit{\Omega}} \sigma \, \mathrm{d}\mathit{\Omega} \quad \hat{\mathcal{P}} = \int_{\mathit{\Omega}} \hat{\pi} \, \mathrm{d}\mathit{\Omega} \tag{2.4}$$

Therein, $\boldsymbol{n}$ is the outwards oriented surface normal on the surface $\mathit{\Gamma}$ of $\mathit{\Omega}$. Consequently $\boldsymbol{\varphi} \cdot \boldsymbol{n}$ is the flux density that leaves $\mathit{\Omega}$, which gives rise to the minus sign in the formula for $\mathcal{F}$. Combining Eqs. (2.3) and (2.4), Gauss's theorem $\int_{\partial \mathit{\Omega}} \boldsymbol{\varphi} \cdot \boldsymbol{n} \, \mathrm{d}\mathit{\Gamma} = \int_{\mathit{\Omega}} \operatorname{div} \boldsymbol{\varphi} \, \mathrm{d}\mathit{\Omega}$, and taking the volume $\mathit{\Omega}$ as fixed in space, i. e., non-moving, one arrives at the local form of the balance law (2.3) in the *Eulerian description*:[9]

$$\frac{\partial \gamma}{\partial t} = - \operatorname{div} \boldsymbol{\varphi} + \sigma + \hat{\pi} \, . \tag{2.5}$$

Let's illustrate this with a concrete example: For the mass balance of a fluid flowing through space, with no other matter present, we have the mass density as quantity to be balanced ($\gamma = \varrho$), the corresponding mass flux density $\boldsymbol{\varphi} = \varrho \boldsymbol{v}$,[10] no supply ($\sigma = 0$; it is hard to imagine that mass is generated by an action at distance), and also no production $\hat{\pi} = 0$, because there is nothing that the fluid can interact with. With that given, Eq. (2.5) simplifies to the continuity equation

$$\frac{\partial \varrho}{\partial t} = - \operatorname{div}(\varrho \boldsymbol{v}) \, . \tag{2.6}$$

A second example will lead us to some more insights: Imagine a system very similar to our fixed bed reactor: There is one solid and one gas phase, both can exchange mass with each other. The mass balances for the two

[9] The Eulerian description emerges from taking $\mathit{\Omega}$ as *fixed in space*. If, on the opposite, one assumed that $\mathit{\Omega}$ be *fixed to the matter*, i. e., moves according to the motion of matter, one would take the *Lagrangian point of view*. They are both physically equivalent, they only lead to slightly different (intermediate) equations; in particular the definition of the flux $\mathcal{F}$, and accordingly the flux density $\boldsymbol{\varphi}$, differs among them.

[10] Checking the units, one can show that $\varrho \boldsymbol{v}$ is a mass flux density: $[\varrho \boldsymbol{v}] = \mathrm{kg\,m^{-3} \cdot m\,s^{-1}} = \mathrm{kg\,s^{-1}\,m^{-2}}$. In the Eulerian point of view the flux density $\boldsymbol{\varphi}$ of the quantity γ always contains a term $\gamma \boldsymbol{v}$. This term is not present in the the flux in the Lagrangian point of view; in contrast, there one has a more complex time derivative, the so-called *material time derivative*.

phases are:[11]

$$
\begin{aligned}
\frac{\partial \varrho_{\mathrm{S}}}{\partial t} &= -\operatorname{div}(\varrho_{\mathrm{S}} \boldsymbol{v}_{\mathrm{S}}) + \hat{\rho}_{\mathrm{S}} \\
\frac{\partial \varrho_{\mathrm{G}}}{\partial t} &= -\operatorname{div}(\varrho_{\mathrm{G}} \boldsymbol{v}_{\mathrm{G}}) + \hat{\rho}_{\mathrm{G}}
\end{aligned}
\tag{2.7}
$$

The left-hand sides sum up to the total mass density change, accordingly, the divergence terms on the right-hand side sum up to the total flux, and the productions to the total production. The mass balance for the overall system just looks like Eq. (2.6). Comparing Eq.s (2.7) to Eq. (2.6),[12] one finds that the production terms must sum up to zero: $\hat{\varrho}_{\mathrm{S}} + \hat{\varrho}_{\mathrm{G}} = 0$,[13] i. e., the mass lost by the solid phase equals the mass gained by the gas phase and vice versa. More generally, this reasoning leads to constraints for the production terms:

$$
\sum_{\alpha=1}^{n} \hat{\pi}_{\alpha} = 0 \tag{2.8}
$$

Now, the focus will be on the "higher" balance laws for momentum and energy. The momentum balance of a constituent α can be expressed in the form of Eq. (2.5), neglecting the source term (i. e., body forces like gravity), as

$$
\frac{\partial \varrho_{\alpha} \boldsymbol{v}_{\alpha}}{\partial t} = -\operatorname{div}(\varrho_{\alpha} \boldsymbol{v}_{\alpha} \otimes \boldsymbol{v}_{\alpha} - \boldsymbol{\sigma}_{\alpha}) + \hat{\boldsymbol{s}}_{\alpha} \,. \tag{2.9}
$$

Now the flux comprises the term $\varrho_{\alpha} \boldsymbol{v}_{\alpha} \otimes \boldsymbol{v}_{\alpha}$ expressing that some momentum is transported by the fluid flow with velocity $\boldsymbol{v}_{\alpha}$, and the mechanical stresses $\boldsymbol{\sigma}_{\alpha}$ exerting forces on the material. Moreover, the *total momentum production* $\hat{\boldsymbol{s}}_{\alpha} = \hat{\boldsymbol{p}}_{\alpha} + \hat{\varrho}_{\alpha} \boldsymbol{v}_{\alpha}$ is a combination of the *direct momentum production* ($\hat{\boldsymbol{p}}_{\alpha}$; e. g., due to friction between phases) and a "lower" production term $\hat{\varrho}_{\alpha} \boldsymbol{v}_{\alpha}$ reflecting that produced mass also carries some momentum. Again, the *total* productions are constrained: $\sum_{\alpha} \hat{\boldsymbol{s}}_{\alpha} = 0$. Using the product rule on the time derivatives and divergence terms in Eq. (2.9), and employing the component mass balance—cf. Eq.s (2.7)—one arrives at:

$$
\varrho_{\alpha} \frac{\partial \boldsymbol{v}_{\alpha}}{\partial t} = -\varrho_{\alpha} \operatorname{grad} \boldsymbol{v}_{\alpha} \cdot \boldsymbol{v}_{\alpha} + \operatorname{div} \boldsymbol{\sigma}_{\alpha} + \hat{\boldsymbol{p}}_{\alpha} \tag{2.10}
$$

Note that in the equation above the lower production term $\hat{\varrho}_{\alpha}$ has vanished.

Similarly, for the constituent energy balance we get:

[11] To be general, we allow the solid to move with velocity $\boldsymbol{v}_{\mathrm{S}}$.

[12] That uses the axiom stating that the whole mixture is governed by the same equations as its individual constituents.

[13] Note that those production terms $\hat{\varrho}_{\mathrm{S}}$ and $\hat{\varrho}_{\mathrm{G}}$ are related to the apparent densities, not to the real/intrinsic densities!

$$\frac{\partial \varrho_\alpha \big(u_\alpha + \frac{1}{2}\boldsymbol{v}_\alpha^2\big)}{\partial t} = -\operatorname{div}\big[\varrho_\alpha \big(u_\alpha + \tfrac{1}{2}\boldsymbol{v}_\alpha^2\big)\boldsymbol{v}_\alpha\big] + \operatorname{div}\big(\boldsymbol{\sigma}_\alpha^\mathsf{T}\boldsymbol{v}_\alpha - \boldsymbol{q}_\alpha\big) + \hat{e}_\alpha \,. \tag{2.11}$$

Therein, the *specific energy* comprises the specific internal energy u_α and the specific kinetic energy $\frac{1}{2}\boldsymbol{v}_\alpha^2$. The flux contains the usual advective term; and the terms $\boldsymbol{\sigma}_\alpha^\mathsf{T}\boldsymbol{v}_\alpha - \boldsymbol{q}_\alpha$ that quantify the acceleration work due to acting stresses and the conductive heat flux, respectively. The total energy production $\hat{e}_\alpha$ can be decomposed in a direct internal energy production term and terms representing the power related to the momentum production, and the internal and kinetic energy carried by the produced mass:

$$\hat{e}_\alpha = \hat{u}_\alpha + \hat{\boldsymbol{p}}_\alpha \cdot \boldsymbol{v}_\alpha + \hat{\varrho}_\alpha \big(u_\alpha + \tfrac{1}{2}\boldsymbol{v}_\alpha^2\big) \tag{2.12}$$

For the total production $\hat{e}_\alpha$ again the constraint $\sum_\alpha \hat{e}_\alpha = 0$ holds. And as in the case of the momentum balance one can eliminate the lower productions from the balance law by using the mass and the momentum balance:

$$\varrho_\alpha \frac{\partial u_\alpha}{\partial t} = -\varrho_\alpha \operatorname{grad} u_\alpha \cdot \boldsymbol{v}_\alpha + \boldsymbol{\sigma}_\alpha : \operatorname{grad} \boldsymbol{v}_\alpha - \operatorname{div} \boldsymbol{q}_\alpha + \hat{u}_\alpha \,. \tag{2.13}$$

We want to conclude this section about the general balance laws with a summary of what has been derived, which will elucidate the connection between the Eulerian and Lagrangian point of view:

$$\begin{aligned}
\frac{\partial \varrho_\alpha}{\partial t} &+ \operatorname{grad} \varrho_\alpha \cdot \boldsymbol{v}_\alpha = -\varrho_\alpha \operatorname{div} \boldsymbol{v}_\alpha + \hat{\varrho}_\alpha && \text{(mass)}\\
\varrho_\alpha \frac{\partial \boldsymbol{v}_\alpha}{\partial t} &+ \varrho_\alpha \operatorname{grad} \boldsymbol{v}_\alpha \cdot \boldsymbol{v}_\alpha = \operatorname{div} \boldsymbol{\sigma}_\alpha + \hat{\boldsymbol{p}}_\alpha && \text{(momentum)}\\
\varrho_\alpha \frac{\partial u_\alpha}{\partial t} &+ \varrho_\alpha \operatorname{grad} u_\alpha \cdot \boldsymbol{v}_\alpha = \boldsymbol{\sigma}_\alpha : \operatorname{grad} \boldsymbol{v}_\alpha - \operatorname{div} \boldsymbol{q}_\alpha + \hat{u}_\alpha && \text{(energy)}
\end{aligned}$$

The left-hand sides of the equations above all contain the partial time derivative and a second term involving $\operatorname{grad} \bullet \cdot \boldsymbol{v}_\alpha$, where the latter represents that fraction of the quantity $\bullet$ that is *advected* by the motion with velocity $\boldsymbol{v}_\alpha$. In the Lagrangian description both terms are usually combined to the *material time derivative* $(\bullet)'_\alpha := \frac{\partial}{\partial t} \bullet + \operatorname{grad} \bullet \cdot \boldsymbol{v}_\alpha$ that measures the change of quantity $\bullet$ contained inside a volume attached to the moving material. E. g., from the mass balance above, one sees that the mass density decreases if the velocity field is divergent.[14] That illustrates that in the end the Eulerian and the Lagrangian point of view lead to the same equations.

[14] In a divergent velocity field (e. g., $\boldsymbol{v}_\alpha(\boldsymbol{x}) = c\boldsymbol{x}$, $c > 0$), the volume of the moving material increases. Since the mass enclosed in that volume stays constant (for zero production), the density decreases. Mathematically, in that case we have $\operatorname{div} \boldsymbol{v}_\alpha > 0$, i. e., the formula given for the mass balance is constistent with our expectation.

2.5 Constitutive Relations → Governing Equations

After having treated the basic balance laws in detail, in this section we'll introduce constitutive relations in order to make contact to the specific application area: thermochemical heat storage devices. Since the whole derivation is rather lengthy, and can be found in [Nag+13], we won't repeat it here in every detail. Rather, we highlight some important features, such that the emerging governing equations can be intuitively understood.

First of all we have to select a set of balance equations that is sufficient to describe the physics in the adsorbent bed. We select the gas mass balance

$$\frac{\partial \varrho_\mathrm{G}}{\partial t} = -\operatorname{div}(\varrho_\mathrm{G} \boldsymbol{v}_\mathrm{G}) + \hat{\varrho}_\mathrm{G}\,, \tag{2.14}$$

the solid mass balance

$$\frac{\partial \varrho_\mathrm{S}}{\partial t} = -\operatorname{div}(\varrho_\mathrm{S} \boldsymbol{v}_\mathrm{S}) + \hat{\varrho}_\mathrm{S} \qquad \text{with } \boldsymbol{v}_\mathrm{S} = 0\,, \tag{2.15}$$

the overall energy balance

$$\begin{aligned} &\varrho_\mathrm{S} \frac{\partial u_\mathrm{S}}{\partial t} + \varrho_\mathrm{S} \operatorname{grad} u_\mathrm{S} \cdot \boldsymbol{v}_\mathrm{S} + \varrho_\mathrm{G} \frac{\partial u_\mathrm{G}}{\partial t} + \varrho_\mathrm{G} \operatorname{grad} u_\mathrm{G} \cdot \boldsymbol{v}_\mathrm{G} \\ &\quad = \boldsymbol{\sigma}_\mathrm{S} : \operatorname{grad} \boldsymbol{v}_\mathrm{S} + \boldsymbol{\sigma}_\mathrm{G} : \operatorname{grad} \boldsymbol{v}_\mathrm{G} - \operatorname{div}(\boldsymbol{q}_\mathrm{S} + \boldsymbol{q}_\mathrm{G}) + \hat{u}_\mathrm{S} + \hat{u}_\mathrm{G} \end{aligned} \tag{2.16}$$

and the vapour mass balance

$$\frac{\partial \varrho_\mathrm{V}}{\partial t} = -\operatorname{div}(\varrho_\mathrm{V} \boldsymbol{v}_\mathrm{V}) + \hat{\varrho}_\mathrm{V}\,. \tag{2.17}$$

They reflect what we are mainly interested in: The mass and heat transport through the bed, and the thermal effects and mass exchange due to the sorption processes. We know additionally that $\hat{e} = \sum_\alpha \hat{e}_\alpha = 0$, because energy cannot be produced from nothing, and $\hat{\varrho}_\mathrm{V} = \hat{\varrho}_\mathrm{G}$, because during the sorption process only water vapour is released or consumed and all other gas components are considered inert. The solid mass balance Eq. (2.15) for the non-moving solid implies that the evolution of the solid mass directly follows from the reaction kinetics. Therefore, the solid density can be treated as an internal variable, thereby eliminating Eq. (2.15) from the PDE system.

Second, we have to pick a suitable set of unknown quantities that we want to solve the PDE system for. We have three equations left, hence three unknowns. We choose the total gas pressure p, the temperature T and the mass fraction $x_{m\mathrm{V}}$ of vapour in the total (moist) air. In principle, instead of $x_{m\mathrm{V}}$ we could also have chosen the vapour partial pressure as the third unknown. Consequently, we have to express all other quantities—ϱ, $\boldsymbol{v}$, u, etc.—in the above formulas in terms of p, T, and $x_{m\mathrm{V}}$. That is done by using constitutive relations for the individual materials.

Let's focus on the gas mass balance Eq. (2.14) now. From Eq. (2.8) we know that $\hat{\varrho}_\mathrm{G} = -\hat{\varrho}_\mathrm{S}$. Furthermore, we can expand the gas density using Eq. (2.2), and we rearrange the divergence term by introducing the Darcy velocity $\boldsymbol{v}_\mathrm{Darcy} := \phi_\mathrm{G}\boldsymbol{v}_\mathrm{G}$:

$$\phi_\mathrm{G}\frac{\partial \varrho_\mathrm{GR}}{\partial t} + \frac{\partial \phi_\mathrm{G}}{\partial t}\varrho_\mathrm{GR} = -\operatorname{div}(\varrho_\mathrm{GR}\boldsymbol{v}_\mathrm{Darcy}) - \hat{\varrho}_\mathrm{S}\,. \tag{2.18}$$

Since volume changes of the packed bed are neglected ($\frac{\partial}{\partial t}\phi_\mathrm{G} = 0$), the second term on the left-hand side vanishes. For the Darcy velocity we can readily employ Darcy's law and the Kozeny-Carman equation:

$$\boldsymbol{v}_\mathrm{Darcy} = -\frac{\boldsymbol{k}}{\mu_\mathrm{G}}\operatorname{grad} p \qquad\qquad \boldsymbol{k} = \frac{d_\mathrm{p}^2\,\phi_\mathrm{G}^3}{150\,(1-\phi_\mathrm{G})^2}\,\mathbf{1}\,, \tag{2.19}$$

where the former relates the pressure gradient to the volume flux through a porous medium,[15] and the latter computes the intrinsic permeability of a packed bed of spheres having a diameter of d_p each.

Taking both the carrier gas and the water vapour inside it as ideal gases, one can compute the intrinsic gas density by

$$\varrho_\mathrm{GR} = \frac{p\,M_\mathrm{G}(x_{m\mathrm{V}})}{RT}\,, \tag{2.20}$$

where M_G is the effective molar mass of the gas mixture, that depends on the vapour mass fraction. Substituting Eq. (2.20) into Eq. (2.18) one arrives at the first governing equation:

$$\phi_\mathrm{G}\varrho_\mathrm{GR}\left(\frac{1}{p}\frac{\partial p}{\partial t} - \frac{1}{T}\frac{\partial T}{\partial t} + \frac{1}{M_\mathrm{G}}\frac{\partial M_\mathrm{G}}{\partial x_{m\mathrm{V}}}\frac{\partial x_{m\mathrm{V}}}{\partial t}\right) = -\operatorname{div}(\varrho_\mathrm{GR}\boldsymbol{v}_\mathrm{Darcy}) - \hat{\varrho}_\mathrm{S}\,. \tag{2.21}$$

Next, we'll consider the vapour mass balance Eq. (2.17). We can split the vapour velocity $\boldsymbol{v}_\mathrm{V}$ into $\boldsymbol{v}_\mathrm{G} + \boldsymbol{d}_\mathrm{V}$, where the second term is the vapour diffusion velocity relative to the overall gas flow. Moreover, we know that $\varrho_\mathrm{V} = x_{m\mathrm{V}}\varrho_\mathrm{G}$ by the definition of $x_{m\mathrm{V}}$ we gave earlier. Applying both yields

$$\phi_\mathrm{G}\rho_\mathrm{GR}\frac{\partial x_{m\mathrm{V}}}{\partial t} = -\operatorname{div}(\varrho_\mathrm{V}\boldsymbol{d}_\mathrm{V}) - \operatorname{grad} x_{m\mathrm{V}}\cdot\varrho_\mathrm{G}\boldsymbol{v}_\mathrm{G} + (1-x_{m\mathrm{V}})\hat{\varrho}_\mathrm{V}\,. \tag{2.22}$$

Again, the factor $\varrho_\mathrm{G}\boldsymbol{v}_\mathrm{G}$ can be transformed using the Darcy velocity. The term $\varrho_\mathrm{V}\boldsymbol{d}_\mathrm{V} =: \dot{\boldsymbol{m}}_{\mathrm{V},\boldsymbol{d}}$ is the vapour diffusion mass flux density. We assume that a simplified Fickian diffusion model can be used for it:

[15] As already mentioned earlier, the Darcy velocity—also called superficial, seepage, or empty tower velocity—is not the velocity of the flowing gas, but its volume flux density. They would be the same if the bed was empty, i. e., if there was only gas in the sorption chamber.

$$\dot{\boldsymbol{m}}_{\mathrm{V},\boldsymbol{d}} = \varrho_{\mathrm{GR}} \boldsymbol{D} \operatorname{grad} x_{m\mathrm{V}} \,, \tag{2.23}$$

Where $\boldsymbol{D}$ is the diffusion tensor for vapour inside the carrier gas. This model is simplified, because usually diffusion is taken to be driven by concentration gradients. Here in contrast, a mass fraction gradient is the driver, see [Nag+13] for details. Taking all together gives rise to the second governing equation:

$$\begin{aligned}&\phi_{\mathrm{G}}\rho_{\mathrm{GR}} \frac{\partial x_{m\mathrm{V}}}{\partial t} = \\ &\quad - \operatorname{div}(\varrho_{\mathrm{GR}} \boldsymbol{D} \operatorname{grad} x_{m\mathrm{V}}) - \varrho_{\mathrm{GR}} \operatorname{grad} x_{m\mathrm{V}} \cdot \boldsymbol{v}_{\mathrm{Darcy}} - (1 - x_{m\mathrm{V}})\hat{\varrho}_{\mathrm{S}} \,.\end{aligned} \tag{2.24}$$

Deriving the last governing equation from the overall energy balance is not as straightforward as were the mass balances. In particular, some relations implied by the entropy inequality are needed to finish the derivation. However, also treating the entropy inequality in detail would clearly go beyond this tutorial book. For brevity, therefore we'll only explain the different terms in the final governing equation. If you are interested in the whole calculation, please refer to [Nag+13]. Et voilà, the final result:

$$\begin{aligned}&[\phi_{\mathrm{G}}\rho_{\mathrm{GR}}c_{p\mathrm{G}} + (1-\phi_{\mathrm{G}})\rho_{\mathrm{SR}}c_{p\mathrm{S}}] \frac{\partial T}{\partial t} + \rho_{\mathrm{GR}}c_{p\mathrm{G}} \operatorname{grad} T \cdot \boldsymbol{v}_{\mathrm{Darcy}} - \phi_{\mathrm{G}} \frac{\partial p}{\partial t} \\ &\qquad = \operatorname{div}\big[\big(\phi_{\mathrm{G}}\boldsymbol{\lambda}_{\mathrm{GR}} + (1-\phi_{\mathrm{G}})\boldsymbol{\lambda}_{\mathrm{SR}}\big) \operatorname{grad} T\big] + (1-\phi_{\mathrm{G}})\hat{\rho}_{\mathrm{SR}}\mathit{\Delta} h_{\mathrm{ads}}\end{aligned} \tag{2.25}$$

The first line in that equation basically corresponds to the material time derivative part—also the first line—of Eq. (2.16): $\sum_\alpha \varrho_\alpha(\frac{\partial}{\partial t}u_\alpha + \operatorname{grad} u_\alpha \cdot \boldsymbol{v}_\alpha)$. During the derivation the energy balance is reformulated in terms of enthalpies

$$h_{\mathrm{G}} = u_{\mathrm{G}} + p/\varrho_{\mathrm{GR}} \qquad\qquad h_{\mathrm{S}} = u_{\mathrm{S}} \,, \tag{2.26}$$

where the specific enthalpy of the solid has that simple form, because deformations of the solid are neglected. Furthermore, for enthalpy changes the following holds:

$$\mathrm{d}h_{\mathrm{G}} = c_{p\mathrm{G}}\,\mathrm{d}T \qquad\qquad \mathrm{d}h_{\mathrm{S}} = c_{p\mathrm{S}}\,\mathrm{d}T \,, \tag{2.27}$$

where for the gas phase the ideal gas law was applied, and in the solid case again no deformations were assumed.[16] Plugging both into Eq. (2.16) yields (i) the term $\frac{\partial}{\partial t}p$, because of the split of u_{G} into h_{G} and the volume work, (ii) the partial time derivative term $\frac{\partial}{\partial t}T$ describing how much heat is stored inside the gas and the solid at a specific position and time, and (iii) the term $\operatorname{grad} T$ quantifying the heat that is advected by the flowing gas.

The second line in Eq. (2.25) comprises (i) the the heat conduction in both the solid and gas phases modelled with Fourier's law, $\boldsymbol{q} = -\boldsymbol{\lambda} \operatorname{grad} T$; and (ii) the heat released or consumed by the sorption process $(1-\phi_{\mathrm{G}})\hat{\rho}_{\mathrm{SR}}\mathit{\Delta} h_{\mathrm{ads}}$.

[16] Additionally we assume the heat capacities $c_{p\alpha}$ are constant.

Thus, the derivation of the governing equations (2.21), (2.24) and (2.25) for the packed adsorbent bed is complete. Well, almost complete. The only thing that's missing are the constitutive laws for the reaction rate and the heat of adsorption. Both will be covered in the following sections. To solve those equations, of course they have to be cast in a form amenable to numerical techniques. In the case of the finite element method that means, that one has to transform the present PDE system (i. e., the strong form) into a weak form that then can be discretized and implemented in some software. The derivation of the weak form is more or less straightforward and does not lead to many new insights. Therefore we leave it to the reader to do this exercise, or to look it up in [Nag+13].

Finally, we want to comment on a simplification in the governing equations in the case of a closed sorption chamber where the gas phase consists of the adsorptive only, i. e., $x_{m\mathrm{V}} \equiv 1$. Then one does not need the vapour mass balance Eq. (2.17). Consequently, Eq. (2.24) is omitted, too; and of course all partial derivatives with respect to $x_{m\mathrm{V}}$ vanish.

2.6 Dubinin-Polanyi Theory of Adsorption

This section deals with constitutive relations describing (i) the adsorption equilibria of microporous materials and (ii) the enthalpy of adsorption. Based on the equilibria, the subsequent section then will give a relation for the adsorption kinetics.

Microporous solids exhibit a bivariate adsorption equilibrium. It can be expressed in the form of the following equation:

$$C_{\mathrm{eq}} = C_{\mathrm{eq}}(p_{\mathrm{V}}, T)\,, \tag{2.28}$$

where C_{eq} is the mass of adsorbate per unit mass of completely desorbed adsorbent under equilibrium conditions, i. e., the equilibrium loading. In terms of densities the loading C is defined as:

$$C := \frac{\varrho_{\mathrm{SR}}}{\varrho_{\mathrm{SR,dry}}} - 1 \tag{2.29}$$

For brevity, here we will refer to water vapour as adsorptive, but the formulas also apply to other adsorptives. Usually, adsorption equilibria are determined experimentally by measuring a set of adsorption isotherms (or isobars), cf. Fig. 2.4. Those data form a discrete picture of Eq. (2.28), but in the course of a simulation run, one needs to know C_{eq} over an almost continuous range of vapour partial pressure and temperature values. To get this continuous picture, one could (i) directly interpolate the measured data, or (ii) fit each isotherm individually with some isotherm model, fitting the temperature dependence of the coefficients of the first fit in a second stage, or (iii) assume

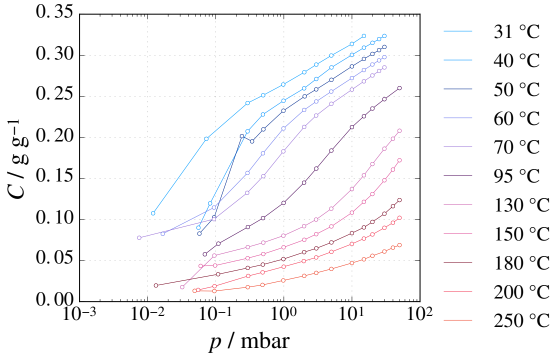


Figure 2.4 Adsorption isotherms for water onto Köstrolith® 13XBFK: Equilibrium loading C plottet against vapour partial pressure p.

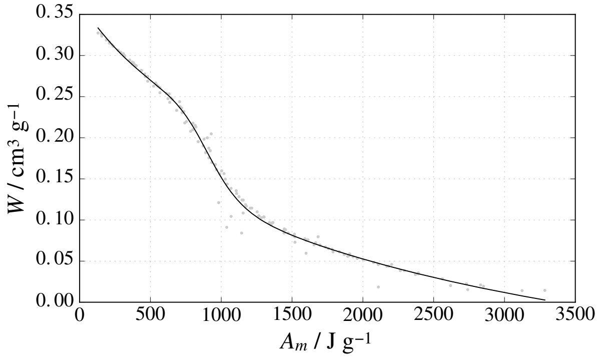


Figure 2.5 Adsorption isotherms from Fig. 2.4 mapped to the W-A_m space (dots), and fit of the emerging characteristic curve (solid line).

a temperature dependence of the coefficients of the isotherm model and fit all data at once.

However, here we pursue a different approach based on the Dubinin-Polanyi theory of micropore filling, cf. [Dub60; BDS66]. In this theory the adsorption equilibrium is not treated in C-p_V-T space, but in the W-A_m space, where W is the specific adsorbed volume

$$W := C/\varrho_\mathrm{ads}(T) \tag{2.30}$$

with the density of the adsorbate ϱ_ads, and A_m is the specific work of adsorption

$$A_m = \frac{RT}{M_\mathrm{V}} \ln \frac{p_\mathrm{sat}}{p_\mathrm{V}} \,. \tag{2.31}$$

Using those equations the measured data triplets $(C_\mathrm{eq}, p_\mathrm{V}, T)$ can be mapped to pairs (W, A_m) in W-A_m space, cf. Fig. 2.5. The central idea of the Dubinin-Polanyi theory is, that if the so mapped data pairs (W, A_m) lie on a single *characteristic curve* $W(A_m)$, one can compute the equilibrium

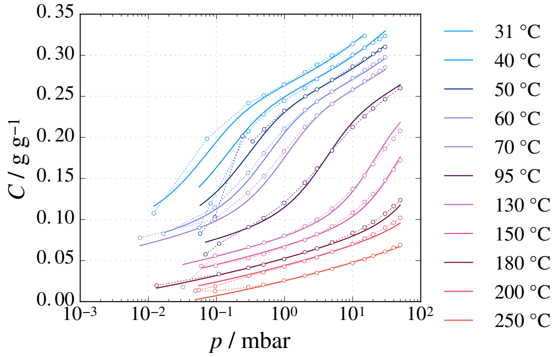


Figure 2.6
Solid lines: Adsorption equlibria reconstructed from the CC Fig. 2.5. Dashed lines: Measured adsorption equilibria, cf. Fig. 2.4.

loading for arbitrary values of p_{V} and T from that characteristic curve, cf. Fig. 2.6. The crucial point here is, that the functional relation $W(A_m)$ in particular must not depend on temperature, i. e., all isotherms are indeed mapped to the same curve.

There exist several formualtions for the characteristic curve. A very widely used one is the Dubinin-Astakhov equation

$$W(A_m) = W_0 \exp\bigl(-(A_m/E)^n\bigr) \tag{2.32}$$

that has been derived for active carbons originally. This equation is not suitable to accurately fit adsorption data of zeolites. In fact, we have shown that doing so leads to significant errors both in the computed adsorption equilibria and in water loading lifts as well as heat storage densities derived from them [Sem+17]. For zeolites a purely empirical fit equation, first proposed by Núñez [Núñ02], has proven to be a better choice:

$$W(A_m) = \frac{a_0 + a_2\,A_m + a_4\,A_m^2 + a_6\,A_m^3}{1 + a_1\,A_m + a_3\,A_m^2 + a_5\,A_m^3}\,. \tag{2.33}$$

The data in Fig. 2.5 have been fitted with this equation.

Employing the Clausius-Clapeyron equation, and doing some transformations, one arrives at a formula for the specific enthalpy of adsorption:

$$\mathit{\Delta} h_{\mathrm{ads}} = \mathit{\Delta} h_{\mathrm{V}} + A_m - T\mathit{\Delta} s\,, \tag{2.34}$$

where $\mathit{\Delta} h_{\mathrm{V}}$ is the specific enthalpy of evaporation, and the last term $-T\mathit{\Delta} s$ measures the entropy decrease upon vapour adsorption:

$$\mathit{\Delta} s = \alpha_{T\,\mathrm{ads}}\,W\,\frac{\partial A_m}{\partial W}\bigg|_T\,. \tag{2.35}$$

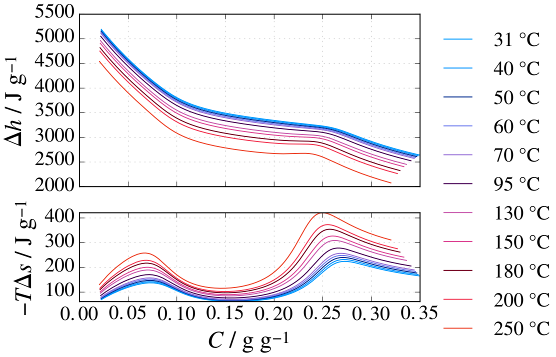


Figure 2.7
Top: Adsorption enthalpy computed from the CC using Eq. (2.34). *Bottom*: Entropic part of the computed adsorption enthalpy.

The resulting loading- and temperature-dependent specific enthalpy of adsorption emerges as a monotonically decreasing function of C, with the main contributions by h_V and A_m, cf. Fig. 2.7.

With that we have obtained the first ingredient, Δh_ads, for the description of the sorption process. The second one, $\hat{\varrho}_\mathrm{SR} \propto \frac{\partial}{\partial t} C$, will be discussed in the following section. Summarizing, the Dubinin-Polanyi theory provides a convenient framework for the reconstruction of the entire adsorption equilibrium from experimental data. In particular, the characteristic curve, Fig. 2.5, provides an intuitive visualization of the fit quality; and the fitting procedure itself is rather robust against individual outliers, because all data are mapped to one curve. Yet, there is a parameter in the Dubinin-Polanyi theory that is not accessible to experimental measurements: The adsorbate density $\varrho_\mathrm{ads}(T)$, which in general can be different from the density of a “free” liquid. Therefore, several different adsorbate density models have been proposed in the literature. However, we could show that the choice of a particular density model—whereas strongly changing Δh_ads via Δs, Eqs. (2.34) and (2.35)—only has little influence on the integral heat storage density computed for a thermochemical heat storage device [Leh+17].

2.7 Adsorption Kinetics

Not only are the THC processes in the packed bed somewhat complex, but so are the processes inside each adsorbent pellet. For a water molecule immersed in the gas flow through the packed to finally attach to an adsorption site in a zeolite micropore, it has to pass several stages: At first, the gas boundary film around the pellet has to be hurdled, then the molecule has to travel through the transport pores of the pellet reaching the adsorption site where it finally can be adsorbed, cf. Fig. 2.8. All those stages entail certain resistances for transport, namely:

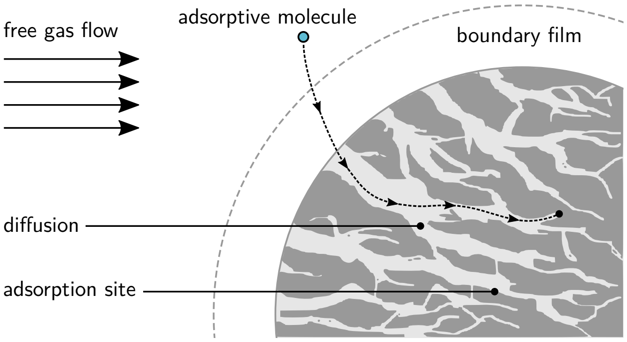


Figure 2.8 Scheme of the path an adsorptive molecule has to take on its way to an adsorption site and of the transport resistances involved. Adapted from [TH17].

- The boundary film resistance whose importance depends on the characteristics of the flow around the pellet.
- Different diffusion mechanisms in the pellet's transport pores, e. g.surface diffusion, Knudsen diffusion or continuum diffusion. Which of those mechanisms play a role, and how they are weighted depends on the specific material, and in particular on the size distribution of the tranport pores.
- If the adsorbed phase is sufficiently mobile—that depends on the adsorbent—, there could be diffusion in the adsorbate, especially in the case of high loadings.
- The geometry of the pore system can give rise to wound particle paths, i. e., for the adsorptive molecules the distance of travel is much higher than the linear distance to the adsorption site. This is usually expressed by the so-called tortuosity factor τ.

But not only those mass transport resistances affect the adsorption kinetics. Also the heat transfer—by conduction inside the pellet, and from pellet to pellet over the contact points; but also from the pellet surface to the surrounding gas flow—can be limiting via the temperature dependence of the adsorption equilibrium. All in all, this coupled heat and mass transfer problem would lead to a PDE system pretty much the same as we derived in Section 2.5 (albeit on a different scale), and one would have to solve this nested PDE system for every single pellet, leading to a full-fledged mulit-scale problem. That is, of course, practically unfeasible. Therefore, effective models for the kinetics have been developed, which collect all those resistances into a few lumped parameters.

A particularly successful model is the *linear driving force* (LDF) model, introduced by Glueckauf in 1955 [Glu55], where the rate of change of loading linearly depends on the difference between the current loading C of the adsorbent, and the equilibrium loading C_{eq}. For a spherical pellet of radius r_{p} it is given by:

$$\frac{\partial C}{\partial t} = \frac{15\,D_{\mathrm{eff}}}{r_{\mathrm{p}}^2}\bigl(C_{\mathrm{eq}}(p_{\mathrm{V}}, T) - C\bigr)\,. \tag{2.36}$$

Therein D_{eff} is an effective diffusion coefficient subsuming all transport resistances. Neglecting the pellet's boundary film resistance, D_{eff} can be computed via [Kas88]

$$D_{\mathrm{eff}} = \frac{D_{\mathrm{comb}}\,\phi_{\mathrm{Gp}}\,M_{\mathrm{V}}}{\tau_{\mathrm{p}}\,\varrho_{\mathrm{SR,dry}}\,R\,T}\left(\frac{\partial C_{\mathrm{eq}}}{\partial p_{\mathrm{V}}}\right)^{-1}\,. \tag{2.37}$$

Clearly, the effective diffusion coefficient depends on the slope of the adsorption isotherm, $\frac{\partial}{\partial p_{\mathrm{V}}} C_{\mathrm{eq}}$. D_{comb} is a combined diffusion coefficient describing all mass transport resistances inside the pellet, and ϕ_{Gp} is the volume fraction of pores inside the adsorbent pellet.

Note that the volume fractions ϕ_{G} and ϕ_{S} used so far, measure the solid and gas volume fractions of the packed bed. I. e., ϕ_{S} is the volume fraction of pellets including all the pellet's pores. In contrast, ϕ_{Gp} is the volume fraction of pores of the volume of a single pellet. Likewise, $\varrho_{\mathrm{SR,dry}}$ is the density of the adsorbent pellets including all their pores. Hence we're dealing with a pore hierarchy reflected in a series of volume fractions and density measures, which makes careful interpretation necessary.

Linking the mass production to the loading change rate is straightforward:

$$\hat{\varrho}_{\mathrm{S}} = (1 - \phi_{\mathrm{G}})\,\varrho_{\mathrm{SR,dry}}\,\frac{\partial C}{\partial t}\,. \tag{2.38}$$

Therewith, contact to the governing equations from Section 2.5 is made, and the theory part of this book is concluded. In the subsequent chapters the software OpenGeoSys will be introduced, and the modelling of thermochemical heat storage devices with it will be explained.

References

Multiphysics Modelling

Books

Many text books have been written about the TPM. Usually, they have a broader scope than the derivation presented in this chapter. In particular, they also cover deformations of the solid material. Some of those books are listed here. If you'd rather avoid the complexity added by the treatment of solid mechanics, you can pick a model from a paper from our review article [Nag+16a].

[EB02] W. Ehlers and J. Bluhm, eds. *Porous Media*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002.

[Eri76] A. C. Eringen, ed. *Mixtures and EM Field Theories*. Continuum physics Vol. 3. New York, NY: Acad. Press, 1976.

[HJ04] K. Hutter and K. Jöhnk. *Continuum Methods of Physical Modeling*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[TN65] C. Truesdell and W. Noll. "The Non-Linear Field Theories of Mechanics". In: *The Non-Linear Field Theories of Mechanics*. Springer Berlin Heidelberg, 1965, pp. 1–579.

[Tru84] C. Truesdell. *Rational Thermodynamics*. New York, NY: Springer New York, 1984.

Other Publications

[Nag+13] T. Nagel, H. Shao, A. K. Singh et al. "Non-Equilibrium Thermochemical Heat Storage in Porous Media: Part 1 − Conceptual Model". In: *Energy* 60 (2013), pp. 254–270.

[Nag+16a] T. Nagel, S. Beckert, C. Lehmann, R. Gläser and O. Kolditz. "Multi-Physical Continuum Models of Thermochemical Heat Storage and Transformation in Porous Media and Powder Beds—A Review". In: *Applied Energy* 178 (2016), pp. 323–345.

Adsorption Equilibria and Kinetics

Books

[Do98] D. D. Do. *Adsorption Analysis: Equilibria and Kinetics*. Series on chemical engineering v 2. London: Imperial College Press, 1998.

[Kas88] W. Kast. *Adsorption aus der Gasphase: ingenieurwissenschaftliche Grundlagen und technische Verfahren*. Weinheim: VCH, 1988.

[Rut84] D. M. Ruthven. *Principles of Adsorption and Adsorption Processes*. New York: Wiley, 1984.

Other Publications

[BDS66] B. Bering, M. Dubinin and V. Serpinsky. "Theory of Volume Filling for Vapor Adsorption". In: *Journal of Colloid and Interface Science* 21.4 (1966), pp. 378–393.

[Dub60] M. Dubinin. “Theory of the Physical Adsorption of Gases and Vapors and Adsorption Properties of Adsorbents of Various Natures and Porous Structures”. In: *Bulletin of the Academy of Sciences of the USSR, Division of Chemical Science* 9.7 (1960), pp. 1072–1078.

[Glu55] E. Glueckauf. “Theory of Chromatography. Part 10. Formulæ for Diffusion into Spheres and Their Application to Chromatography”. In: *Transactions of the Faraday Society* 51 (1955), p. 1540.

[Leh+17] C. Lehmann, S. Beckert, R. Gläser, O. Kolditz and T. Nagel. “Assessment of Adsorbate Density Models for Numerical Simulations of Zeolite-Based Heat Storage Applications”. In: *Applied Energy* 185 (2017), pp. 1965–1970.

[Núñ02] T. Núñez. “Charakterisierung und Bewertung von Adsorbentien für Wärmetransformationsanwendungen”. PhD thesis. Freiburg im Breisgau: Albert-Ludwigs-Universität, 2002.

[Sem+17] S. Semprini, C. Lehmann, S. Beckert et al. “Numerical Modelling of Water Sorption Isotherms of Zeolite 13XBF Based on Sparse Experimental Data Sets for Heat Storage Applications”. In: *Energy Conversion and Management* 150 (2017), pp. 392–402.

[TH17] K. Tan and B. Hameed. “Insight into the Adsorption Kinetics Models for the Removal of Contaminants from Aqueous Solutions”. In: *Journal of the Taiwan Institute of Chemical Engineers* 74 (2017), pp. 25–48.

Chapter 3
The FEM Simulation Software OpenGeoSys 6

OpenGeoSys (OGS) together with its predecessors looks back at a long history as a simulation code of coupled thermal, hydraulic, mechanical and chemical (THMC) phenomena in fractured porous media: Starting as the Fortran code *Rockflow* in the 1986, ranging over a C implementation, and now using different flavours of the C++ programming language; in the meantime undergoing two renames, namely to GeoSys and now OpenGeoSys. The development of OpenGeoSys version 6—largely from scratch, with first drafts dating six years back now—was started due to the requirement of having a more performant, flexible and maintainable simulation software offering a range of new numerical methods to solve multi-physics problems.

This flexibility, maintainability and extensibility is achieved on the implementation level by adhering to object oriented design, which brings about modular, well encapsulated source code. On the project management level these goals are supported by procedures of quality assurance, namely code review and continuous integration (CI): Each code change proposed to be included in OGS-6 is reviewed by two other developers before it is merged into the main development line. This process is completely public on GitHub[1]. Moreover, several CI tests—both unit tests and test simulations with known reference results—are run whenever changes are proposed, and additionally in regular intervals on the main development line.

The new version of OpenGeoSys, OGS-6, aims to be a solver of fully coupled THMC processes. As of this writing (autumn 2017), full THMC coupling has not been achieved in OGS-6[2], yet, but several subsets of those four letters can already be tackled: H, T, M, HT, HM, TM, HC, THM, and—most important for this tutorial book—THC processes.

[1] https://github.com/ufz/ogs/pulls

[2] The well-established variety of models available in OpenGeoSys 5 remains, of course, available.

The central part of OGS-6 is a command-line utility, `ogs`[3], that takes as input (i) a description of the problem[4] to be solved along with (ii) a mesh representing the simulation domain and (iii) a description of the geometrical entities on which boundary conditions are imposed. The problem description is given as an XML file, called "project file", and written with extension `prj` by convention. As mesh format `ogs` accepts unstructured grid files (`vtu`) from the widespread VTK visualization library.[5]And geometries are defined in an XML file with extension `gml`[6].

Unfortunately for beginners and visually oriented people, there is no graphical user interface for OGS-6. There is our own open source visualization software Data Explorer[7] as well as GINA developed by Dr. Herbert Kunz at the Federal Institute for Geosciences and Natural Resources[8], which are able to generate input files for OGS-5, but which have not been adapted to OGS-6, yet. However, our online resources and some tools shipped together with OGS-6 can serve as a starting point for your own numerical experiments. And hopefully this book, too, will be helpful in this regard.

3.1 Getting Started

There is a whole lot of different places where information about OpenGeoSys can be found. Therefore it is necessary to put everything into order, now. First a small comment, where not to look for OGS-6: `http://www.opengeosys.org`.[9] This website mainly deals with the legacy version OGS-5, which is still used by many people. But the main development effort of us is clearly put into OGS-6 now.

[3] In this book `ogs` and OGS-6 are used interchangeably. `ogs` emphasizes the command-line utility and always means OpenGeoSys version 6, never OGS-5!

[4] Usually it's a initial and boundary value problem for some partial differential equation.

[5] `ogs` also accepts OGS-5's legacy mesh format, however, OGS-6 is strict about node ordering. In the case of MPI parallelized simulations `ogs` requires the use of a custom binary mesh format.

[6] This has nothing to do with the Geography Markup Language, though. Furthermore, non-uniform boundary conditions can be specified using `vtu` files.

[7] Which can be downloaded from `https://docs.opengeosys.org/download`.

[8] For this and other tools also see the tutorials previously published in this series.

[9] That holds true as of this writing (autumn 2017) and will change in the future.

3.1.1 Precompiled Binaries

Precompiled binaries of OGS-6 can be downloaded from https://docs.opengeosys.org/download. Both the latest release and "nightly builds"[10] are supported for several operating systems. Since OGS-6 is in continuous development, the latest release might already be a bit dated. So even though the nightly build might be unstable, you might want to use that. And in terms of reliability there is no difference between a development version and a state flagged as "release": The continuous integration checks always run and make it unlikely that new developments break things that worked in the past.[11]

3.1.2 Alternative: Compiling OGS-6 by Yourself

There are many reasons to compile software by yourself: Debugging, "exotic" operating system, or simply the desire for challenges, to name but a few. In any case, you'll need the source code for that. OGS-6 source code can be downloaded from GitHub. Either you clone the Git repository

```
$ git clone https://github.com/ufz/ogs.git
```

or you download the source code of some release from https://github.com/ufz/ogs/releases.

Since OGS-6 depends on several third-party libraries, such as Boost, VTK and Eigen; and since OGS-6 is subject to permanent change steadily and has not reached a fully mature development stage, yet, the build process is also subject to ongoing changes. Basically, building ogs looks like this (in a Linux[12] terminal[13]):

```
$ mkdir build; cd build
$ cmake ../ogs   # ../ogs is the source code directory
$ make
```

Of course, there are much more details to the build process, and probably the "user experience" building OGS-6 will be less straightforward for many new users. However, building OGS-6 yourself is anything but hopeless. There are

[10] Actually it is not built nightly, but after each modification to the Git master branch.

[11] Of course, since many parts of OGS-6 change quickly, you might be forced to update your project files from time to time, but that also holds true if you use switch from one release to another.

[12] Throughout this book, statements involving "Linux" usually also apply to other Unix-like operating systems, e. g., macOS.

[13] The $ symbol denotes the terminal prompt. The author of these lines is a passionate Linux user. Therefore many code examples in this book are soaked with Linux commands. But we guess, clever Windows users will be able to substitute their native commands for those printed here. Anyway, the online installation instructions are maintained for multiple platforms, see also the links on page 48.

up-to-date build instructions for different operating systems available online[14]. They are very detailed and the described build process is a bit fragile. So if you follow them, make sure you follow them very precisely, otherwise the build might fail!

Generally, it has been reported by users that building on Linux is much easier than on Windows, and if your Linux distribution uses sufficiently recent software, maybe you can even fulfil all of OGS-6's prerequisites from your distribution's binary packages[15]. However, we make great efforts to also streamline the build process under Windows, mainly by using the cross-platform C/C++ package manager Conan.

3.2 Running OGS-6's Self Tests

Continuous integration (CI) ensures that every part of OGS-6 stays functional, also in the case of major code changes. CI tests are run on different platforms, e. g., on a self-hosted Jenkins server, and using the Travis CI service. Being an open source project, also our CI tests are open to public inspection.[16] But also every single user can run those CI tests locally on their machines. And although we already tested the code, it is worthwhile that you do the same, because your processor architecture and third-party library versions might be different from ours. For testing, the following commands have to be run from the build directory (assuming that you already ran the commands from Section 3.1.2):

```
bin/testrunner   # runs unit tests
ctest -E LARGE   # runs end—to—end tests
```

The first command runs unit tests that test small, isolated parts of OGS-6's functionality, e. g., if the Gauss-Legendre quadrature works correctly. The second command runs test simulations, checking if the individual T/H/M/C processes implemented are solved within the prescribed error bounds for certain reference problems. The arguments `-E LARGE` exclude tests having the word "LARGE" in their name. If you want to run all tests, just omit them and spend some extra time waiting for `ctest` to finish. If all tests succeeded, you can confidently start working on your own simulations, if something fails, we are happy to receive your bug report on GitHub[17].

[14] As a part of our developer guide at https://docs.opengeosys.org/docs/devguide.

[15] Unfortunately, the current Ubuntu LTS version 16.04 has a VTK package that is too outdated.

[16] E. g., the tests of the master Git branch can be watched at https://jenkins.opengeosys.org/job/ufz/job/ogs/job/master or https://travis-ci.org/ufz/ogs. Badges indicating the CI test status can also be found on our GitHub project page https://github.com/ufz/ogs

[17] We are not happy about the bug, but about any user feedback. Bug reports, feature requests and other issues can be filed at https://github.com/ufz/ogs/issues. Bug

3.3 Pre- and Postprocessing tools

Own simulations necessarily start with data preprocessing: Setting up the mesh and boundary conditions among others. This section very briefly introduces some of the pre- and postprocessing tools of OGS-6. Prebuilt binaries of them can be downloaded from https://docs.opengeosys.org/download. They are included in the Data Explorer downloads, both for the official releases and for the nightly builds. The provided tools cover a broad range from simple mesh generation (`generateStructuredMesh`), over mesh quality checking (`checkMesh`), to several (mesh) file converters (e. g., `OGS2VTK`).

Some of them are documented online[18], but all of them at least print a brief summary of their command-line parameters if they are invoked with the `-h` flag, e. g. (empty lines in the command output are suppressed, here):

```
$ bin/checkMesh -h
USAGE:
   bin/checkMesh  [-p] [-v] [--] [--version] [-h] <string>
Where:
   -p,  --print_properties
     print properties stored in the mesh
   -v,  --validation
     validate the mesh
   --,  --ignore_rest
     Ignores the rest of the labeled arguments following
        ↪ this flag.
   --version
     Displays version information and exits.
   -h,  --help
     Displays usage information and exits.
   <string>
     (required)  input mesh file
   Checks mesh properties
```

Some of those tools are accessible via the Data Explorer GUI. Since OGS-6 uses open and (mostly) standard formats for file IO, of course many other applications can be used for pre- and postprocessing. E. g., simulation results can be visualized by ParaView, and meshes can be generated using Gmsh and converted to the `vtu` format with our `GMSH2VTK` tool, or they can be generated with some CAD tool, such as the free software SALOME, and converted to `vtu` using ParaView, which supports many different formats for reading and writing.

reports should contain the standard minimum information: (a) Which version of OGS-6 was used (i. e., at best the exact Git commit hash)? (b) What is the error (e. g., post some terminal output)? (c) How can the error be reproduced? But note: If you are using an old version of OGS-6, the bug might already have been fixed in the current development state; and sometimes even one month old is old.

[18] At https://docs.opengeosys.org/docs/tools/getting-started/overview and pages linked there.

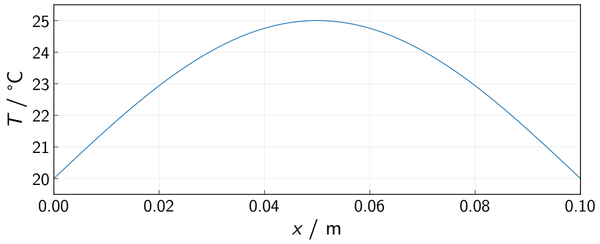


Figure 3.1 Initial temperature profile for the current heat conduction problem.

In this tutorial book, however, no fancy meshes are needed. In fact, the meshes used in the examples discussed can be generated using the `generateStructuredMesh` utility[19] shipped with OGS-6.

We want to close this section with a remark for those compiling OGS-6 by themselves: The mentioned tools are not built by default. If you want to build them, you have to enable the CMake setting `OGS_BUILD_UTILS`, i. e., run

```
cmake -DOGS_BUILD_UTILS=ON .
make
```

in the build directory, or use any CMake GUI of your choice for that. With that said, the time is right for a "Hello World!" example, which will be covered in the following section.

3.4 A first Example: The Heat Equation

As a first example, we consider the heat equation

$$\varrho c_p \frac{\partial T}{\partial t} + \operatorname{div}(\lambda \operatorname{grad} T) = 0\,. \tag{3.1}$$

This equation describes heat conduction in a solid material of density ϱ and with specific isobaric heat capacity c_p and thermal conductivity λ. And T is, of course, the absolute temperature. In order to be more specific, let's imagine a 10 cm long copper rod, which has a sinusoidal initial temperature profile $T_0(x) = 20\,°\mathrm{C} + 5\,°\mathrm{C} \cdot \sin(\pi x / L)$, see Fig. 3.1. Both ends of the rod are kept at the temperature $T_{\mathrm{BC}} = 20\,°\mathrm{C}$. We neglect heat losses over the curved surface of the rod; therefore we can describe it in one spatial dimension.

All important parameters of this example problem are given in Table 3.1. The initial condition (IC) together with the Dirichlet boundary conditions (BC) at the ends of the rod make the definition of the initial boundary value problem (IBVP) complete:

[19] Documented at `https://docs.opengeosys.org/docs/tools/meshing/structured-mesh-generation`.

Table 3.1 Parameters for the example problem of heat conduction in a copper rod.

Length of the rod	L	0.1 m
Density of copper	ϱ	8960 $\mathrm{kg\,m^{-3}}$
Heat capacity of copper	c_p	385 $\mathrm{J\,kg^{-1}\,K^{-1}}$
Thermal conductivity of copper	λ	400 $\mathrm{W\,m^{-1}\,K^{-1}}$
Temperature of the ends of the rod	T_{BC}	20 °C
Initial temperature variation	$\mathit{\Delta} T_{\mathrm{var}}$	5 °C

$$T|_{t=0} = T_{\mathrm{BC}} + \mathit{\Delta} T_{\mathrm{var}} \cdot \sin \frac{\pi x}{L} \qquad\qquad T|_{x=0} = T|_{x=L} = T_{\mathrm{BC}}\,. \tag{3.2}$$

The analytial solution of the IBVP given by Eqs. (3.1) and (3.2) is

$$T(x,t) = T_{\mathrm{BC}} + \mathit{\Delta} T_{\mathrm{var}} \cdot \sin \frac{\pi x}{L} \cdot \exp\left(-\frac{\lambda}{\varrho c_p}\,\frac{\pi^2}{L^2}\,t\right)\,, \tag{3.3}$$

which can be derived, e. g., by separation of variables, or just confirmed by checking if it fulfils Eqs. (3.1) and (3.2).

In order to solve the heat equation numerically with OGS-6, a mesh, a geometry definition and a `prj` file are needed. The following sections will descibe those in detail.

3.4.1 Mesh and Geometry

For the present 1D problem the tools coming with OGS-6 are sufficient to create the mesh, see also Section 3.3. The command[20]

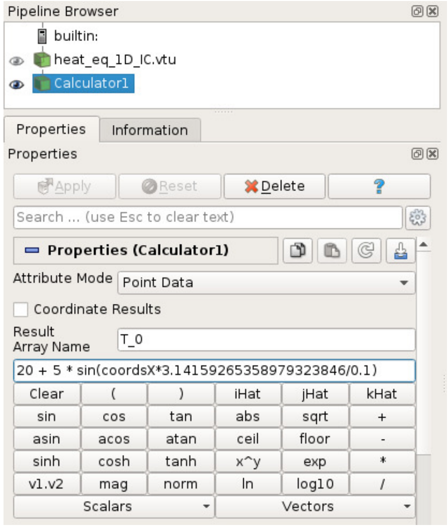
```
generateStructuredMesh -e line --lx 0.1 --nx 50 -o
   ↪ heat_eq_1D_IC.vtu
```

creates a one-dimensional mesh, which extends 0.1 (m) in x-direction[21] and consists of 50 line elements of equal size. The generated mesh is written to the file `heat_eq_1D_IC.vtu`. For our example problem, the initial temperature profile has to be written to the mesh. You can use, e. g., ParaView for that, cf. Fig. 3.2.

Once that is done, only the geometry file is left. For our simple example it is written quickly by hand. It is shown in Listing 3.1. Therein the two ends of the rod are defined, namely `<point>` $(0, 0, 0)$ named `left` and `<point>` $(0.1, 0, 0)$ named `right`. The given `id`s do not matter in this example. In fact, they are only used if one wants to construct lines or surfaces from the points, which we don't need, yet. The tag name `<OpenGeoSysGLI>` derives from the `gli` files used in OGS-5. The name of the entire geometry, `line_geometry`,

[20] If `generateStructuredMesh` is not present in your `$PATH`, as usual the full path to the command must be used instead of just `generateStructuredMesh`. In the following, paths will be omitted without further comment.

[21] From $x = 0$ to $x = 0.1\,\mathrm{m}$, to be precise.



Attribute Mode: Point Data

☐ Coordinate Results

Result Array Name: T_0

20 + 5 * sin(coordsX*3.14159265358979323846/0.1)

Figure 3.2
Compunting the initial temperature profile $T_0(x) = T_{\text{BC}} + \Delta T_{\text{var}} \cdot \sin \frac{\pi x}{L}$ with ParaView.

Listing 3.1 `heat_eq_1D.gml`. Herein the geometry of the heat equation example is defined.

```
<OpenGeoSysGLI>
  <name>line_geometry</name>
  <points>
    <point id="0" x="0"   y="0" z="0" name="left"/>
    <point id="1" x="0.1" y="0" z="0" name="right"/>
  </points>
</OpenGeoSysGLI>
```

is important, because exactly the same name has to be used in the `prj` file. The `prj` file itself will be discussed in the next section.

3.4.2 The Project File

The `prj` file is the pivotal element of OGS-6's inputs. Here, the IBVP being solved is defined along with the solution method used and which output to produce. This section describes the `prj` file of the heat conduction problem in detail. If you don't want to know all the details now, but rather try OGS-6 out first, you can skip to Section 3.4.3 and come back later.

Specification of the IBVP

All elements in the `prj` file can be given in arbitrary order. The order of the example file presented was chosen such that all options can be explained most easily. The first part of the `prj` file covers the specification of the heat conduction problem to be solved. To begin with, the `<mesh>` and `<geometry>` files created in the preceding section are referenced:

```
<OpenGeoSysProject>
  <mesh>heat_eq_1D_IC.vtu</mesh>
  <geometry>heat_eq_1D.gml</geometry>
```

Both file names are treated as paths relative to the directory the `prj` file is placed in. I. e., if only the file name is given, the `vtu` and the `gml` file have to be in the same directory as the `prj` file.

Next, the PDE being solved is defined. In OGS-6 all PDEs are implemented as so-called “processes”, since they correspond to some physical process taking place. In principle, several `<processes>` can be defined, which can be solved by a staggered coupling scheme, but in this case we only have one `<process>` of `<type>` `HEAT_CONDUCTION`:

```
  <processes>
    <process>
      <name>HeatConduction</name>
      <type>HEAT_CONDUCTION</type>

      <integration_order>2</integration_order>

      <!-- Physical parameters of the heat equation. -->
      <thermal_conductivity>lambda</thermal_conductivity>
      <heat_capacity>c_p</heat_capacity>
      <density>rho</density>

      <process_variables>
        <process_variable>temperature</process_variable>
      </process_variables>

      <secondary_variables>
        <secondary_variable type="static"
           ↪ internal_name="heat_flux_x"
           ↪ output_name="heat_flux"/>
      </secondary_variables>
    </process>
  </processes>
```

The process can be referenced by its `<name>` `HeatConduction`[22] in other parts of the `prj` file. You’ll see an application of that soon. The `<integration_order>` of `2` determines the number of integration points used during the assembly of

[22] Note that everything in the `prj` file is case-sensitive. Moreover, all XML tags (except the top-level `<OpenGeoSysProject>`) are written lower-case with underscores separating words.

the FEM equation systems. For the line elements used here, the order of 2 also yields 2 integration points[23], which is usually sufficient[24].

The physical parameters of the heat equation are the <thermal_conductivity>, the <heat_capacity> and the <density> of the solid. The given values lambda, c_p and rho are the names of parameters[25] defined subsequently. These names can be chosen by the user. The tag names, however, are defined in the OGS-6 source code.[26]

The heat equation is solved for one unknown physical field: the temperature. Here, we also chose to call the field temperature. This, again, only references a <process_variable> defined further down in the prj file. Finally, a secondary variable is defined: The conductive heat flux in x-direction. In OGS-6 it has the internal name heat_flux_x, but we want to call it simply heat_flux in the output files. The XML attribute type="static" is mandatory, but will not be explained now.

In the following section of the prj file, actual parameters are defined:

```
<parameters>
  <parameter>
    <name>lambda</name>
    <type>Constant</type>
    <value>400</value>
  </parameter>
  <parameter>
    <name>c_p</name>
    <type>Constant</type>
    <value>385</value>
  </parameter>
  <parameter>
    <name>rho</name>
    <type>Constant</type>
    <value>8960</value>
  </parameter>
  <parameter>
    <name>T_IC</name>
    <type>MeshNode</type>
    <field_name>T_0
      ↪ </field_name>
  </parameter>
  <parameter>
    <name>T_BC</name>
    <type>Constant</type>
    <value>20</value>
  </parameter>
</parameters>
```

lambda, c_p, rho and T_BC are Constant parameters—i. e., they are constant in time and uniform in space—having values as defined in Table 3.1 for λ, c_p, ϱ, and T_{BC}, respectively. The former three of them are referenced in the preceding section by <name>. T_IC will be used as initial temperature subsequently. Therefore it is defined as a MeshNode parameter. I. e., at each node of the mesh a different value is given for it. Those values will be read from

[23] For other kinds of elements, the corresponding number of integration points differs. E. g., for quadrilateral elements an order of 2 means 4 integration points, for hexahedra 8 integration points.

[24] For linear (not to be confused with “line”) elements. For higher-order shape functions, also higher-order integration should be used. Consult your favourite FE book regarding details.

[25] Unfortunately the same word is used for the parameters of the PDE and OGS-6’s internal parameters. In OGS-6 a parameter is an arbitrary function of space and time, which can be—among others—spatially (non-)uniform and time (in-)dependent. Until now, we didn’t find a better name for that concept.

[26] That holds true for all tag names: They are never user-defined.

the nodal field T_0 in the mesh file heat_eq_1D_IC.vtu that we computed with ParaView in the previous section.

For the IBVP to be completely described, the initial and boundary conditions have to specified. For that we set up our <process_variable> temperature, which already has been referenced by that <name> earlier:

```
    <process_variables>
      <process_variable>
        <name>temperature</name>

        <components>1</components>
        <order>1</order>

        <initial_condition>T_IC</initial_condition>

        <boundary_conditions>
          <boundary_condition>
            <geometrical_set>line_geometry</geometrical_set>
            <geometry>left</geometry>
            <type>Dirichlet</type>
            <parameter>T_BC</parameter>
          </boundary_condition>
          <boundary_condition>
            <geometrical_set>line_geometry</geometrical_set>
            <geometry>right</geometry>
            <type>Dirichlet</type>
            <parameter>T_BC</parameter>
          </boundary_condition>
        </boundary_conditions>
      </process_variable>
    </process_variables>
```

Being a scalar field, the temperature has 1 <components>. Furthermore, it is linearly interpolated between the mesh nodes, i. e., the shape function <order> is 1.[27] The <initial_condition> is read from the previously defined parameter T_IC. Eventually, the two Dirichlet <boundary_conditions> are set: One at the left and one at the right end of the domain, where those two geometrical entities are defined in the gml file. Both ends of the domain are held at the constant temperature given by the T_BC parameter just defined. Note that the name of the <geometrical_set> has to match the <name> set in the gml file!

Solver and Output Settings

The remainder of the prj file, and of this section, is concerned with the solver settings and with the settings for generating output:

[27] OGS-6 also supports quadratic shape functions. If you want to use them, you'll need a corresponding mesh containing also higher-order nodes. If you do so, you might have to use a higher <integration_order>, too.

```
    <time_loop>
      <processes>
        <process ref="HeatConduction">
          <time_discretization>
            <type>BackwardEuler</type>
          </time_discretization>

          <nonlinear_solver>basic_picard</nonlinear_solver>
          <convergence_criterion>
            <type>DeltaX</type>
            <norm_type>NORM2</norm_type>
            <abstol>1.e-6</abstol>
          </convergence_criterion>
```

Applying the finite element method discretizes the PDE (3.1) in space, yielding an ODE system. In the HeatConduction <process> we want to discretize this ODE system in time using the BackwardEuler <time_discretization>. The nonlinear equation systems obtained from this discretization, in turn, shall be solved by the <nonlinear_solver> named basic_picard, which will be defined later on. The nonlinear solver is considered to have converged, once the ℓ_2-norm (NORM2) of the difference (DeltaX) of the solutions $T_i - T_{i-1}$ between two subsequent iterations $i - 1$ and i is smaller than 10^{-6} [28.

The following listing defines the time steps made in the solution of the PDE:

```
          <time_stepping>
            <type>FixedTimeStepping</type>
            <t_initial>0.0</t_initial>
            <t_end>60</t_end>
            <timesteps>
              <pair><repeat>50</repeat><delta_t>0.1</delta_t>
                  ↪ </pair>
              <pair><repeat>50</repeat><delta_t>0.2</delta_t>
                  ↪ </pair>
              <pair><repeat>20</repeat><delta_t>0.5</delta_t>
                  ↪ </pair>
            </timesteps>
          </time_stepping>
```

A scheme making a time steps of predifined size (FixedTimeStepping) is used. The simulation starts at <t_initial> = 0.0 s and ends at <t_end> = 60 s. At first, 50 time steps of size <delta_t> = 0.1 s are made, then, 50 steps of <delta_t> = 0.2 s, and finally 20 steps of <delta_t> = 0.5 s.

Alright? … No, obviously not! $50 \cdot 0.1\,\mathrm{s} + 50 \cdot 0.2\,\mathrm{s} + 20 \cdot 0.5\,\mathrm{s} = 25\,\mathrm{s} < 60\,\mathrm{s}$. What about the missing 35 s? Are they dropped? No, the specified <t_end> "wins". So all 60 s will be simulated, and the last time step size is repeated

[28] I. e., $1.0 \cdot 10^{-6}$ K in this case, because there is only one primary variable in the solution vector to the HEAT_CONDUCTION process. If there are several primary variables, each being a different physical quantity, one probably should use a different <convergence_criterion>.

to fill the whole simulation time. I. e., the taken time steps will be at $t = 0.1, 0.2, \dots, 4.9, 5; 5.2, 5.4, \dots, 14.8, 15.0; 15.5, 16.0, \dots, 59.5, 60\,\mathrm{s}$. Likewise, if the specified time steps add up to more than $\Delta t :=$ <t_end> − <t_initial>, the last time steps are dropped in order to fit the time span Δt. Furthermore, if simple adding or dropping of time steps of the predefined size cannot fit Δt, the last time step will be truncated, s. t. it fits Δt exactly.

The last process-specific setting is, which of its <variables>, primary or secondary, to output. Obviously we are interested in the temperature, and earlier we also defined the heat_flux. So we want both to be written to the simulation results:

```
          <output>
            <variables>
              <variable>temperature</variable>
              <variable>heat_flux</variable>
            </variables>
          </output>
        </process>
      </processes>
```

The general output configuration is separated from the process-specific one. That might not be very convenient, but it is the *status quo* as of autumn 2017. Maybe this part of the prj file will change, soon. Anyway, we want output to be written to some VTK file format—it will be written to a series of vtu files "held together" by a pvd file (ParaView Data):

```
      <output>
        <type>VTK</type>
        <prefix>heat_eq_1D</prefix>
        <timesteps>
          <pair><repeat>15</repeat>
                <each_steps>10</each_steps></pair>
          <pair><repeat>1</repeat>
                <each_steps>20</each_steps></pair>
        </timesteps>
      </output>
    </time_loop>
```

The file names will be <prefix>ed with heat_eq_1D, the rest of the file name will contain the process number, and for the vtu files additionally the time step number and the current simulation time. In <timesteps> it is specified at which time steps output files are written. The setting works similar to the FixedTimeStepping configuration: Here, 15 times output is written at every 10th time step. After that, 1 time output is written at every 20th time step; but again, the last value is repeated until the end of the simulation. I. e., with the given <timesteps> here and in <time_stepping>, output will be generated at $t = 0; 1, 2, 3, 4, 5; 7, 9, 11, 13, 15; 20, 25, 30, 35, 40; 50$, and $60\,\mathrm{s}$.[29] Obviously this setting is rather unintuitive to use, and the output times

[29] OGS-6 always generates output at <t_initial> and at <t_end>.

would have been even more “unpredictable”, if we had chosen to write output, e. g., every third time step. Therefore you are advised that the settings

```
<pair><repeat> i </repeat><delta_t> Δt </delta_t></pair> and
<pair><repeat> j </repeat><each_steps> k </each_steps></pair>
```

match up for each such line you specify, i. e., $i = j \cdot k$ in this example.

Finally, the nonlinear solver, named basic_picard, is defined:

```
    <nonlinear_solvers>
      <nonlinear_solver>
        <name>basic_picard</name>
        <type>Picard</type>
        <max_iter>10</max_iter>
        <linear_solver>general_linear_solver</linear_solver>
      </nonlinear_solver>
    </nonlinear_solvers>
```

This one uses a fixed-point iteration scheme (Picard) to solve the nonlinear equation system arising from FEM spatial and BackwardEuler temporal discretization. The solver iterates at most 10 times to satisfy the <convergence_criterion> imposed earlier. If that number of iterations is not sufficient, the simulation fails and OGS-6 will abort. The linear equation systems that have to be solved in each iteration of the nonlinear solver, in turn, will be solved by the <linear_solver> named general_linear_solver, which itself is defined as:

```
    <linear_solvers>
      <linear_solver>
        <name>general_linear_solver</name>
        <eigen>
          <solver_type>CG</solver_type>
          <precon_type>DIAGONAL</precon_type>
          <max_iteration_step>10000</max_iteration_step>
          <error_tolerance>1e-16</error_tolerance>
        </eigen>
      </linear_solver>
    </linear_solvers>
  </OpenGeoSysProject>
```

It uses an iterative conjugate gradient (CG) solver[30] from the Eigen library, together with a DIAGONAL <preconditioner>. The maximum number of

[30] CG solvers can only solve linear equation systems $Ax = b$ if the matrix A is symmetric. The heat equation gives rise to a symmetric matrix after linearization.

iterations to solve the linear system is 10 000 [31, and the error_tolerance of the linear solver is 10^{-16} [32, which can be considered as rather strict.

Remarks

That concludes the description of OGS-6's input files. Finally, we want to summarize certain general properties of the prj file format, and of how OGS-6 reads it. Some of them have already been mentioned in the preceding sections.

- Everything is case sensitive, i. e., XML tags and attributes as well as their content.
- XML tags and attributes, except the top-level tags, are lower-case, and use underscores to separate individual words. E. g.: <linear_solver>.
- Type checking is strict: If OGS-6 expects a number somewhere, and it gets, e. g., 3.141e2x, an error message will be generated, and OGS-6 will abort. E. g.:

  ```
  error: ConfigTree: In file 'heat_eq_1D.prj' at path
     ↪ <parameters/parameter/value>: Value '3.141e2x' is
     ↪ not convertible to the desired type. at
     ↪ ConfigTree.cpp, line 230
  ```

- All XML tags (and attributes) used in the prj file must be processed by OGS-6. That means (i) it is an error to use entirely unknown tags, such as, e. g., <xyz />:

  ```
  error: ConfigTree: In file 'heat_eq_1D.prj' at path
     ↪ <processes/process>: Key <xyz> has been read 1
     ↪ time(s) less than it was present in the
     ↪ configuration tree. at ConfigTree.cpp, line 230
  ```

  (ii) it is an error to write a tag more often than necessary, e. g., specifying a <type> twice for the same thing; and (iii) it is an error to use tags that have no meaning in the current context, e. g., if you specify a <field_name> for a Constant <parameter>.
- There are only very few optional parameters. Therefore almost every setting taken is immediately visible from the prj file. Furthermore, if you accidentally forget to provide some setting, it probably will be an error. E. g., if you don't specify the maximum number of iterations of the nonlinear solver:

[31] In the current example 10 000 iterations are much more than sufficient: There are 51 nodes in the mesh, so the linear equation system has 51 unknowns (in fact, two of them are already determined by the Dirichlet BCs). Therefore, after at most 50 iterations a Krylov subspace solver, such as CG, must have produced the exact solution of the linear equation system (if one neglects round-off errors in the CPU).

[32] Note: Different linear solvers might use different error criteria that might be, additionally, influenced by used norm and preconditioner settings. When in doubt, you should consult the manual of the linear solver library you use.

```
error: ConfigTree: In file ‘heat_eq_1D.prj’ at path
   ↪ <nonlinear_solvers/nonlinear_solver>: Key
   ↪ <max_iter> has not been found at ConfigTree.cpp,
   ↪ line 230
```

We believe that the behaviour just described makes it more unlikely that users who accidentally introduce errors into their input files, e. g., by copying from other pre-existing files, will not be notified of those errors: OGS-6 won't just silently accept things and run and produce results—results that are only "slightly wrong", in the worst case, such that nobody notices.

3.4.3 Running the Heat Conduction Simulation

Eventually, we can try out our new "toy". Let's make a dry run first:

```
$ ogs --version

ogs  version: 6.0.8-1555-g6f17352ef
```

Hooray, it works! If it didn't work for you, you can ask us for help at the OpenGeoSys mailing list or you can open an issue on our GitHub page; links to both are provided on page 48.[33] Now we can do something more serious:

```
$ ogs -o out heat_eq_1D.prj
```

This will run the example problem we just set up. All output files will be written to the directory `out`, which must already exist:

```
$ ls out | sort -V
heat_eq_1D_pcs_0.pvd
heat_eq_1D_pcs_0_ts_0_t_0.000000.vtu
heat_eq_1D_pcs_0_ts_1_t_1.000000.vtu
...
heat_eq_1D_pcs_0_ts_19_t_60.000000.vtu
```

As already mentioned in the preceding section, the `pvd` file can be used to conveniently postprocess the simulation results with ParaView.

The temperature profiles arising from the IBVP Eq. (3.1) & (3.2) are plotted in Fig. 3.3. As we already saw in the analytical solution Eq. (3.3), the temperature profile retains its sinusoidal shape, and—of course—decays over time. Furthermore, the difference between the analytical and numerical solution is largest in the center of the rod, and zero—due to the Dirichlet BC—at the boundary.

Figure 3.4 depicts the temporal evolution of the temperature in the center of the rod. The exponential decay of T becomes clear, and the fact that

[33] If you ever face a problem with OGS-6, we ask you to provide the OGS-6 version number in the problem description that you'll, hopefully, send to us. I. e., `6.0.8-1555-g6f17352ef` in the current example.

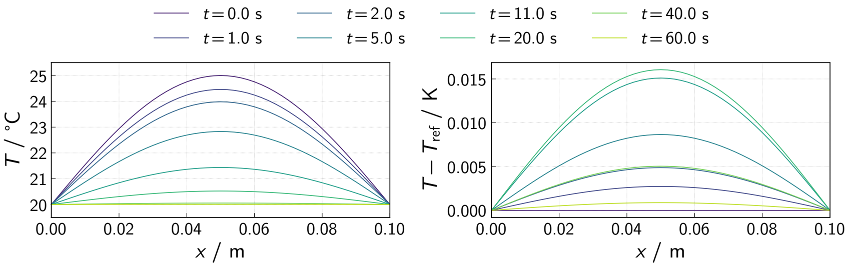


Figure 3.3 *Left*: Temperature profiles in the copper rod computed with OGS-6 for different time steps t. *Right*: Difference between the numerical solution T and the analytical solutions T_{ref} for different time steps.

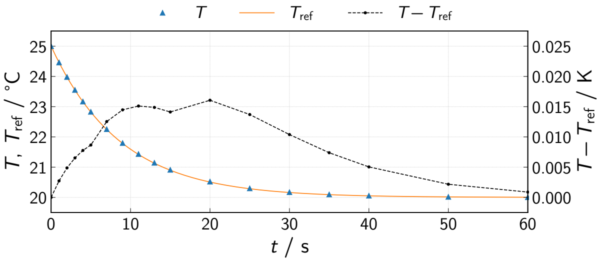


Figure 3.4 Temperature evolution over time at the center of the rod. Numerically and analytically obtained solutions, T and T_{ref}, and their difference are plotted.

errors in the numerical solution procedure first accumulate, culminating at $t = 10 \dots 20$ s, and then decay again, compared to the analytical results, as T approaches the steady-state temperature.

That concludes the heat conduction example. Now you can play around with the settings, and verify that the `BackwardEuler` method is a first order time discretization scheme, or that the spatial discretization is of second order in the spatial resolution of the grid. Of course, you can also go ahead to something more relevant for applications, e. g., to Chapter 4. Or you continue reading this introductory chapter: The following section covers the online documentation of the `gml` and `prj` files.

3.5 Input File Documentation

The first example has been worked through, and can serve as a starting point for your own endeavours. But only a small fraction of OGS-6's features has been discussed therein. More "template" problems can be found in our test suite. They test every process implemented in OGS-6, and are located in the `Tests/Data` directory in OGS-6's source code, or—equivivalently—at `https://github.com/ufz/ogs-data`.

Listing 3.2 Parameter configuration in a `prj` file. This listing shows a part of the file discussed in Section 3.4.2.

```
<OpenGeoSysProject>
  ...
  <parameters>
    ...
    <parameter>
      <name>rho</name>
      <type>Constant</type>
      <value>8960</value>
    </parameter>
    <parameter>
      <name>T_IC</name>
      <type>MeshNode</type>
      <field_name>T_0</field_name>
    </parameter>
  </parameters>
  ...
</OpenGeoSysProject>
```

But not every feature of OGS-6 appears in those test cases: E. g., it would be extremely labourious to provide them for every single material model. Hence, a full list of features can only be found in the online documentation,[34] which will be briefly described now. A manual that has a "how to read this manual" is not perfect, yet. because obviously it is not very intuitive. Unfortunately that applies to our input file documentation: This section is exactly a manual for the manual.

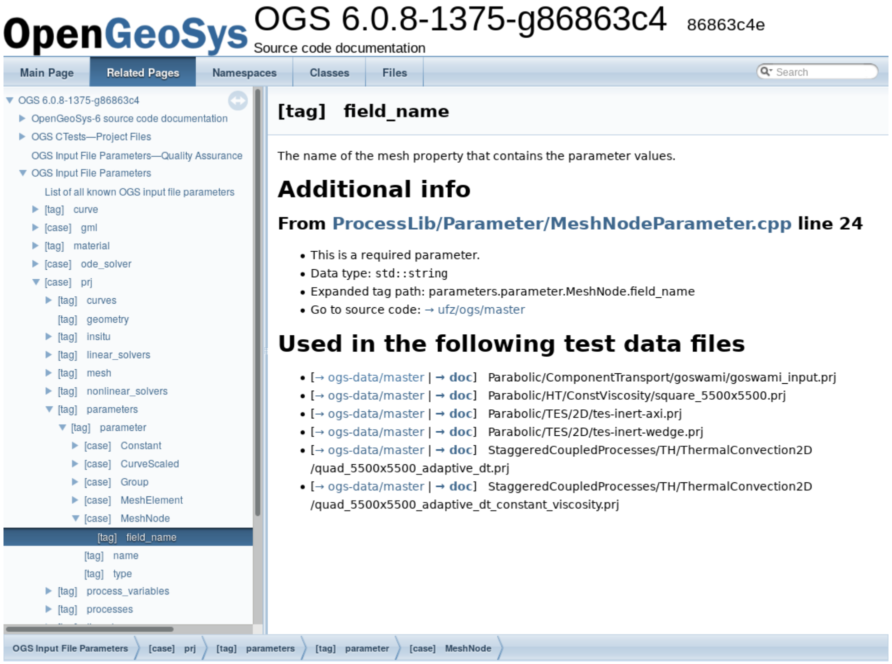
In Listing 3.2 you see a part of the parameter configuration in the `prj` file, which has already been discussed on page 38. The corresponding online documentation is displayed in Fig. 3.5. In the following, Listing 3.2 will be used as an example to explain Fig. 3.5.

`[tag]`, `[attr]` and `[case]` Pages

In the navigation section on the left of Fig. 3.5 there is a tree structure rooted at `[case] prj` that resembles the tree structure of the `prj` file. In the navigation tree there are three different kinds of items: `[tag]`, `[attr]` (not shown in the figure), and `[case]`. For the former two there is a one-to-one correspondence between the documentation page and an XML tag or attribute in the `prj` file. E. g., the page `[tag] parameters` → `[tag] parameter` → `[tag] name` directly corresponds to the setting at `<OpenGeoSysProject><parameters><parameter><name>`.

The `[case]` pages, however, need a bit more explanation. Consider the `<name>` tags in Listing 3.2. They are common to all kinds of parameters. Therefore there documentation is located at `[tag] parameters` → `[tag] parameter` → `[tag] name` in Fig. 3.5. Apart from that, the `<field_name>` is a special setting that applies to parameters of `<type>` `MeshNode` only. Therefore it is not documented directly beneath `[tag] parameter`, but there is a `[case] MeshNode` interposed in between.

[34] At `https://doxygen.opengeosys.org/ogs_file_param__ProjectFile.html`. If you have doxygen, and optionally Python, installed, you can also generate the documentation by yourself: `make doc` will put it in the `docs` folder in the build directory.



[tag] field_name

The name of the mesh property that contains the parameter values.

Additional info

From ProcessLib/Parameter/MeshNodeParameter.cpp line 24

- This is a required parameter.
- Data type: `std::string`
- Expanded tag path: parameters.parameter.MeshNode.field_name
- Go to source code: → ufz/ogs/master

Used in the following test data files

- [→ ogs-data/master | → doc] Parabolic/ComponentTransport/goswami/goswami_input.prj
- [→ ogs-data/master | → doc] Parabolic/HT/ConstViscosity/square_5500x5500.prj
- [→ ogs-data/master | → doc] Parabolic/TES/2D/tes-inert-axi.prj
- [→ ogs-data/master | → doc] Parabolic/TES/2D/tes-inert-wedge.prj
- [→ ogs-data/master | → doc] StaggeredCoupledProcesses/TH/ThermalConvection2D /quad_5500x5500_adaptive_dt.prj
- [→ ogs-data/master | → doc] StaggeredCoupledProcesses/TH/ThermalConvection2D /quad_5500x5500_adaptive_dt_constant_viscosity.prj

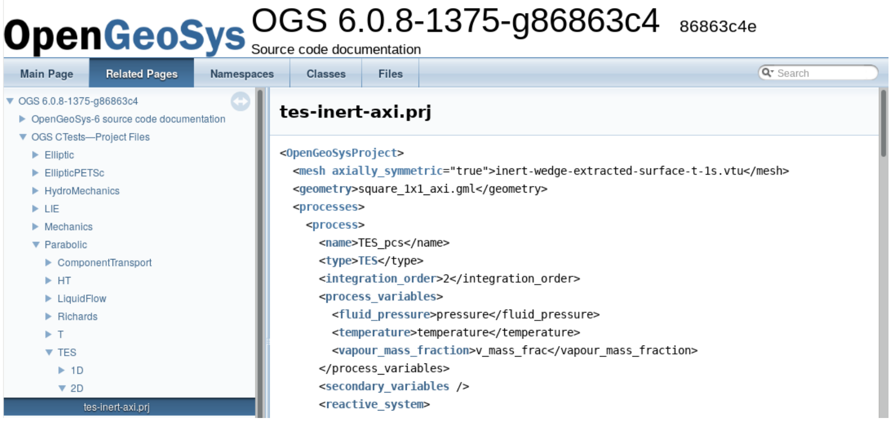
Figure 3.5 Online documentation of OGS-6's input files.

Structure of the Individual Documentation Pages

Each documentation page starts with a hand-written section explaining the meaning of the particular setting. Usually, this section is rather short—We are better at automating things, than doing them manually `:-)`. The rest of the page is autogenerated, and will tell you:

- Where the setting is read in the OGS-6 source code, including links to documentation and the very location in the code.
- If the setting has to be specified in the `prj` file.
- And of which data type the value of the setting is.

Finally, a list of test cases, where the specific setting is used, is given. There are both links to the plain `prj` files in the OGS-6 data repository, and to an enriched, cross-linked version located in the doxygen documentation, cf. Fig. 3.6. Of course, there is such a cross-linked page for every `prj` file of every single test case. If you want to browse through them, the starting point is `https://doxygen.opengeosys.org/ogs_ctest_prj____.html`.



tes-inert-axi.prj

```
<OpenGeoSysProject>
  <mesh axially_symmetric="true">inert-wedge-extracted-surface-t-1s.vtu</mesh>
  <geometry>square_1x1_axi.gml</geometry>
  <processes>
    <process>
      <name>TES_pcs</name>
      <type>TES</type>
      <integration_order>2</integration_order>
      <process_variables>
        <fluid_pressure>pressure</fluid_pressure>
        <temperature>temperature</temperature>
        <vapour_mass_fraction>v_mass_frac</vapour_mass_fraction>
      </process_variables>
      <secondary_variables />
      <reactive_system>
```

Figure 3.6 Cross-linked prj file of a test case: The TES process on an axially symmetric domain is tested. Everything typeset in bold-face and blue is a link to the corresponding prj file documentation page.

Links

The following list concludes this introduction to OGS-6. Most of the links have already been mentioned in this chapter. The list might help to quickly find the right (and up-to-date) documentation resources.

- Hub for any kind of documentation related to OGS-6:
  https://docs.opengeosys.org/docs
- Compilation instructions:
  https://docs.opengeosys.org/docs/devguide
- Documentation for pre- and postprocessing tools shipped with OGS-6:
  https://docs.opengeosys.org/docs/tools
- Download location for OGS-6, tools and Data Explorer binaries:
  https://docs.opengeosys.org/download
- Source code documentation (and a bit beyond) for OGS-6:
  https://doxygen.opengeosys.org
- Cross-linked prj files of OGS-6's test cases:
  https://doxygen.opengeosys.org/ogs_ctest_prj____.html
- prj file documentation:
  https://doxygen.opengeosys.org/ogs_file_param__ProjectFile.html
- GitHub page of OGS-6:
  https://github.com/ufz/ogs
- Input files and reference results of OGS-6's test cases:
  https://github.com/ufz/ogs-data
- Subscribing to the OpenGeoSys users mailing list.:
  http://groups.google.com/group/ogs-users/subscribe

Chapter 4

Laboratory-Scale Adsorption Chamber Simulation

In this chapter, the first real-world scenario is simulated using the model developed in Chapter 2: Water vapour adsorption in a lab-scale fixed bed reactor based on a scenario in [Met+14]. Water vapour is flown through a bed of binderless zeolite beads of brand Köstrolith® 13XBFK. The sorption chamber is sketched in Fig. 4.1. The aim is to simulate the water vapour breakthrough curves and the temperature evolution in the zeolite bed. Compared to the simulation procedure in article [Met+14], some simplifications have to be made, s. t. this task can be tackled with OGS-6:

- Currently it is not possible to couple processes on different domains on the domain boundaries. Therefore it is not possible to simulate the heat conduction in the reactor wall explicitly in a clean way. Rather, an effective Robin boundary condition will be used at the reactor wall.
- In [Met+14] an extended Brinkman equation is used as fluid momentum balance. In OGS-6, however, we use a simpler model that does not involve the fluid velocity as primary variable. As a consequence, in this example

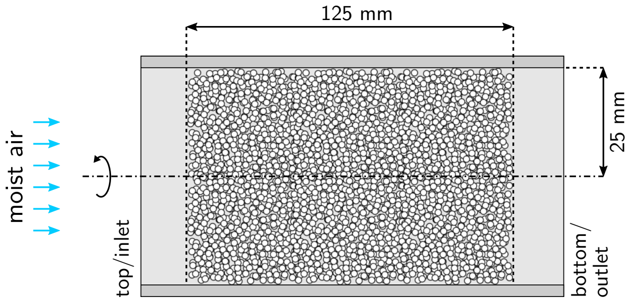


Figure 4.1
Scheme of the sorption chamber that will be simulated in the present chapter.

Table 4.1 Boundary conditions. Empty table cells denote implied zero-flux Neumann BCs that don't have to be specified explicitly in the FEM. The same pattern of BCs is applied in all examples discussed in this book.

Unknown	**Inlet**	**Outlet**	**Wall**
p	Neumann $\dot{\boldsymbol{m}} \cdot \boldsymbol{n} = \dot{m}_{\text{in}}$	Dirichlet $p = p_{\text{out}}$	
T	Dirichlet $T = T_{\text{in}}$		Robin $\lambda \operatorname{grad} T \cdot \boldsymbol{n} = h_{\text{wall}} \cdot (T_{\text{amb}} - T)$
$x_{m\text{V}}$	Dirichlet $x_{m\text{V}} = x_{m\text{V, in}}$		

the radial porosity profile inside the reactor will be neglected, leading to a plug flow through the reactor[1].

Before the adsorption runs the zeolite beads have been desorbed at 180 °C and a water vapour pressure of 1000 Pa. Given those values one can read the initial loading of the zeolite bed from the adsorption isotherms Fig. 2.4, or compute it, e. g., using the characteristic curve. During the adsorption phase, air with different humidity values at different inlet temperatures is flown through the reactor. For now, we choose an inlet temperature of $T_{\text{in}} = 30\,°\text{C}$ and a vapour partial pressure of $p_{\text{V}} = 3000\,\text{Pa}$ as operating conditions. The ambient temperature is $T_{\text{amb}} = 20\,°\text{C}$ and, since the reactor is operated in open mode, the air pressure at the outlet is equal to the ambient pressure of $p_{\text{out}} = 1.013\,25 \cdot 10^5\,\text{Pa}$. The boundary conditions are summarized in Table 4.1.

In the following two sections the mesh and geometry for the simulation will be generated, and afterwards the project file for OGS-6 will be explained. Only the parts differing from the heat conduction example in Chapter 3 will be treated in detail. Finally the obtained simulation results will be discussed.

4.1 Mesh and Geometry

Like in the heat conduction example in Section 3.4, at first we generate the mesh and write the `gml` file. The bed has a radius of 25 mm and a height of 125 mm. Since the domain is axially symmetric, and our model equations and all boundary conditions that will be applied are, too, we expect the solution to also exhibit that symmetry. We can exploit that fact by providing a two-dimensional mesh only, and tell OGS-6 about the symmetry properties.

[1] Except for the variations in temperature and pressure, arising from the nonisothermal formulation and the adsorption processes, that can lead to variations of the fluid velocity across the bed cross-section.

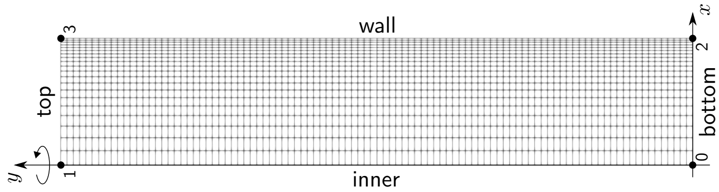


Figure 4.2 The mesh that has been generated for the sorption chamber. The element extension in x direction decreases from the inner boundary towards the wall. Numbers indicate the point ids used in the `gml` file. The labels are the names given to the respective (poly-)line geometries. The problem is considered to be axially symmetric about the y axis.

The aspect ratio of the two-dimensional domain is 1 : 5, so we take the same ratio for the number of elements in x and y direction. Furthermore, we expect that there are rather steep temperature gradients near the chamber wall. Therefore, we want the mesh to be finer there. That can be achieved with the `--mx factor` parameter of the `generateStructuredMesh` utility that causes the size of the $(n+1)^{\text{th}}$ element in x direction to be `factor` times as big as the n^{th} element. The utility figures out the element sizes automatically based on the number of elements and extension in x direction and the value of `factor`. Altogether:

```
$ generateStructuredMesh -e quad --lx 0.025 --ly 0.125
    ↪ --nx 20 --ny 100 --mx 0.9 -o sorption.vtu
info: Mesh created: 2121 nodes, 2000 elements.
```

The resulting mesh together with some `gml` file related information is shown in Fig. 4.2. Note that the mesh must lie in the x-y plane, and that the axial symmetry will be around the y axis. I. e., the relation between the coordinates in OGS-6 and the usual cylindrical coordinates (r, φ, z) is $x_{\text{OGS-6}} \mapsto r$ and $y_{\text{OGS-6}} \mapsto z$.

The corresponding `gml` file is presented in Listing 4.1. At first, the set of `<points>` spanning the two-dimensional domain is defined. Afterwards, the `<polylines>` representing the boundaries of the domain are constructed referring to those points by their `id`. The order of the points/ids does not matter.

4.2 Project File and Parameters

In this section, we explain new settings in the `prj` file, i. e., those settings already discussed in Section 3.4.2 will be omitted here. An indicator of such an omission is that the line numbers between two successive listings jump by more than around four. The `<mesh>` now gets the optional, and self-

Listing 4.1 gml file for the mesh shown in Fig. 4.2.

```
<OpenGeoSysGLI>
  <name>geometry</name>
  <points>
    <point id="0" x="0"     y="0"     z="0"/>
    <point id="1" x="0"     y="0.125" z="0"/>
    <point id="2" x="0.025" y="0"     z="0"/>
    <point id="3" x="0.025" y="0.125" z="0"/>
  </points>
  <polylines>
    <polyline id="0" name="inner" ><pnt>0</pnt>
                                   <pnt>1</pnt></polyline>
    <polyline id="1" name="wall"  ><pnt>2</pnt>
                                   <pnt>3</pnt></polyline>
    <polyline id="2" name="bottom"><pnt>0</pnt>
                                   <pnt>2</pnt></polyline>
    <polyline id="3" name="top"   ><pnt>1</pnt>
                                   <pnt>3</pnt></polyline>
  </polylines>
</OpenGeoSysGLI>
```

documenting, attribute axially_symmetric="true". Apart from that, the mesh and geometry specification is already known:

```
<OpenGeoSysProject>
  <mesh axially_symmetric="true"> adsorption.vtu </mesh>
  <geometry> adsorption.gml </geometry>
```

The THC process that we are going to simulate is called inside OGS-6: . . . Surprise! . . . "TES"[2], and and has three primary variables: p, T, $x_{m\mathrm{V}}$.

```
  <processes>
    <process>
      <name>TES_pcs</name>
      <type>TES</type>
      <integration_order>2</integration_order>
      <process_variables>
        <fluid_pressure>p</fluid_pressure>
        <temperature>T</temperature>
        <vapour_mass_fraction>x_mV</vapour_mass_fraction>
      </process_variables>
```

Next, secondary variables are set up. There are quite a few of them: ϱ_{SR}, ϱ_{GR}, $\hat{\varrho}_{\mathrm{SR}}$, $\boldsymbol{v}_{\mathrm{Darcy}}$, $\boldsymbol{q}$, p_{V} etc. Many of them are defined in the "standard way":

```
<secondary_variable type="static"
   ↪ internal_name="solid_density"
   ↪ output_name="solid_density" />
```

[2] For *thermal energy storage*. We thought that thermochemical heat storage is so important, that it must become *the* thermal energy storage process in OGS-6.

But some are novel. E. g., the following code snippet defines the variable vapour_partial_pressure ↦ p_V in the way already known, and after that the variable equilibrium_loading ↦ C_eq:

```
<secondary_variable type="static"
    ↪ internal_name="vapour_partial_pressure"
    ↪ output_name="p_V" />
<secondary_variable type="dynamic"
    ↪ internal_name="equilibrium_loading"
    ↪ output_name="C_eq">
  <plug sink_arg="adsorptive_partial_pressure"
      ↪ source_fct="vapour_partial_pressure" />
  <plug sink_arg="adsorbent_temperature"
      ↪ source_fct="TES_temperature" />
</secondary_variable>
```

The latter is of type="dynamic", and expresses a functional relationship: $C_{\mathrm{eq}} = C_{\mathrm{eq}}(p_{\mathrm{V}}, T)$, or equivalently in a more codelike way

```
equilibrium_loading = equilibrium_loading(
    adsorptive_partial_pressure,
    adsorbent_temperature)
```

and in even more detail:

```
equilibrium_loading = equilibrium_loading(
    adsorptive_partial_pressure ↤ vapour_partial_pressure,
    adsorbent_temperature ↤ TES_temperature)
```

I. e., the <secondary_variable> symbolizes the function ...(..., ...), and the <plug> symbolizes the ... ↤ ... expressions. In this concrete case, the adsorptive_partial_pressure will be the secondary variable vapour_partial_pressure previously defined, and the adsorbent_temperature will be the TES_temperature, which is exactly the primary variable temperature. Note that each source_fct is referenced by its internal_name, not by its output_name.

Most parameters of the constitutive models involved are given as plain double values. They are summarized in Table 4.2 on page 60. This way of setting parameters is subject to change, because the current policy in OGS-6 is that all physical parameters be given as <parameters>, which is much more flexible, albeit also a bit less convenient in the most simple cases. However, also in the TES process some properties are already set in terms of <parameters>, and for the diffusion coefficient of vapour in air we use a full temperature and pressure dependent model:

```
<porosity>porosity</porosity>
<permeability>permeability</permeability>
<diffusion_coefficient>
  <type>WaterNitrogenMarrero</type>
</diffusion_coefficient>
```

Since all parameters of the model are taken to be uniform in space and constant in time, there is nothing new to their configuration. Furthermore, the configuration of the process variables is completely analogue to Section 3.4.2. The boundary conditions are set according to Table 4.1 and Fig. 4.2. We didn't specify a Neumann BC before:

```
<boundary_condition>
  <geometrical_set>geometry</geometrical_set>
  <geometry>top</geometry>
  <type>Neumann</type>
  <parameter>mass_flux_in</parameter>
</boundary_condition>
```

Note that the Neumann BC determines the flux (i. e., the mass flux as set in the `mass_flux_in` `<parameter>` in this case) and not directly the gradient of the primary variable (i. e., $\operatorname{grad} p$). And we did not either have Robin BCs, yet:

```
<boundary_condition>
  <geometrical_set>geometry</geometrical_set>
  <geometry>wall</geometry>
  <type>Robin</type>
  <alpha>heat_transfer_coefficient</alpha>
  <u_0>T_ambient</u_0>
</boundary_condition>
```

`<alpha>` and `<u_0>` again refer to some `<parameters>` by name, and denote the respectice quantities for the general Robin BC of an arbitrary scalar variable u:

$$\langle\text{normal flux}\rangle = \langle\text{flux vector}\rangle \cdot \boldsymbol{n} = \alpha\,(u - u_0) \tag{4.1}$$

In the current case u is the temperature T, furthermore $u_0 = T_{\text{amb}}$, and $\alpha = h$. Using Fourier's law for the heat flux, $\boldsymbol{q} = -\boldsymbol{\lambda} \operatorname{grad} T$, the equation above specializes to the commonly known form of the temperature Robin BC:

$$(\boldsymbol{\lambda} \operatorname{grad} T) \cdot \boldsymbol{n} = h\,(T_{\text{amb}} - T)\,. \tag{4.2}$$

That concludes the specification of the IBVP, and now the right solver settings have to be chosen. Since now we have three primary variables, each of a different magnitude ($p \approx 10^5$ Pa, $T \approx 3 \cdot 10^2$ K, $x_{m\text{V}} \approx 0 \ldots 10^{-1}$, where $x_{m\text{V}}$ will be clearly greater than zero), taking some ℓ_p norm of the whole solution vector (or differences thereof) and applying the error tolerance to that, obviously is not a good idea. Instead, we apply relative tolerances of `10`$^{-6}$ to each of the three unknowns individually, cf. `<type>PerComponentDeltaX</type>`.

```
<time_loop>
  <processes>
    <process ref="TES_pcs">
      <nonlinear_solver>basic_picard</nonlinear_solver>
      <convergence_criterion>
        <type>PerComponentDeltaX</type>
        <norm_type>NORM2</norm_type>
```

```
          <reltols>1.e-6 1.e-6 1.e-6</reltols>
        </convergence_criterion>
```

Since many secondary variables have been defined, of course also the <output> specification will be rather long now. Here, we only want to comment on an additional output feature: The output of extrapolation residuals together with every secondary variable that is being output:

```
      <variable> p_V                     </variable>
      <variable> relative_humidity       </variable>
    </variables>
    <output_extrapolation_residuals>true
       ↪ </output_extrapolation_residuals>
  </output>
```

In fact, whenever you output secondary variables, you should enable that option! Checking the extrapolation residual is especially important in OGS-6. That comes from the extrapolation procedure used: Since many secondary variables are defined intially at the integration points, they have to be processed in order to output them in a form that is digestible by a broad range of postprocessing software. I. e., the integration point values have to be transformed to data defined either per cell or per node. In OGS-6 we selected nodal data for their output. Therefore the integration point data have to be extrapolated to the nodes. That is implemented cellwise by solving a least-squares problem, and is rather efficient. After that first step, however, each node in the mesh will be assigned different values, namely those values coming from the extrapolation of the integration point data of each adjacent cell. We resolve that multivaluedness by averaging over these values, and that introduces a lot of problems. By the nodal average the least-squares property of the cellwise extrapolation is destroyed. A common consequence of that is, that slopes of the extrapolated variables are flatter in the elements at the boundary than in the interior of the mesh. We, the developers of OGS-6 got confused by those extrapolation and averaging results already many times. The bottomline is: **Always compute the extrapolation residual, and check it!** Especially if the results look weird. The residual will give you an estimate of how reliable the extrapolated data are. In fact, we did observe cases, where the residual error was a big as the extrapolated value! This is the status quo, but since we are not happy with that situation, it is subject to change.

Since the emerging linear equation systems are not symmetric in the present case, the CG iterative solver cannot be used for the TES process. Instead we use a BiCGStab solver here together with an incomplete LU decomposition as preconditioner. The configuration is given both for the Eigen and the Lis linear iterative solver libraries. The former is OGS-6's default, the latter has to be installed and configured manually.

```
<linear_solvers>
  <linear_solver>
```

```
    <name>general_linear_solver</name>
    <lis>-i bicgstab -p ilu -tol 1e-14 -maxiter 10000 -print
       ↪ mem</lis>
    <eigen>
      <solver_type>        BiCGSTAB </solver_type>
      <precon_type>        ilu      </precon_type>
      <max_iteration_step> 10000    </max_iteration_step>
      <error_tolerance>    1e-14    </error_tolerance>
    </eigen>
  </linear_solver>
</linear_solvers>
```

That concludes the explanation of the largest part of the `prj` file for the present adsorption simulation. The probably most important part, however, has been skipped: The specification of the adsorption working pair. We come back to it now. The specification of the reaction kinetics reflects the entire complexity of Eqs. (2.36) and (2.37), it is given in Listing 4.2. Let's dissect those over 30 lines. Everything is wrapped into `<reaction_rate>` tags. In OGS-6 the purpose of the `ReactionRate` C++ class is to cope with the strong nonlinearities caused by the sorption process. So it's a purely algorithmical ingredient. However, it acts as a wrapper around the `<reaction_kinetics>`, which is of `<type>` `LinearDrivingForceCoefficientKast`, here. This type exactly corresponds to Eq. (2.36) with D_{eff} provided by Eq. (2.37). Hence, the different parameters of those equations must be specified in the `prj` file: The constants r_{p}, ϕ_{Gp}, τ, M_{V}, and of course the `<diffusion_coefficient>` D_{comb}. Since in our setting, the mean free path of water in air is $\approx 40\,\text{nm}$, but the macropore diameter is $600\,\text{nm}$ [Non16], the regime in the zeolite pellet is such that both Knudsen diffusion and continuum diffusion play a role. This is reflected in the diffusion of `<type>` `GasAndKnudsen`, which computes the combined diffusion coefficient D_{comb} via

$$\frac{1}{D_{\text{comb}}} = \frac{1}{D_{\text{Kn}}} + \frac{1}{D_{\text{cont}}}\,, \tag{4.3}$$

where the continuum diffusion coefficient of water vapour in nitrogen is computed using a formula from [MM72].

Besides the effective diffusion coefficient, the second essential ingredient of the LDF model is the adsorption equilibrium. Here, it is computed using the `DubininPolanyi` theory together with the experimental data named `Z13XBF_Hauer`, i. e., Köstrolith® 13XBFK whose characteristic curve has been generated using the adsorbate density model by Hauer [Hau02]:

$$\varrho_{\text{ads}}(T) = \varrho_{\text{ads}}(T_0)\left[1 - 3.781 \cdot 10^{-4}\,\text{K}^{-1} \cdot (T - T_0)\right] \tag{4.4}$$

with $T_0 = 283.15\,\text{K}$.

The remaining three settings control the reaction rate computation algorithm. The fast adsorption kinetics of zeolite leads to two main problems: (i) it might adsorb so much, that it consumes all vapour leading to negative

Listing 4.2 Specification of the adsorption equilibrium and kinetics inside the `prj` file.

```
<reaction_rate>
  <type>UpdateMultiple</type>

  <reaction_kinetics>
    <type>LinearDrivingForceCoefficientKast</type>

    <pellet_radius>0.001</pellet_radius>
    <pellet_porosity>0.4</pellet_porosity>
    <pellet_tortuosity>4.0</pellet_tortuosity>
    <molar_mass_reactive_component>0.018016
       ↪ </molar_mass_reactive_component>

    <diffusion_coefficient>
      <type>GasAndKnudsen</type>
      <diffusion_coefficient_gas>
        <type>WaterNitrogenMarrero</type>
      </diffusion_coefficient_gas>
      <diffusion_coefficient_knudsen>
        <type>Knudsen</type>
        <pore_diameter>600e-9</pore_diameter>
        <molar_mass>0.018016</molar_mass>
      </diffusion_coefficient_knudsen>
    </diffusion_coefficient>

    <equilibrium>
      <type>DubininPolanyi</type>
      <adsorbent_density_dry>1150 </adsorbent_density_dry>
      <equilibrium_data>Z13XBF_Hauer</equilibrium_data>
    </equilibrium>
  </reaction_kinetics>

  <equilibrium_reaction_vapour_pressure_limit>5
     ↪ </equilibrium_reaction_vapour_pressure_limit>
  <upper_vapour_pressure_limit>1e5
     ↪ </upper_vapour_pressure_limit>
  <maximum_reaction_rate_updates>15
     ↪ </maximum_reaction_rate_updates>
</reaction_rate>
```

values for $x_{m\mathrm{V}}$ in intermediate iteration results. Those negative values then make it impossible to compute a new C_{eq}, and therefore a better estimate for the reaction rate. (ii) for low p_{V} the equilibrium loading is very sensitive to small variations of p_{V}, which leads to oscillatory behaviour of the solution between successive iterations or time steps.

In order to cope with these problems, we implemented an algorithm that determines the reaction rate iteratively. It is shown as pseudo code resembling Python in Listing 4.3, and works as follows: At the beginning of each assembly of the linearized equation system, in each integration point the reaction rate is computed. For that, the following cases are distinguished (the conditions are checked in that order and only the first matching case applies):

1. p_V is zero or negative, or very small ($< p_\mathrm{V,eq,max}$), but with finite reaction rate, or huge ($> p_\mathrm{V,max}$). Then p_V is set to that value, such that $C = C_\mathrm{eq}(p_\mathrm{V}, T)$ holds, and the maximum reaction rate for this time step is reduced (set to the reaction rate from the previous iteration). If the reaction rate is finite, it will be halved. If it is tiny, however, it will be set to some small negative value ($-10^{-5}\,\mathrm{kg\,m^{-3}\,s^{-1}}$), such that p_V can recover to some positive value. Furthermore, in the current time step, the reaction rate computation in the current element is flagged as “damped”.
2. The number of recomputations of the reaction rate exceeds a maximum value ($i_{\mathrm{max},\hat{\varrho}}$). Then no update is performed anymore, in order to let the nonlinear solver converge.
3. There has been no damping in the current time step. Then the new estimate for $\hat{\varrho}_\mathrm{SR}$ is the average over the reaction rates computed in the current time step.
4. There has been damping in the current time step. Then the new estimate for $\hat{\varrho}_\mathrm{SR}$ is also computed by an average. But now the maximum reaction rate allowed for the current time step is used as the new value in that average rather than the reaction rate computed by the kinetics model.

The settings `<equilibrium_reaction_vapour_pressure_limit>`, `<upper_vapour_pressure_limit>` and `<maximum_reaction_rate_updates>` determine the parameters $p_\mathrm{V,eq,max}$, $p_\mathrm{V,max}$ and $i_{\mathrm{max},\hat{\varrho}}$, respectively. Clearly, bigger values for $i_{\mathrm{max},\hat{\varrho}}$ will lead to better estimates for $\hat{\varrho}_\mathrm{SR}$, but will increase the number of nonlinear iterations accordingly.

With that the treatment of the `prj` file is really finished. We didn't give the parameters of the model, yet. The parameters are summarized in Table 4.2, some of them will be explained in more detail now. The solid density of dry adsorbent ϱ_SR is computed via $\varrho_\mathrm{SR} = \varrho_\mathrm{S}/\phi_\mathrm{S}$ with a bulk density of $\varrho_\mathrm{S} = 690\,\mathrm{kg\,m^{-3}}$ and a solid volume fraction of $\phi_\mathrm{S} = 1 - \phi_\mathrm{G} = 0.6$. The latter value is valid for any packed bed of rough or imperfect or polydisperse spheres [Tso13]. The pellet porosity ϕ_Gp and pore diameter d_pore can be obtained from the pore size distribution, e. g., from mercury intrusion porosimetry experiments. The pellet tortuosity just has been set to the value obtained from fitting by Mette [Met14].

The permeability k of the bed has been computed from the ϕ_G and r_p using the Kozeny-Carman equation Eq. (2.19). The tortuosity of the bed has not been explained, yet. It is present in the model implemented in OGS-6 and enhances the diffusion resistance expressed by the term

$$-\operatorname{div}(\varrho_\mathrm{GR} \boldsymbol{D} \operatorname{grad} x_{m\mathrm{V}}) \qquad \rightarrow \qquad -\operatorname{div}(\varrho_\mathrm{GR}\, \tau_\mathrm{bed}\, \boldsymbol{D} \operatorname{grad} x_{m\mathrm{V}}) \tag{4.5}$$

in Eq. (2.24). Since diffusion inside the flowing air is of minor importance here, we set the tortuosity to 1.

Listing 4.3 Pseudo code for the routine computing the reaction rate. Symbols used: n … time step, i … iteration, $q := \hat{\varrho}_{\mathrm{SR}}$, d … Boolean indication whether the reaction rate has been artificially damped by the algorithm.

```
# Called at the beginning of every time step $n$.
def at_begin_timestep():
    ${\varrho_{\mathrm{SR}0}}^n \leftarrow \varrho_{\mathrm{SR}}$  # save solution from previous time step
    ${q_{\mathrm{max},0}}^n \leftarrow \hat{\varrho}_{\mathrm{SR}}$
    $d_0^n \leftarrow$ False

# Called at the beginning of the assembly for each time step $n$
# and each iteration $i$. Computes the reaction rate.
# In: $p_{\mathrm{V}}$, $p$, $T$, $\varrho_{\mathrm{SR}}$, $\Delta t$
# Out: $q := \hat{\varrho}_{\mathrm{SR}}$, $\varrho_{\mathrm{SR}}$, possibly $p_{\mathrm{V}}$
# Internal variables : $d$, $q_{\mathrm{max}}$
# Parameters: $p_{\mathrm{V,eq,max}}$, $p_{\mathrm{V,max}}$, $i_{\mathrm{max},\hat{\varrho}}$
def at_begin_assembly():
    $\langle$reaction damped$\rangle \leftarrow$ False

    if ${p_{\mathrm{V}}}_{i-1}^n \leq 0$
          or $({p_{\mathrm{V}}}_{i-1}^n < p_{\mathrm{V,eq,max}}$ and $|q_{i-1}^n| > 10^{-3}\,\mathrm{kg\,m^{-3}\,s^{-1}})$
          or ${p_{\mathrm{V}}}_{i-1}^n > p_{\mathrm{V,max}}$:
        ${q_{\mathrm{max},i}}^n \leftarrow q_{i-1}^n$  # reduce the maximum reaction rate
        ${p_{\mathrm{V}}}_{i-1}^n \leftarrow p_{\mathrm{V,eq}}(p_{i-1}^n, T_{i-1}^n, {\varrho_{\mathrm{SR}}}_{i-1}^n)$  # set $p_{\mathrm{V}}$ to local equilibrium value
        $\langle$reaction damped$\rangle \leftarrow$ True
        if $|q_{i-1}^n| > 2 \cdot 10^{-5}\,\mathrm{kg\,m^{-3}\,s^{-1}}$:
            $q_i^n \leftarrow q_{i-1}^n/2$  # reduce $\hat{\varrho}_{\mathrm{SR}}$ for the current iteration
        elif $q_{i-1}^n > -10^{-5}\,\mathrm{kg\,m^{-3}\,s^{-1}}$:
            # desorb a tiny bit of vapour to make $p_{\mathrm{V}}$ positive again
            $q_i^n \leftarrow -10^{-5}\,\mathrm{kg\,m^{-3}\,s^{-1}}$
    elif $i > i_{\mathrm{max},\hat{\varrho}}$:
        # don't update anymore to let the nonlinear solver converge
        $q_i^n \leftarrow q_{i-1}^n$
    elif not $d_{i-1}^n$:
        # iterate to get the final reaction rate as an average value over
        # all iterations , new values become less and less important
        $q_i^n \leftarrow \frac{1}{i+1} \cdot \left(i \cdot q_{i-1}^n + \hat{\varrho}_{\mathrm{SR}}({p_{\mathrm{V}}}_{i-1}^n, p_{i-1}^n, T_{i-1}^n, {\varrho_{\mathrm{SR}}}_{i-1}^n)\right)$
    else:
        # damped in this time step, but try to get closer to the maximum
        # rate again, because we might have damped too much
        $q_i^n \leftarrow \frac{1}{i+1} \cdot (i \cdot q_{i-1}^n + {q_{\mathrm{max},i}}^n)$

    ${\varrho_{\mathrm{SR}i}}^n \leftarrow {\varrho_{\mathrm{SR}0}}^n + (t^n - t^{n-1}) \cdot q_i^n$
    $d_i^n \leftarrow d_{i-1}^n$ or $\langle$reaction damped$\rangle$
```

Table 4.2 Material parameters and operation conditions.

Adsorbent and Kinetics Properties			
density of dry adsorbent material	$\varrho_{\mathrm{SR,dry}}$	1150	$\mathrm{kg\,m^{-3}}$
molar mass of water	M_{V}	0.018 016	$\mathrm{kg\,mol^{-1}}$
pellet radius	r_{p}	0.001	m
pellet porosity	ϕ_{Gp}	0.4	
pellet tortuosity	τ_{p}	4.0	
pore diameter	d_{pore}	$600 \cdot 10^{-9}$	m
Other Material and Packed Bed Properties			
bed porosity	ϕ_{G}	0.4	
bed permeability	k	$4.74 \cdot 10^{-9}$	$\mathrm{m^2}$
bed tortuosity	τ_{bed}	1	
heat conductivity of the solid	λ_{S}	0.4	$\mathrm{W\,m^{-1}\,K^{-1}}$
isobaric heat capacity of the solid	$c_{p\mathrm{S}}$	880	$\mathrm{J\,kg^{-1}\,K^{-1}}$
isobaric heat capacity of air	$c_{p\mathrm{G}}$	1012	$\mathrm{J\,kg^{-1}\,K^{-1}}$
Initial Conditions			
initial pressure	p_0	$1.013\,25 \cdot 10^5$	Pa
initial temperature	T_0	303.15	K
initial vapour mass fraction	$x_{m\mathrm{V}0}$	0.019 24	
initial solid density	$\varrho_{\mathrm{SR}0}$	1242	$\mathrm{kg\,m^{-3}}$
Parameters Related to Boundary Conditions			
mass flux density at the inlet	$\dot{m}_{\mathrm{in}}$	0.4244	$\mathrm{kg\,m^{-2}\,s^{-1}}$
temperature at the inlet	T_{in}	303.15	K
vapour mass fraction at the inlet	$x_{m\mathrm{V,\,in}}$	0.019 24	
ambient temperature	T_{amb}	293.15	K
heat transfer coefficient of the wall	h_{wall}	1.4	$\mathrm{W\,m^{-2}\,K}$
pressure at the outlet	p_{out}	$1.013\,25 \cdot 10^5$	Pa

The initial vapour mass fraction in the bed has just been set to the BC value, since it does not influence the simulation results at all: No matter what initial value you take[3], there's only a negligible amount of vapour in the bed compared to the adsorption capacity of the zeolite. The initial solid density ϱ_{SR0} corresponds to an initial loading of $C = 0.08$, which in turn is the equilibrium loading for the chosen p_{V} and T conditions of the preceding desorption, cf. Fig. 2.6. The inlet mass flux density corresponds to a mass flux of $3\,\mathrm{kg\,h^{-1}}$ evenly distributed over the whole sorption chamber cross-section. The inlet vapour mass fraction $x_{m\mathrm{V}}$ can be computed from the inlet vapour partial pressure $p_{\mathrm{V,in}}$ and Dalton's law:

[3] As long as it stays below some sensible limit, of course. The equilibrium vapour pressure (i. e., 100 % relative humidity) is such a sensible upper limit.

$$x_{n\mathrm{V}} = \frac{p_\mathrm{V}}{p} \qquad\qquad x_{m\mathrm{V}} = \frac{M_\mathrm{V}\, x_{n\mathrm{V}}}{M_\mathrm{V}\, x_{n\mathrm{V}} + M_\mathrm{air}\,(1 - x_{n\mathrm{V}})}\,. \tag{4.6}$$

That reflects the importance of choosing primary unknowns that are easy accessible to experimental measurements: Even in the current case, where $x_{m\mathrm{V}}$ can be easily obtained from p_V, the transformation is rather inconvenient. One can easily imagine, that more complex relations will make it much more difficult to actually work with the model. Sometimes a single best choice is not possible, e. g., the humidity is measured with different quantities, like p_V, $x_{m\mathrm{V}}$, or mass of water per mass of dry air.

There is one last important parameter to be discussed: The overall run time of the simulation: As already said, we start with an initial loading of $C_0 = 0.08$, and the equilibrium loading for the chosen values p_in and T_in will be $C_\mathrm{end} = 0.33$, cf. Fig. 2.6. Dividing the corresponding mass of water adsorbed by the inlet mass flux yields a duration of 2640 s required to reach adsorption equilibrium. Since the vapour might break through earlier than that, we add an ample “security supplement”, and set the overall simulation time to one hour. Therewith we are prepared to run the simulation and analyze its results.

4.3 Running and Postprocessing

OGS-6 can be run in the usual way:

```
$ export OMP_NUM_THREADS=1   # optional
$ ogs -o out adsorption.prj
```

However, the first command in the above listing is new. Its purpose is to limit the maximum number of OpenMP threads used in the simulation. The background of that is the following: The linear solvers set up on Page 55 can solve the system using several threads in parallel in order to run faster. That, however, can comes with some problems: (i) The mesh generated in Section 4.1 has only 2121 nodes. Accordingly there are $2121 \times (p, T, x_{m\mathrm{V}}) = 6363$ unknowns in the linear equation systems that are to be solved. That number is not very high, so the linear solver might not profit very much from being run in parallel, anyway.[4] (ii) Using multiple threads, the ability of the linear solver to solve the equation system at all might deteriorate. Since the equation systems emanating from the THC process are hard to solve anyway, using multiple threads can make the solver fail. Then one might observe the behaviour that the linear solver fails for four threads, succeeds with three of them, but fails for two and succeeds for one. If you are confronted with linear solver failure in a multithreaded environment, you might therefore try to limit the number of threads (in addition to or before changing the preconditioner or linear solver).

[4] Notwithstanding the fact that the equation system assembly is not parallelized, yet.

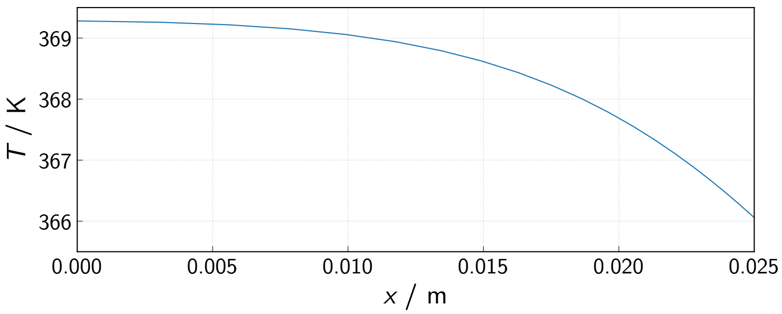


Figure 4.3 Radial temperature profile in the bed at the outlet at $t = 1500\,\mathrm{s}$.

Since this is a text dealing with a piece of software, among others, we are obliged to tell you, that you might want to grab a cup of coffee after you have hit the enter button at the end of the listing above, assuming that the simulation will run through. To give you a very rough time estimate: The present example took $4\frac{1}{2}$ hours on a laptop with Intel i5 CPU using a single thread. If you are curious to see some intermediate results: OGS-6 (re-)writes the `pvd` file after each output operation. Therefore you can open it in ParaView anytime and look what has already happened.

Taking a first look at the simulation results, the reaction front moving through the packed bed becomes visible, cf. Fig. 4.4: At the front the temperature rises due to the adsorption process. The total temperature lift is apprximately 70 K. At the same time the relative humidity drops from the inlet value of 72 % to almost zero. We plotted the relative humidity $p_\mathrm{V}/p_\mathrm{sat}$ here because according to Eqs. (2.30) and (2.31) it determines the local equilibrium loading. The water loading C of the zeolite raises at the reaction front, and the Darcy velocity accelerates due to the thermal expansion of the flowing gas.

In Fig. 4.4 one only few signs of the heat loss over the chamber wall are evident. In order to show the radial temperature gradient more clearly, a line graph in the radial (x) direction at the outlet of the chamber is shown in Fig. 4.3, since near the outlet due to the high temperatures the gradients are most pronounced. But even in that case the radial temperature variation is only $\approx 3\,\mathrm{K}$, which is almost negligible compared to the 70 K temperature lift in the axial direction. However, if in a different application where the wall heat transfer coefficient is different, the radial gradients might be more pronounced.

Apart from the spatial distribution of physical quantities one is often interested in the temperatures at specific points in the bed (as measured with thermocouples), and the temperature and water content of the outflowing air. Fig. 4.5shows exemplarily the temperature evolution of a point close to

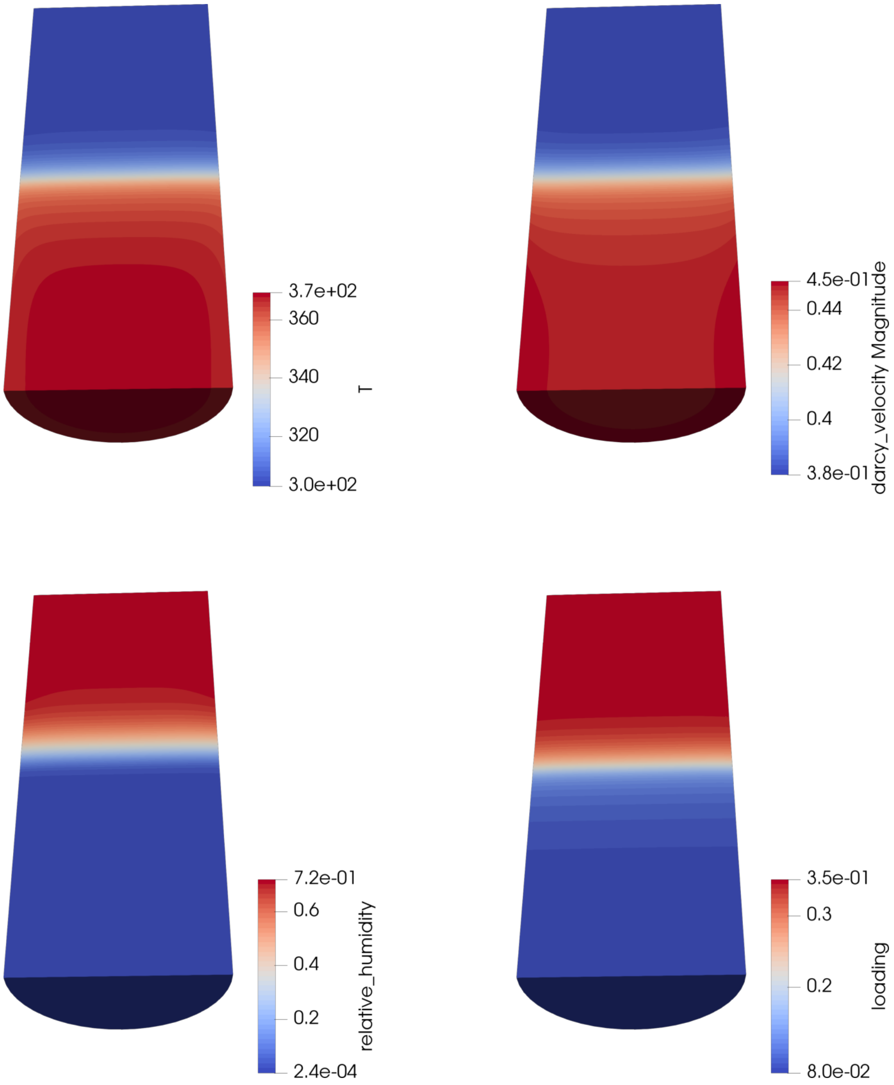


Figure 4.4 Spatially resolved values of different physical quantities at $t = 1500\,\mathrm{s}$. Moist air flows from the top through the sorption chamber. *Top left:* temperature T in K. *Top right:* Darcy velocity $|\boldsymbol{v}_{\mathrm{Darcy}}|$ in $\mathrm{m\,s^{-1}}$. Essentially, the velocity vectors point towards the negative y axis. *Bottom left:* relative humidity. *Bottom right:* adsorbent loading C.

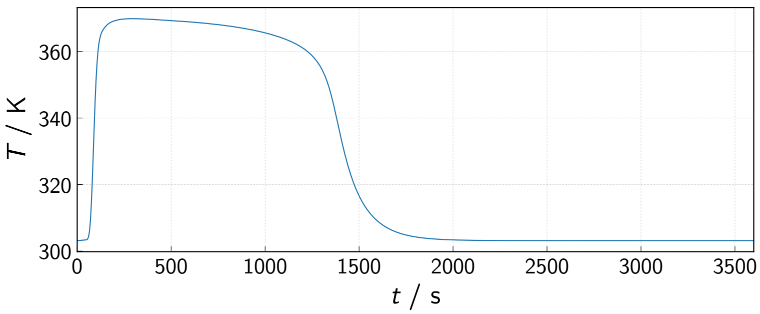


Figure 4.5 Temperature evolution over time at a point located at the axis of the cylindrical bed at a distance of $\Delta y = 0.065\,\mathrm{m}$ from the outlet

the very center of the packed bed. At the beginning of the simulation the temperature quickly raises to its maximum value of 370 K. Then it slowly decreases until the reaction front reaches that location in the bed at roughly $t = 1300 \ldots 1600\,\mathrm{s}$, accompanied by the temperature decrease, and followed by a short fading out of the temperature to the value of the inlet temperature. The slow temperature decline between $t = 250\,\mathrm{s}$ and $t = 1000\,\mathrm{s}$ is due to the fact that once the zeolite loading in the upstream of the point increases, the differential enthalpy of adsorption Δh_{ads} decreases. Therefore, adsorbing the same amount of water upstream leads to a lessening amount of released heat, and thereby to a lower temperature lift with respect to the inlet temperature. The differences between the simulation results in this example and the results in [Met+14] can be attributed to different heat conductivity and heat transfer parameters as well as the missing thermal mass of the wall in our model.

The temperature and humidity of the outflowing air are easily amenable to measurements. Therefore they are central quantities in the model validation, and for predictions made with the models. Furthermore, they can of course be used to control the thermochemical heat storage device in practical applications. Assuming perfect mixing of the outlet air and that the heat capacity of the outlet air does not vary over the cross-section of the adsorbent bed, the average temperature can be computed as the mass flux weighted average of the temperature at the outlet:

$$\bar{T}_{\mathrm{out}} = \int_{\Gamma_{\mathrm{out}}} T\,\dot{\boldsymbol{m}} \cdot \boldsymbol{n}\,\mathrm{d}\Gamma \bigg/ \int_{\Gamma_{\mathrm{out}}} \dot{\boldsymbol{m}} \cdot \boldsymbol{n}\,\mathrm{d}\Gamma\,, \tag{4.7}$$

where $\boldsymbol{n}$ is the normal vector of the outlet surface Γ_{out}, and the denominator is the total mass flux leaving the sorption chamber. Analogously the average vapour mass fraction in the outflowing air can be computed via

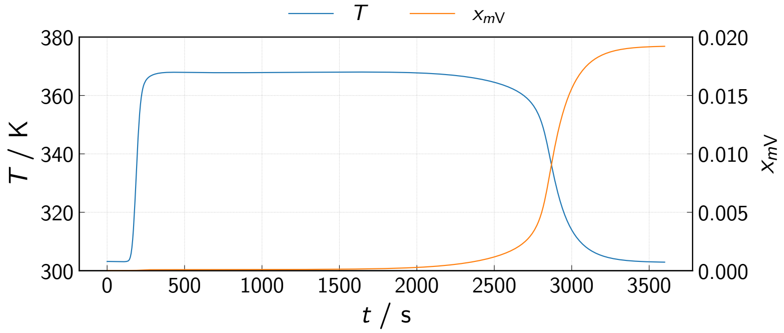


Figure 4.6 Mass flux weighted averages of T and $x_{m\mathrm{V}}$ at the outlet according to Eqs. (4.7) and (4.8). Those are the the quantities yielded by outflowing air measurements.

$$\bar{x}_{m\mathrm{V,out}} = \int_{\Gamma_\mathrm{out}} x_{m\mathrm{V}}\,\dot{\boldsymbol{m}}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma \Big/ \int_{\Gamma_\mathrm{out}} \dot{\boldsymbol{m}}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma\,, \tag{4.8}$$

where the numerator is the total vapour mass flux leaving the chamber. The mass flux density $\dot{\boldsymbol{m}}$ itself can be computed from $\operatorname{grad} p$ using Darcy's law Eq. (2.19) and the relation $\dot{\boldsymbol{m}} = \varrho_\mathrm{GR}\boldsymbol{v}_\mathrm{Darcy}$. In the current axially symmetric setting the surface integral over Γ is given by[5]:

$$\int_\Gamma f(r,z)\,\mathrm{d}\Gamma = \int_0^{2\pi}\mathrm{d}\varphi\int_0^{r_\mathrm{bed}} r\,f(r,z)\,\mathrm{d}r = 2\pi\int_0^{r_\mathrm{bed}} r\,f(r,z)\,\mathrm{d}r\ . \tag{4.9}$$

Performing those integrals for the present simulation yields the time lines shown in Fig. 4.6. After the heating up of the chamber (taking roughly 200 s), its heat output reaches a maximum and is maintained until $t = 2500$ s. After that the reaction front reaches the end of the chamber, such that the outlet temperature drops accompanied by the breakthrough of water vapour.

The outlet temperature is essential in order to decide if the temperature level the sorption chamber meets the requirements of the specific application. However, it is of equal importance to know the heating power and the heat storage density of the thermochemical heat storage device. Since heat is transported through the bed mainly advectively, the advective heating power $\dot{H}_\mathrm{adv}$ is given by

$$\dot{H}_\mathrm{adv} = \int_{\Gamma_\mathrm{out}} c_{p\mathrm{G}}\,(T - T_\mathrm{in})\,\dot{\boldsymbol{m}}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma \tag{4.10}$$

Where the integration is performed over the outlet, because during the adsorption phase we are mainly interested in what we get out of the reactor, not what

[5] For a surface whose normal points along the z direction (i. e., y direction inside OGS-6).

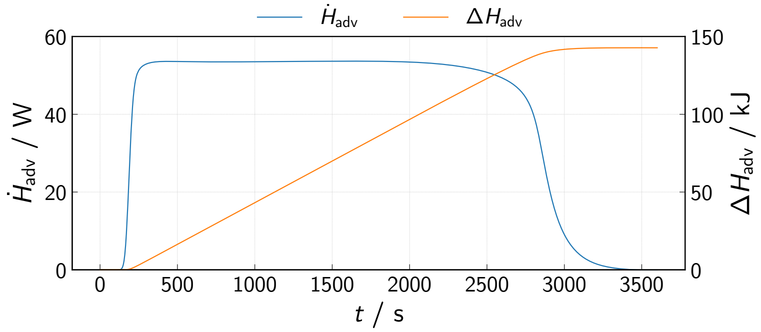


Figure 4.7 *Left axis*: Heat power due to advective heat transfer. *Right axis*: Integral heat advected by the air flow leaving the sorption chamber.

we have to put into it. The time integral over $\dot{H}_{\mathrm{adv}}$ then gives the amount of thermal energy obtained from the adsorbent bed $\mathit{\Delta}H_{\mathrm{adv}}$. Both quantities are shown in Fig. 4.7. The advected heat power shows the same progression as the temperature in Fig. 4.6. The integral advected heat amounts to 143 J, which corresponds to a heat storage density of 162 kWh m^{-3}.

With that said, the main performance characteristics of a sorption heat storage device have been examined. What has not been shown in this chapter is the strong dependence of those performance markers on the chosen operating conditions (especially humidity and mass flux rate). It's up to the reader to change them in the simulation and assess their effect, or to estimate it from the adsorption equilibrium and the results described in the chapter. For an exemplary illustration, Fig. 4.8 shows two different operating scenarios in the van't Hoff plot of a zeolite 13XBF and Fig. 4.9 shows the corresponding operating windows on the characteristic curve. Based on that, adsorption enthalpies per cycle can be estimated, while mass flux considerations are required to assess the power output. The following chapter will deal with two extensions to the present simulation: Considering the radial porosity profile, and modelling adsorption-desorption cycles via transient boundary conditions.

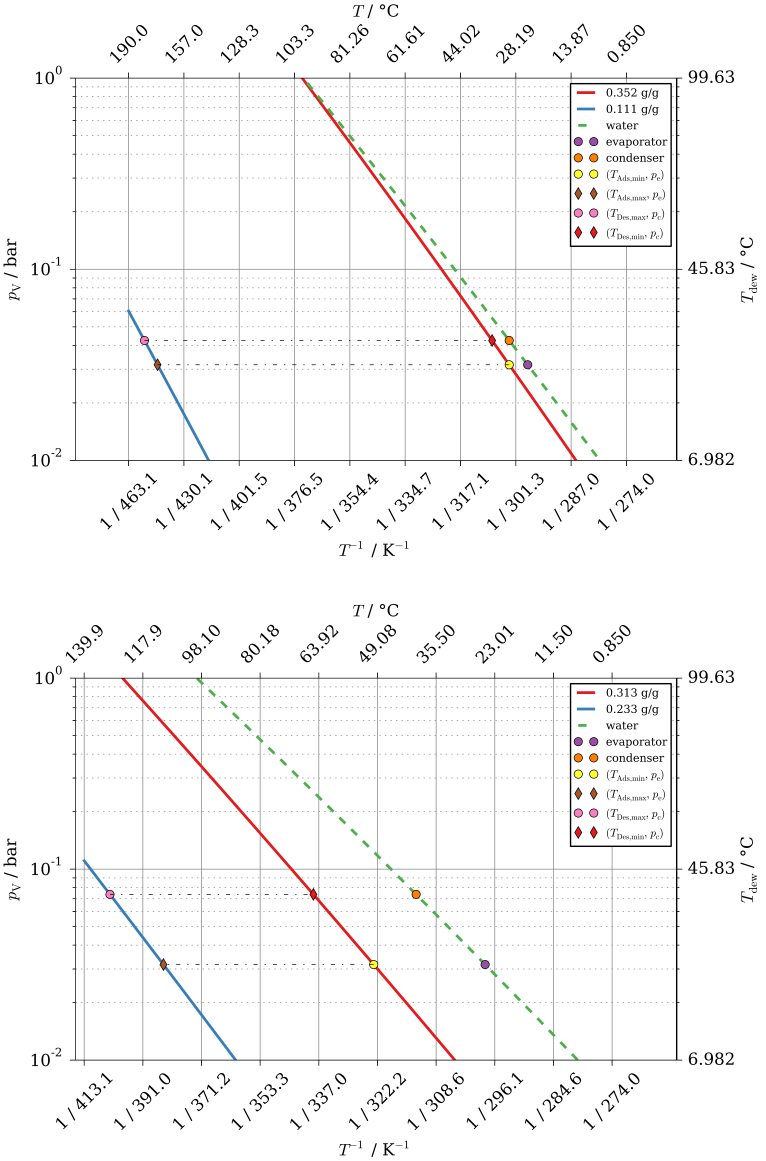


Figure 4.8 Adsorption and desorption isosteres corresponding to two different sets of adsorption and desorption conditions (i.e. temperature and partial vapour pressure). The isostere of the liquid-vapour equilibrium is also plotted and is related to evaporator/condenser operation.

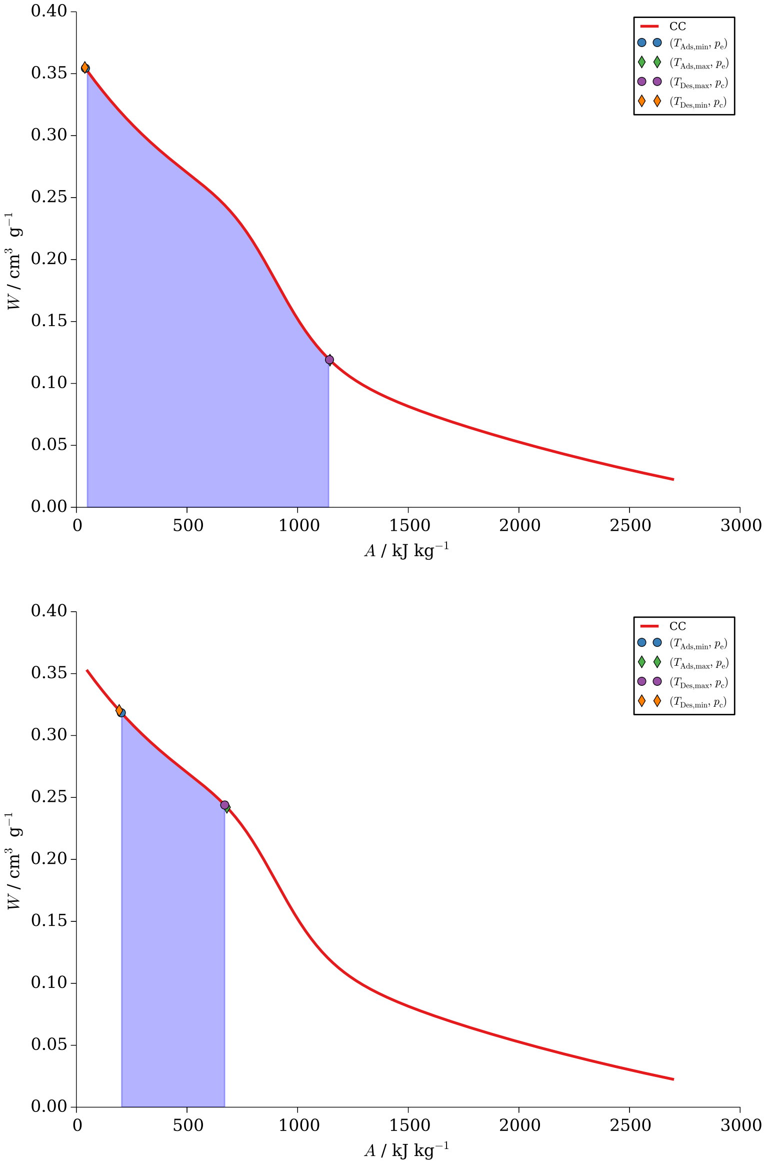


Figure 4.9 Characteristic curves and operating window corresponding to the conditions illustrated in Fig. 4.8.

Chapter 5
Extensions

5.1 Radial Porosity Profile

In Chapter 4, for simplicity we assumed a uniform value for the porosity ϕ_G of the packed bed. However, near the chamber wall the packing of zeolite beads cannot be as tight as it is in the interior of the bed. Therefore, the porosity will change from the bulk value of 0.4, and in fact it will nearly approach 1 at the wall. For packed beds of rough or imperfect or polydisperse spheres an empirical relation between the radial position in the bed and the porosity at that position has been found [Gie98]:[1]

$$\phi_\mathrm{G}(r) = 0.4 \cdot \left[1 + 1.36 \cdot \exp\left(-5 \cdot \frac{r_\mathrm{bed} - r}{d_\mathrm{p}}\right)\right] , \tag{5.1}$$

where the wall is located at r_bed, see Fig. 5.1. If we wanted to use this porosity profile directly together with Darcy's law and the Kozeny-Carman relation Eq. (2.19) in our simulations the following problem would arise: The permeability would be higher near the wall than in the interior of the bed by several orders of magnitude. Therefore nearly the whole air flow would take place in the regions near the wall and the bulk of the bed would adsorb vapour only at a very low rate. That clearly contradicts our expectations and experimental evidence.

Using the simplistic Darcy law in our model on the one hand spares us solving the fluid momentum balance for the fluid velocity. On the other hand, we are not able to directly model the interplay between porosity changes, zero velocity BC at the wall and the fluid viscosity. Neglecting these effects is what causes the unexpected behaviour described in the previous paragraph. In order to account for them, unfortunately we have to fall back to some

[1] That formula has to be interpreted in a homogenized sense. But since ϕ_G is a homogenized quantity, this should be clear anyway.

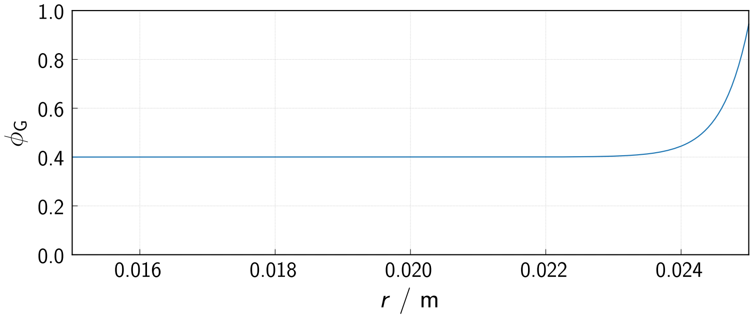


Figure 5.1 Porosity profile of the bed in radial direction. The chamber wall is at $r_{\text{bed}} = 0.025\,\text{m}$.

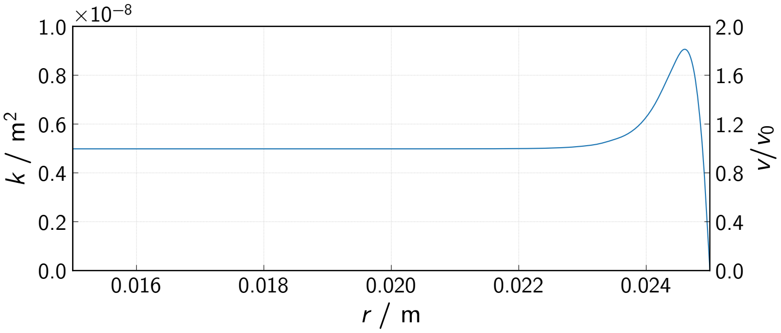


Figure 5.2 Velocity profile inside a packed bed of non-uniform porosity, normalized to the velocity in the center of the bed (right axis). At the left axis the effective permeability values, obtained from applying Darcy's law to the velocity profile, are shown.

other simulation software able to solve Stokes' equation[2]. If one does that, e. g., following the explanation in [Tso13], one obtains a velocity profile akin to the one shown in Fig. 5.2 at the right axis. The details of the profile depend on the Reynolds number. Using that profile and Darcy's law one can compute an effective radial permeability profile of the bed, as shown in Fig. 5.2 at the left axis. This permeability profile will be used in the simulation in the current section. Furthermore, it is justified to use the velocity distribution just computed also for the mass flux BC, because even if one takes a parabolic velocity profile as the BC for the Stokes flow simulation, the velocity distribution inside the packed bed will essentially be equal the one in Fig. 5.2 from the very start of the packed bed, see Fig. 5.3.

[2] Stokes' equation can easily be implemented into OGS. In fact, it has a lot of similarities with the implementation of linear elastic isotropy, especially with implementations considering (near) incompressibility.

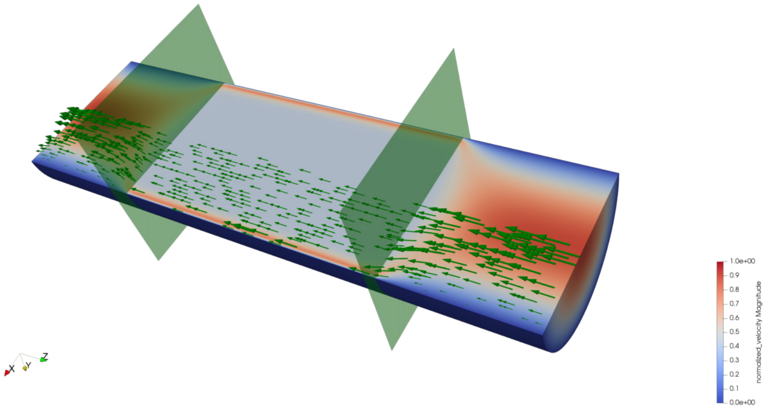


Figure 5.3 Simulation of Stokes flow inside a pipe partially filled with a sphere packing. The spheres are located between the green planes. As boundary condition at the inlet of the pipe a parabolic velocity profile has been set (Poiseuille flow). The velocity profile inside the packed bed part of the pipe essentially does not change in the axial direction.

Input Files

Since now we have to resolve the steep gradients in the permeability profile near the wall, the mesh should be more refined near the wall than in the last chapter. A cheap way to do this without changing the number of elements in the radial direction is to adjust the `--mx` parameter of the `generateStructuredMesh` utility:

```
$ generateStructuredMesh -e quad --lx 0.025 --ly 0.125
   ↪ --nx 20 --ny 100 --mx 0.75 -o adsorption.vtu
info: Mesh created: 2121 nodes, 2000 elements.
```

That comes of course at the expense of coarser elements at the centre of the bed. Subsequently, the porosity and permeability profiles, Eq. (5.1) and Fig. 5.2, must be written as element values to the generated mesh, and the definitions of the respective <parameters> in the `prj` file must be adapted:

```
<parameter>
  <name>permeability</name>
  <type>MeshElement</type>
  <field_name>permeability</field_name>
</parameter>

<parameter>
  <name>porosity</name>
  <type>MeshElement</type>
  <field_name>porosity</field_name>
</parameter>
```

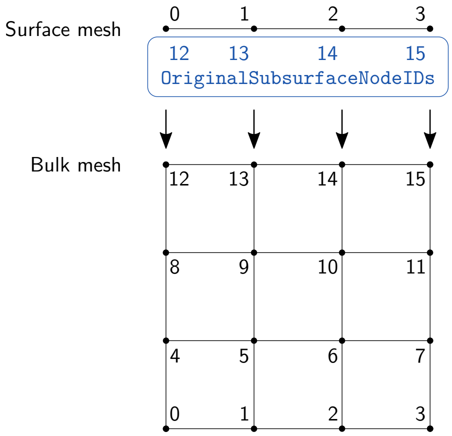


Figure 5.4
Mapping from the boundary mesh to the bulk mesh via the nodal field `OriginalSubsurfaceNodeIDs` of the boundary mesh. The black numbers are the node ids on each mesh.

And finally, we don't have a uniform Neumann BC anymore, but one varying over the cross section of the bed. Such a BC is setup as a `NonuniformNeumann` `<boundary_condition>`.

```
<boundary_condition>
  <!-- dummy values -->
  <geometrical_set>geometry</geometrical_set>
  <geometry>top</geometry>
  <!-- end dummy values -->

  <type>NonuniformNeumann</type>
  <field_name>mass_flux</field_name>
  <mesh>inlet.vtu</mesh>
</boundary_condition>
```

Its specification in the `prj` file is currently still not entirely finished. Therefore one has to give a geometry specification `<geometrical_set>` and `<geometry>`, which must be valid, i. e.must also exist in the `gml` file, but will not be used in the simuation. Instead the non-uniform Neummann BC will read the elements to whom it will be applied and the flux values from the specified `<mesh>`, namely from the nodal field with the given `<field_name>`. The boundary mesh, here `inlet.vtu`, itself must additionally contain the nodal field `OriginalSubsurfaceNodeIDs`, which maps each node of the surface mesh to a node of the bulk mesh.[3] This mapping is depicted schematically in Fig. 5.4. The rest of the `prj` file is the same as in the previous chapter, such that now the simulation can be run and the results can be discussed.

[3] To be precise: The field must be a 64 bit unsigned integer field, i. e., in the `vtu` file the corresponding tag must read:
`<DataArray type="UInt64" Name="OriginalSubsurfaceNodeIDs" ...>`

Results

Figure 5.5 shows the conditions inside the sorption chamber at $t = 1500\,\mathrm{s}$. When compared to Fig. 4.4 there are a few important differences: As prescribed via boundary condition and radial permeability profile, the Darcy velocity close to the chamber wall is much higher than in the interior of the bed. Consequently a curved reaction front arises, which moves faster near the chamber wall than in the interior due to enhanced advective heat transport and which is reflected in the shape of the temperature, humidity and loading plots. Furthermore, now there are more pronounced radial temperature gradients in the bed, too.

Since the reaction front near the wall moves ahead of the inner regions of the bed, we expect that the water vapour breaks through earlier than in the simulation in Chapter 4. And indeed that's what the averages over the outlet show, Fig. 5.6: An earlier breakthrough of moist air accompanied by a decline of the outlet temperature, because now a growing fraction of the water vapour in the air passes the zeolite bed without getting adsorbed which also results in a drop of heat release in this zone. As we already saw in the previous chapter, the heat power shows a behaviour completely analogous to the temperature. The integral advected heat now amounts to 139 J. The difference between that number and the 143 J from the previous chapter is due to the fact that now in fact there is less adsorbent mass in the sorption chamber.[4] Eventually, the time evolution of the average loading of the zeolite bed develops very similar for the simulation in the current section compared to Chapter 4. Essentially only for $t = 2500 \dots 3000\,\mathrm{s}$ there are some small deviations visible, see Fig. 5.7.

We want to close this section with a comment about the reliability of the extrapolation procedure as a continuation of what was already stated in Chapter 4. For this purpose we consider the mass flux in y direction at the inlet of the reactor. We select the inlet, because there we can compare the mass flux obtained from OGS-6 to the value set as a BC. In Fig. 5.8 top, the radial mass flux profile in (negative) y direction is plotted as computed with OGS-6 at different time steps along with the mass flux BC. It is clearly visible that for very early time steps the computed mass flux deviates considerably from the preset BC, see also Fig. 5.8 bottom. That can be partly attributed to the errors introduced by the extrapolation procedure as measured by the extrapolation residual Fig. 5.8 center. But also the mesh resolution in general plays a role: Remember that the porosity and permeability fields are defined per cell, but the mass flux is given as a nodal field. That fact alone might introduce additional discretization errors, notwithstanding that a Neumann BC is a natural BC and as such is fulfilled only to a level of accuracy determined by the mesh resolution. All in all, in the current example the extrapolation residual reports

[4] We only changed the porosity, but did not adapt the intrinsic density, such that $m_\mathrm{S} = \int_\Omega \varrho_\mathrm{S}\,\mathrm{d}\Omega = \int_\Omega \varrho_\mathrm{SR}\,\phi_\mathrm{S}\,\mathrm{d}\Omega$ decreased. In an experiment one would probably determine the apparent density ϱ_S of the zeolite, so care must be taken.

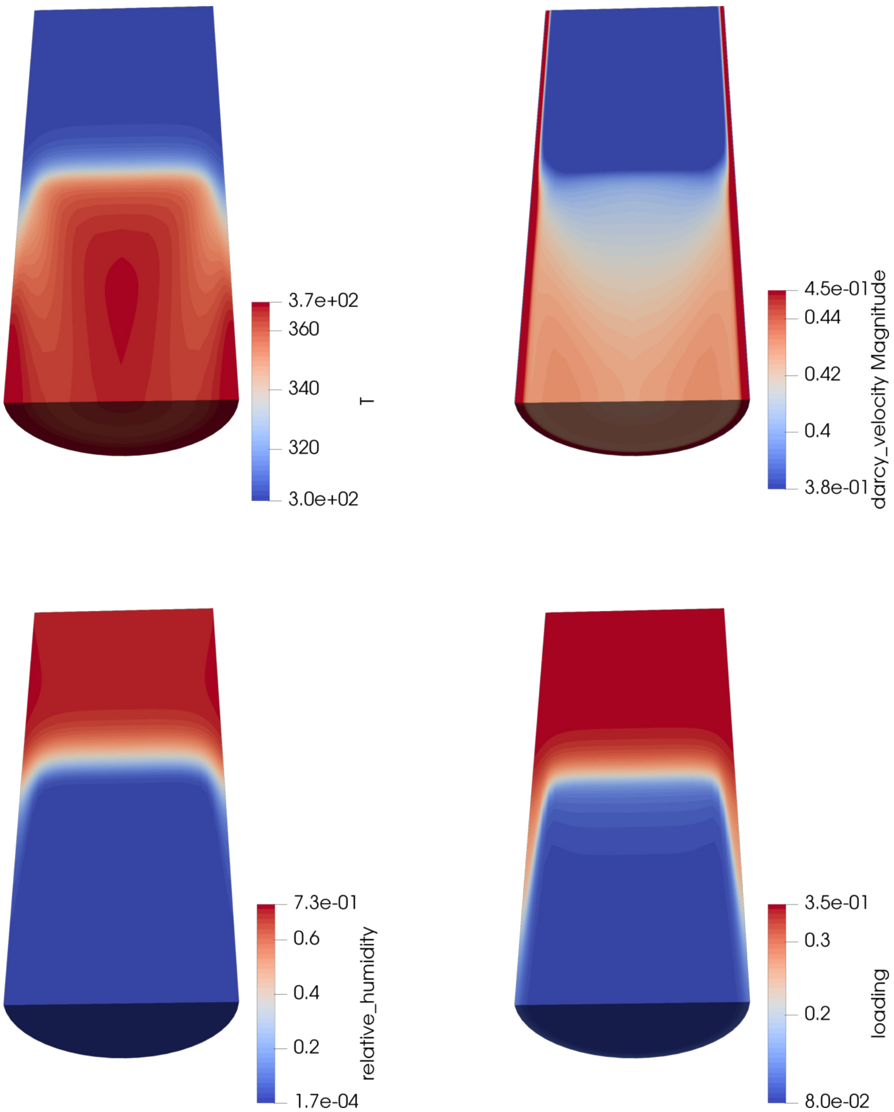


Figure 5.5 Spatially resolved values of different physical quantities at $t = 1500\,\mathrm{s}$. Moist air flows from the top through the sorption chamber. *Top left:* temperature T in K. *Top right:* Darcy velocity $|\boldsymbol{v}_{\mathrm{Darcy}}|$ in $\mathrm{m\,s^{-1}}$. Essentially, the velocity vectors point towards the negative y axis. The color map is capped at $0.45\,\mathrm{m\,s^{-1}}$ to be the same as in Fig. 4.4 and to increase the contrast in the interior of the bed. *Bottom left:* relative humidity. *Bottom right:* adsorbent loading C.

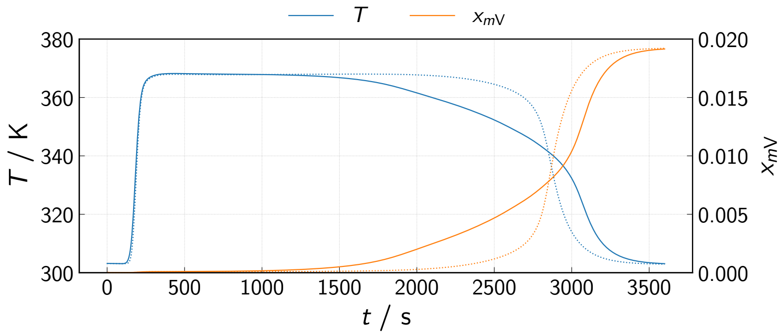


Figure 5.6 Mass flux weighted averages of T and $x_{m\mathrm{V}}$ at the outlet. Dotted lines are those from Fig. 4.6.

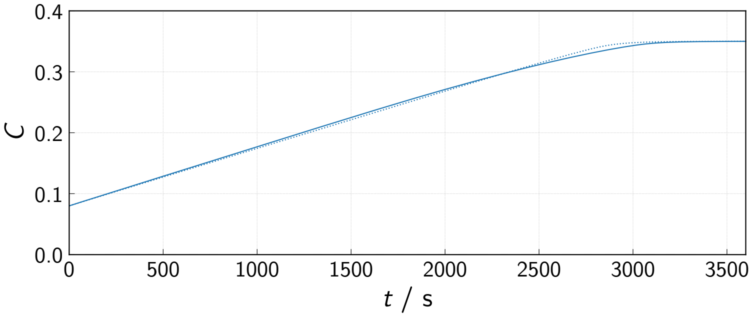


Figure 5.7 Temporal evolution of the average loading in the sorption chamber. *Solid line*: Simulation from this section. *Dotted line*: Simulation from Chapter 4.

a maximum relative error of roughly $0.05\,\mathrm{kg\,m^{-2}\,s^{-1}}/0.8\,\mathrm{kg\,m^{-2}\,s^{-1}} \approx 6\,\%$, and for later time steps it even overestimates the error between the BC and the numerically obtained mass flux. We conclude that the extrapolation residual is a reliable marker of the extrapolation quality of secondary variables. And of course one has to be very careful when using such extrapolated quantities in further postprocessing, as the numerical errors can add up. Eventually, we want to stress that this is a problem solely of the extrapolation, i.e. that it occurs during post-processing: The application of Neumann BCs proceeds in the standard FEM way and the extrapolation error does not affect the actual simulation run, having no effect on the primary variables or the values of the secondary variables computed in the integration points. Nevertheless, extrapolated quantities can give you a headache from time to time.

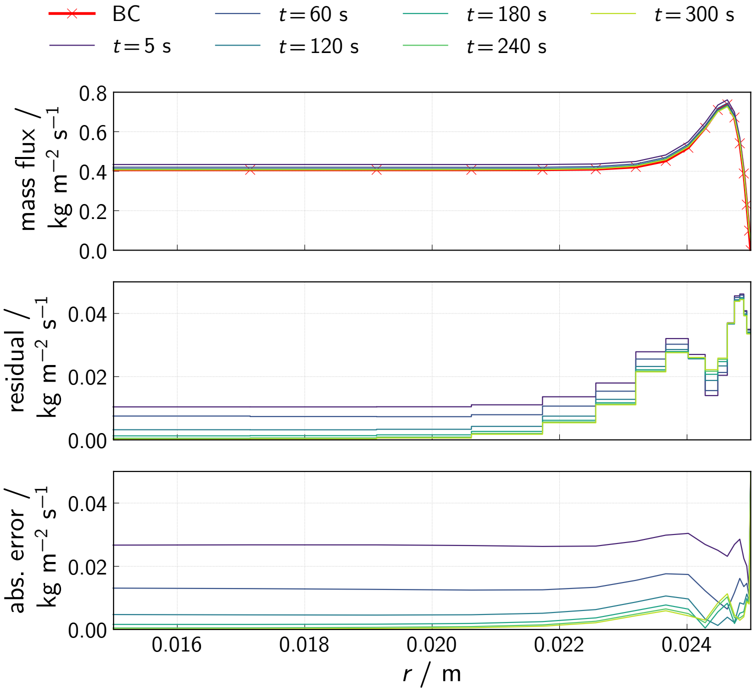


Figure 5.8 Mass flux at the inlet of the sorption chamber. *Top* Mass flux in negative y direction as prescribed by the BC and as obtained numerically during post-processing at different time steps. *Center* Extrapolation residual for the y component of the mass flux. *Bottom* Difference between the prescribed BC values and the extrapolated mass flux $|\dot{m}_{\mathrm{in,BC}} - \dot{m}_{\mathrm{in,OGS\text{-}6}}|$ obtained during post-processing. For a discussion of the implications and the difference between the values used during the simulation and the values given for post-processing, see the text.

5.2 Cyclic Discharging and Charging

Let's assume that the application we are interested in needs a temperature lift of 40 K over the reactor, i. e., the outlet temperature must be at least 70 °C/343.15 K. The outlet temperature falls below that limit at $t \approx 2800$ s, cf. Fig. 5.6. Hence, we now allow the regeneration of the fixed bed with $p_{\mathrm{V,in}} = 1752$ Pa[5] and $T_{\mathrm{in}} = 180$ °C at the same mass flux for the same duration, with a small switching time in between. A closer look at the state of the adsorbent bed at that point in time (2800 s), Fig. 5.9, confirms that the zeolite bed has not yet reached its entirely adsorbed state. In particular near the outlet the loading is partly still at its initial value. Therefore, we can expect that the same holds for the desorption phase, too, and that the next adsorption phase starts from a different initial state then the previous one. As such, it should take several cycles until the initial and final states of the bed for each cycle don't change anymore. Now we are interested in how many cycles of adsorption and desorption it takes until the system reaches a cyclic steady state, i. e., after how many cycles all subsequent cycles remain largely the same.

Input Files

To simulate those cycles we need transient boundary conditions. OGS-6 currently supports a special kind of general transient BCs: $f(\boldsymbol{x}, t) = f_1(\boldsymbol{x})\, f_2(t)$, i. e., a product of a function of position and a function of time. The former is set up as we already know using `Constant`, `MeshNode`, or `MeshElement` <parameters>. The latter is specified using <curves>:

```
<curves>
  <curve>
    <name>T_in_curve</name>
    <coords>
      0.0 2800.0 3000.0 5800.0 6000.0 8800.0 9000.0
         ↪ 11800.0 12000.0 14800.0 15000.0 17800.0
         ↪ 18000.0 20800.0 21000.0 23800.0 24000.0
         ↪ 26800.0 27000.0 29800.0
    </coords>
    <values>
      303.15 303.15 453.15 453.15 303.15 303.15 453.15
         ↪ 453.15 303.15 303.15 453.15 453.15 303.15
         ↪ 303.15 453.15 453.15 303.15 303.15 453.15
         ↪ 453.15
    </values>
  </curve>
```

[5] That corresponds to 75 % relative humidity at 20 °C, which is a common value for ambient conditions in central Europe. I. e., to regenerate the bed, ambient air of 20 °C is heated up to 180 °C before it is pumped through the adsorbent bed.

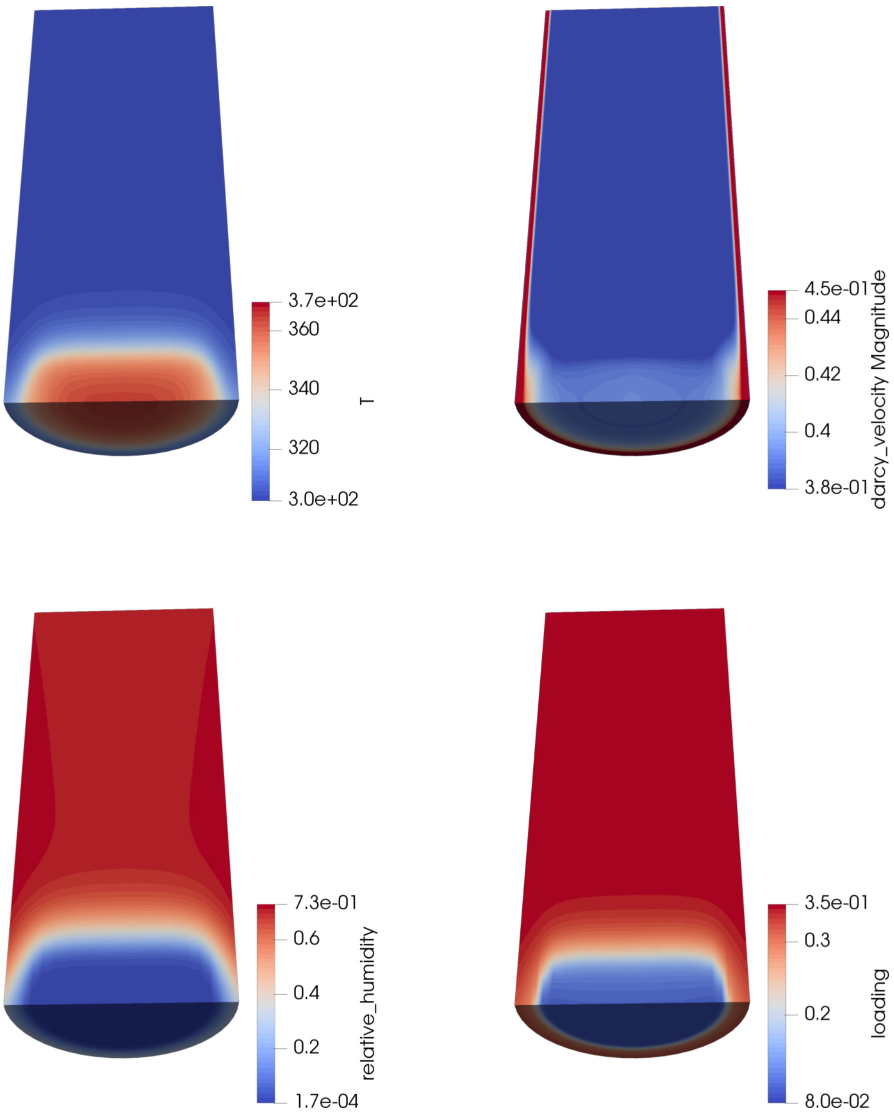


Figure 5.9 Spatially resolved values of different physical quantities at $t = 2800\,\mathrm{s}$ for the simulation from Section 5.1, i. e., near the end of the simulation run. At that point in time, the average outlet temperature undercuts 70 °C. Moist air flows from the top through the sorption chamber. *Top left:* temperature T in K. *Top right:* Darcy velocity $|\boldsymbol{v}_{\mathrm{Darcy}}|$ in $\mathrm{m\,s^{-1}}$. Essentially, the velocity vectors point towards the negative y axis. The color map is capped at $0.45\,\mathrm{m\,s^{-1}}$. *Bottom left:* relative humidity. *Bottom right:* adsorbent loading C.

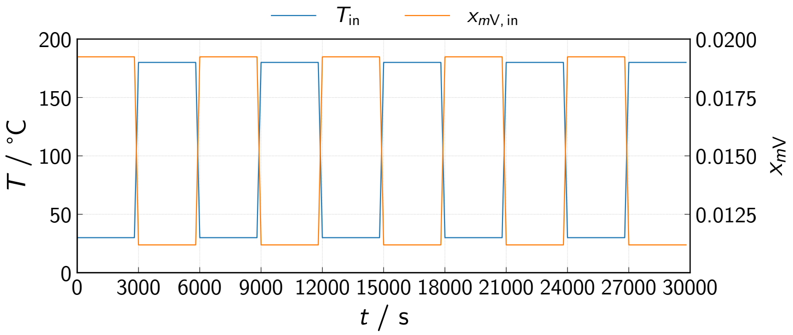


Figure 5.10 Operation conditions: Inlet temperature and humidity.

The curve just defined has a <name> for later reference. The <coords> are the points in time where function <values>, here temperatures, are given. Between the supporting points the curve is linearly interpolated. A similar curve is also defined for $x_{m\text{V,in}}$. As already visible in the listing the switching time between the adsorption and desorption cycles lasting 2800 s each is 200 s. Five adsorption-desorption cycles will be simulated. The operation conditions are summarized in Fig. 5.10.

Finally, we have to declare special time dependent <parameters> as just explained, which can be used like any other parameter for BCs:

```
<parameters>
  <parameter>
    <name>one</name>
    <type>Constant</type>
    <value>1</value>
  </parameter>

  <parameter>
    <name>T_in</name>
    <type>CurveScaled</type>
    <curve>T_in_curve</curve>
    <parameter>one</parameter>
  </parameter>
```

The second parameter of CurveScaled <type> represents the function $f(\boldsymbol{x},t) = f_1(\boldsymbol{x})\, f_2(t)$, where $f_1(\boldsymbol{x})$ is the <parameter> one, i. e., the first parameter defined in the listing: Constant unity. And $f_2(t)$ is the <curve> T_in_curve, i. e., the transient piecewise linear function we just discussed. The rest of the prj file as well as the gml file and the meshes are the same as in the preceding section. So we are once again ready to run ogs.

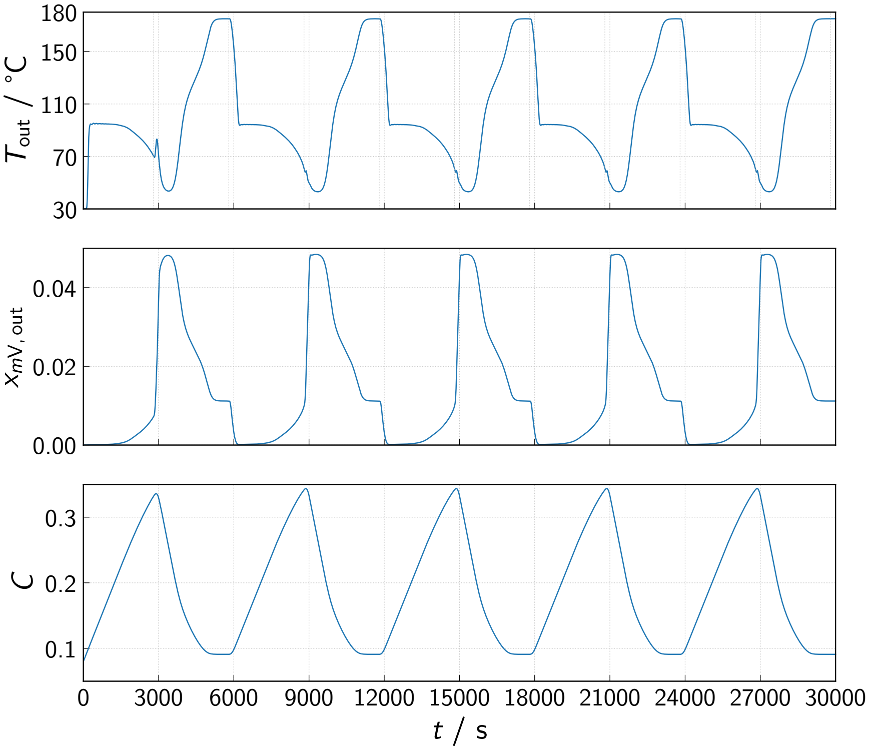


Figure 5.11 Mass flux-weighted averages of T and $x_{m\mathrm{V}}$ at the outlet (top and center) and average loading C of the adsorbent bed (bottom). The thin vertical lines in the T plot denote all points in time where a switching of BCs according to the prescribed <curves> occurs.

Results

The outlet temperature and humidity together with the evolution of the average loading in the bed tell us that the cyclic steady state has already been reached after the first complete adsorption-desorption cycle of 6000 s, see Fig. 5.11. However, there are some noteworthy details in those plots:

- Due to the short switching time between the individual phases, the adsorbent bed does not cool down to the initial temperature of 30 °C anymore. And at the beginning of the second, third, etc. adsorption phases the bed does not have to be heated up, but on the contrary it cools down from the desorption temperature of 180 °C during the adsorption phase.
- There is a peak in the outlet temperature during the switching after the first adsorption phase, $t = 2800 \ldots 3000$ s. This is caused by the fact that

(i) the hot inlet air starts the desorption near the inlet. (ii) The hot inlet air cools down due to the heat of desorption. And (iii) near the outlet the adsorbent loading is still low, such that the moist air from upstream is adsorbed again leading to further heat release, cf. Fig. 5.12.

- At the end of the desorption phase, steady state loading and outlet temperature are reached. Therefore, in principle at the selected operating conditions the desorption phase could be made shorter. That, however, might not hold true for other desorption conditions, e. g., if one uses inlet air containing only very little water vapour but having a lower temperature, as was suggested, e. g., in [Met+13] using a second adsorption bed to pre-dry the inlet air. Albeit steady state is reached, the outlet temperature does not reach the inlet temperature of 180 °C due to surface heat losses. They are larger in the desorption than in the adsorption phase due to the larger temperature difference to the ambient. That entails larger radial temperature gradients over the fixed bed, cf. Fig. 5.14. A further consequence of the higher loading is that at the end of the second, third, etc. adsorption phase there is no peak in the outlet temperature, cf. Fig. 5.11.
- In the second and all subsequent adsorption phases, the outlet temperature falls below the limit of 70 °C before the switching time is reached. The reason for that is that in the preceding desorption phases the initial loading of 8 % is not exactly restored again. Therefore, after the same time of adsorption the loading in the bed is higher than in the first phase, and the outlet temperature drops earlier, see Fig. 5.13, but also Fig. 5.12.

Maybe you would like to consider a scenario where the adsorbent bed of an open adsorption chamber cools down before it is discharged again. In order to simulate that with OGS-6, one would have to flush the sorption chamber with extremely dry, cold air that is in adsorption equilibrium with the zeolite, such that no sorption processes take place. The more natural possibility of just sealing the sorption chamber and wait until it has cooled down via surface heat losses cannot be modelled at the current state, because unfortunately it is not yet possible to define the then transient non-uniform Neumann-BC for the mass flux. But that feature will be implemented in the future for sure. In the meantime you could also run a series of separate simulations where later simulations use the final state of the preceding simulation as their initial conditions. That concludes the adsorption heat storage simulation examples in this book, and we think that you are now well prepared to run your own simulations with OGS-6 in that area.

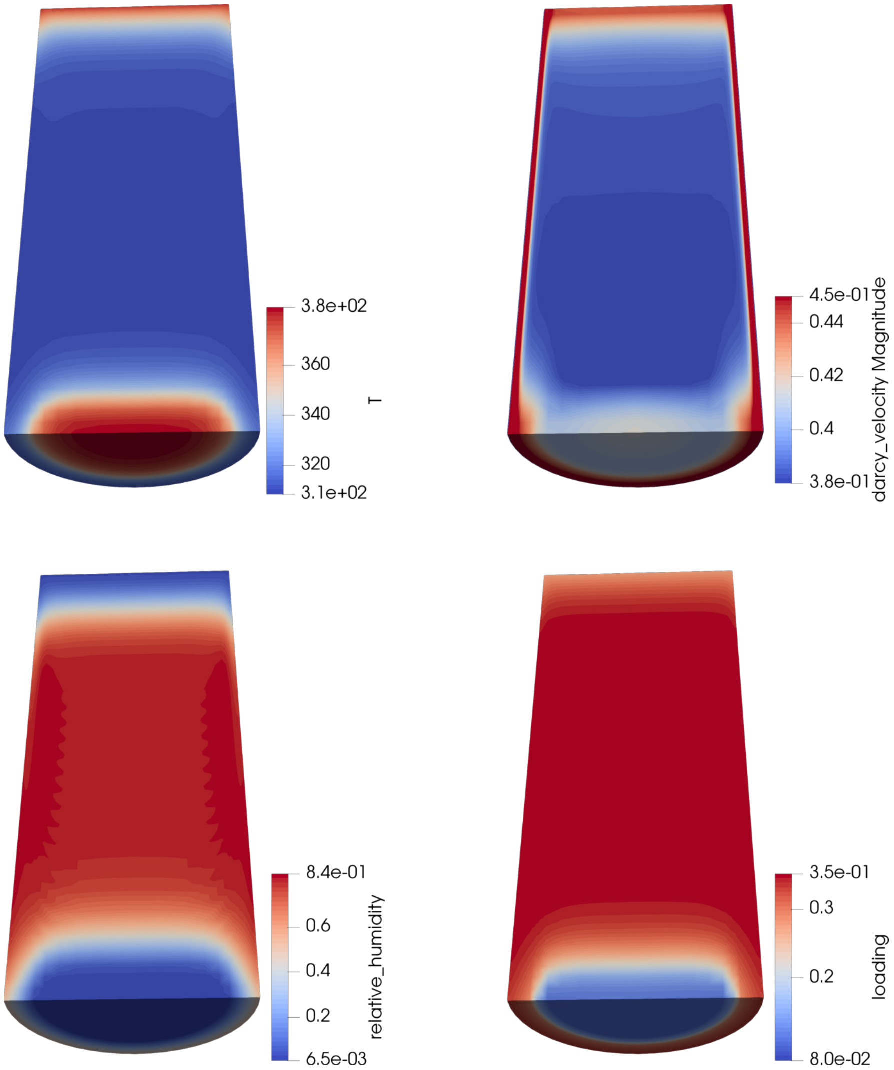


Figure 5.12 Spatially resolved values of different physical quantities at $t = 2900\,\mathrm{s}$, i. e., in the middle of the switching phase from adsorption to desorption operation conditions. Air flows from the top through the sorption chamber. The color maps have been rescaled compared to the preceding figures. *Top left:* temperature T in K. *Top right:* Darcy velocity $|\boldsymbol{v}_{\mathrm{Darcy}}|$ in $\mathrm{m\,s^{-1}}$. Essentially, the velocity vectors point towards the negative y axis. The color map is restricted to $[0.38\,\mathrm{m\,s^{-1}}, 0.45\,\mathrm{m\,s^{-1}}]$. *Bottom left:* relative humidity. *Bottom right:* adsorbent loading C.

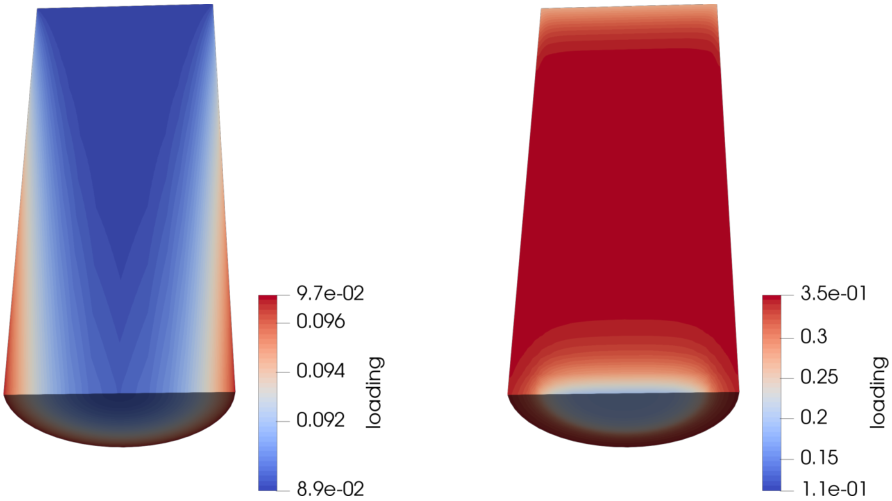


Figure 5.13 Adsorbent loadings C. *Left*: At $t = 5800\,\mathrm{s}$, i. e., at the end of the first desorption phase. The initial loading of uniformly $8\,\%$ is not reached again. *Right*: At $t = 8900\,\mathrm{s}$, i. e., at the end of the second adsorption phase. Near the outlet the loading is now higher than after the first adsorption phase, cf. Fig. 5.12.

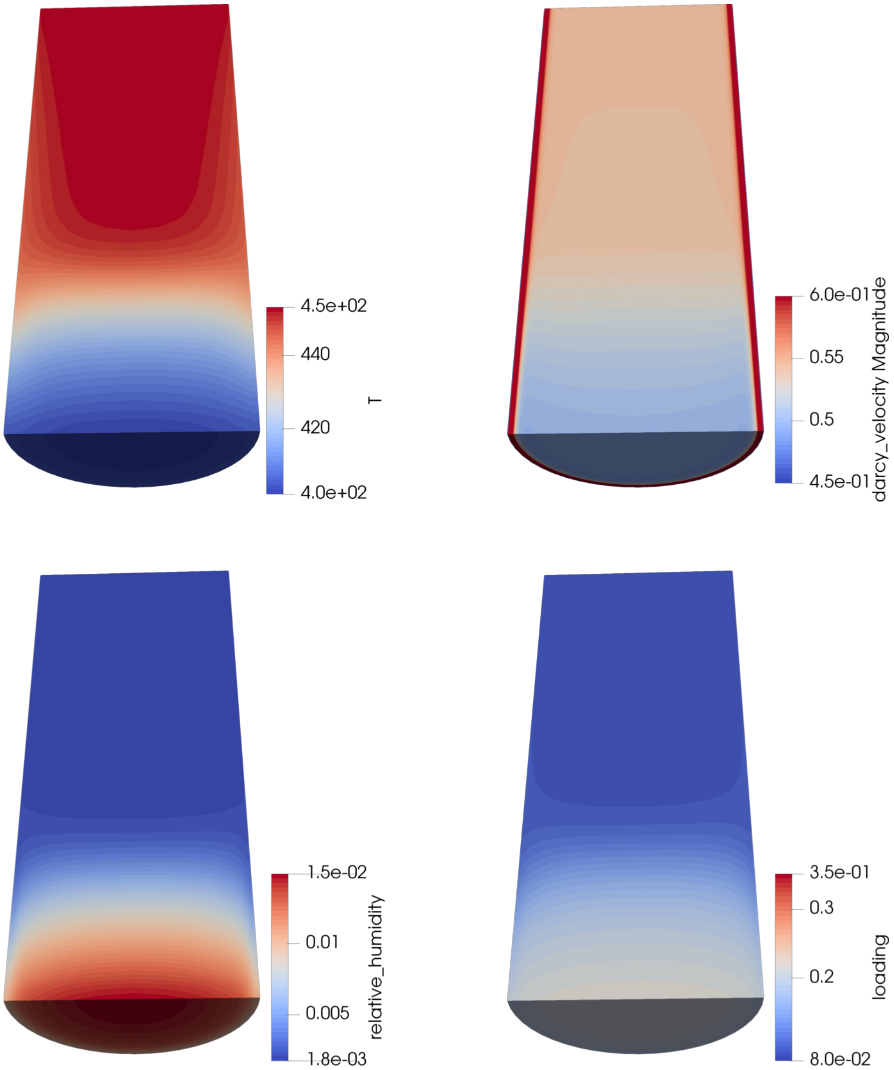


Figure 5.14 Spatially resolved values of different physical quantities at $t = 4500\,\mathrm{s}$, i. e., in the middle of the first desorption phase. Hot, dry air flows from the top through the sorption chamber. The color maps have been rescaled compared to the preceding figures. *Top left:* temperature T in K. *Top right:* Darcy velocity $|\boldsymbol{v}_{\mathrm{Darcy}}|$ in $\mathrm{m\,s^{-1}}$. Essentially, the velocity vectors point towards the negative y axis. The color map is restricted to $[0.45\,\mathrm{m\,s^{-1}}, 0.6\,\mathrm{m\,s^{-1}}]$. *Bottom left:* relative humidity. *Bottom right:* adsorbent loading C.

Chapter 6
Closing remarks

This book gave an introduction into the multi-physical modelling of thermochemical heat storage with a particular focus on systems based on the capacity of zeolites to reversibly adsorb water from a humid gas stream. There are many topics that were not discussed here but that are very relevant: other adsorption working pairs or systems based on absorption or even chemical reactions, high-temperature heat storage for the process industry or power generation, material degradation, the array of possible constitutive relations for the individual processes (flow, heat transport, etc.) and boundary conditions, reactor design, etc. For now, we refer the reader to the wealth of literature cited here and referenced in those citations.

OpenGeoSys is constantly growing with new material models, processes, and features becoming available. We would like to encourage a regular visit to www.opengeosys.org. There, you can also find examples on how to model many of the things we did not address here.

Last but not least, any active participation is greatly appreciated. If you would like to get your hands dirty and extend the software, implement other, better models, don't hesitate to download the source code, extend it, and send us a pull-request via GitHub. How to do that? Find out at—you guessed it—https://docs.opengeosys.org/docs.

References

[AF01] S. Afandizadeh and E. Foumeny. “Design of packed bed reactors: guides to catalyst shape, size, and loading selection”. In: *Applied thermal engineering* 21.6 (2001), pp. 669–682.

[AHP12] A. Arteconi, N. Hewitt and F. Polonara. “State of the art of thermal storage for demand-side management”. In: *Applied Energy* 93.0 (2012), pp. 371–389.

[AR11] A. H. Abedin and M. A. Rosen. “A Critical Review of Thermochemical Energy Storage Systems”. In: *The Open Renewable Energy Journal* 4 (2011), pp. 42–46.

[AR12] A. Abedin and M. Rosen. “Closed and open thermochemical energy storage: Energy- and exergy-based comparisons”. In: *Energy* 41.1 (2012), pp. 83–92.

[Ari07] Y. I. Aristov. “Novel Materials for Adsorptive Heat Pumping and Storage: Screening and Nanotailoring of Sorption Properties”. In: *Journal of Chemical Engineering of Japan* 40.13 (2007), pp. 1242–1251.

[Ari13] Y. I. Aristov. “Challenging offers of material science for adsorption heat transformation: A review”. In: *Applied Thermal Engineering* 50.2 (2013), pp. 1610–1618.

[BDS66] B. Bering, M. Dubinin and V. Serpinsky. “Theory of Volume Filling for Vapor Adsorption”. In: *Journal of Colloid and Interface Science* 21.4 (1966), pp. 378–393.

[Böt+16] N. Böttcher, N. Watanabe, U.-J. Görke and O. Kolditz. *Geoenergy Modeling I.* Springer-Verlag GmbH, 2016.

[CCC12] J. Cot-Gores, A. Castell and L. F. Cabeza. “Thermochemical energy storage and conversion: A-state-of-the-art review of the experimental research under practical conditions”. In: *Renewable and Sustainable Energy Reviews* 16.7 (2012), pp. 5207–5224.

[CD77] D. Close and R. Dunkle. “Use of adsorbent beds for energy storage in drying of heating systems”. In: *Solar Energy* 19.3 (1977), pp. 233–238.

[CLR13] C. Chan, J. Ling-Chin and A. Roskilly. “Reprint of “A review of chemical heat pumps, thermodynamic cycles and thermal energy storage technologies for low grade heat utilisation””. In: *Applied Thermal Engineering* 53.2 (2013), pp. 160–176.

[CM90] H. Choi and A. Mills. “Heat and mass transfer in metal hydride beds for heat pump applications”. In: *International Journal of Heat and Mass Transfer* 33.6 (1990), pp. 1281–1288.

[Do98] D. D. Do. *Adsorption Analysis: Equilibria and Kinetics*. Series on chemical engineering v 2. London: Imperial College Press, 1998.

[Dub60] M. Dubinin. “Theory of the Physical Adsorption of Gases and Vapors and Adsorption Properties of Adsorbents of Various Natures and Porous Structures”. In: *Bulletin of the Academy of Sciences of the USSR, Division of Chemical Science* 9.7 (1960), pp. 1072–1078.

[EB02] W. Ehlers and J. Bluhm, eds. *Porous Media*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002.

[Eri76] A. C. Eringen, ed. *Mixtures and EM Field Theories*. Continuum physics Vol. 3. New York, NY: Acad. Press, 1976.

[GG95] P. Gardie and V. Goetz. *Thermal Energy Storage System by Solid Absorption for Electric Automobile Heating and Air-Conditioning*. Tech. rep. SAE Technical Paper, 1995.

[Gie98] M. Giese. “Strömung in Porösen Medien Unter Berücksichtigung Effektiver Viskositäten”. PhD thesis. TU München, 1998.

[Glu55] E. Glueckauf. “Theory of Chromatography. Part 10. Formulæ for Diffusion into Spheres and Their Application to Chromatography”. In: *Transactions of the Faraday Society* 51 (1955), p. 1540.

[Has98a] S. Hasnain. “Review on sustainable thermal energy storage technologies, Part I: heat storage materials and techniques”. In: *Energy Conversion and Management* 39.11 (1998), pp. 1127–1138.

[Has98b] S. Hasnain. “Review on sustainable thermal energy storage technologies, Part II: cool thermal storage”. In: *Energy conversion and management* 39.11 (1998), pp. 1139–1153.

[Hau02] A. Hauer. “Beurteilung Fester Adsorbentien in Offenen Sorptionssystemen Für Energetische Anwendungen”. PhD thesis. Technische Universität Berlin, 2002.

[HJ04] K. Hutter and K. Jöhnk. *Continuum Methods of Physical Modeling*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[Jeo+10] J. Jeong, S. Yamaguchi, K. Saito and S. Kawai. “Performance analysis of four-partition desiccant wheel and hybrid dehumidification air-conditioning system”. In: *International Journal of Refrigeration* 33.3 (2010), pp. 496–509.

[Kas88] W. Kast. *Adsorption aus der Gasphase: ingenieurwissenschaftliche Grundlagen und technische Verfahren.* Weinheim: VCH, 1988.

[Kay99] K. Kaygusuz. “The viability of thermal energy storage”. In: *Energy Sources* 21.8 (1999), pp. 745–755.

[Ker+11] H. Kerskes, B. Mette, F. Bertsch, S. Asenbeck and H. Drück. “Development of a thermo-chemical energy storage for solar thermal applications”. In: *Proceedings. ISES, Solar World Congress Proceedings.* 2011.

[Leh+17] C. Lehmann, S. Beckert, R. Gläser, O. Kolditz and T. Nagel. “Assessment of Adsorbate Density Models for Numerical Simulations of Zeolite-Based Heat Storage Applications”. In: *Applied Energy* 185 (2017), pp. 1965–1970.

[LN14] H. Liu and K. Nagano. “Numerical simulation of an open sorption thermal energy storage system using composite sorbents built into a honeycomb structure”. en. In: *International Journal of Heat and Mass Transfer* 78 (2014), pp. 648–661.

[Mai+11] P. Maier-Laxhuber, R. Schmidt, R. Wörz, A. Becky and G. Richter. “Sorption dryer with zeolite”. 2011.

[Met+13] B. Mette, H. Kerskes, H. Drück and H. Müller-Steinhagen. “New Highly Efficient Regeneration Process for Thermochemical Energy Storage”. In: *Applied Energy* 109.0 (2013), pp. 352–359.

[Met+14] B. Mette, H. Kerskes, H. Drück and H. Müller-Steinhagen. “Experimental and Numerical Investigations on the Water Vapor Adsorption Isotherms and Kinetics of Binderless Zeolite 13X”. In: *International Journal of Heat and Mass Transfer* 71 (2014), pp. 555–561.

[Met14] B. Mette. “Experimentelle und numerische Untersuchungen zur Reaktionsführung thermochemischer Energiespeicher”. PhD thesis. Universität Stuttgart, 2014.

[Meu13] F. Meunier. “Adsorption heat powered heat pumps”. In: *Applied Thermal Engineering* 61.2 (2013), pp. 830–836.

[Mia+16] X.-Y. Miao, C. Beyer, U.-J. Görke, O. Kolditz, H. Hailemariam and T. Nagel. “Thermo-Hydro-Mechanical Analysis of Cement-Based Sensible Heat Stores for Domestic Applications”. In: *Environmental Earth Sciences* 75.18 (2016), p. 1293.

[MM72] T. R. Marrero and E. A. Mason. “Gaseous Diffusion Coefficients”. In: *Journal of Physical and Chemical Reference Data* 1.1 (1972), pp. 3–118.

[Nag+13] T. Nagel, H. Shao, A. K. Singh et al. “Non-Equilibrium Thermochemical Heat Storage in Porous Media: Part 1 – Conceptual Model”. In: *Energy* 60 (2013), pp. 254–270.

[Nag+16a] T. Nagel, S. Beckert, C. Lehmann, R. Gläser and O. Kolditz. “Multi-Physical Continuum Models of Thermochemical Heat Storage and Transformation in Porous Media and Powder Beds—A Review". In: *Applied Energy* 178 (2016), pp. 323–345.

[Nag+16b] T. Nagel, M. Wokon, M. Linder, C. Lehmann, S. Beckert and O. Kolditz. "Accounting for Material Degradation in the Design of Thermochemical Energy Storage Reactors and Processes Using Numerical Simulations". In: *First EERA Annual Conference, 24–25 November 2016, Birmingham, UK.* 2016.

[Nag+17] T. Nagel, N. Böttcher, U.-J. Görke and O. Kolditz. "Computational Geotechnics: Storage of Energy Carriers". In: (2017).

[Non16] T. Nonnen. "Salz/Zeolith-Komposite Für Die Sorptionswärmespeicherung". PhD thesis. Universität Leipzig, 2016.

[NTs+09] K. E. N'Tsoukpoe, H. Liu, N. Le Pierrès and L. Luo. "A review on long-term sorption solar energy storage". In: *Renewable and Sustainable Energy Reviews* 13.9 (2009), pp. 2385–2396.

[Núñ02] T. Núñez. "Charakterisierung und Bewertung von Adsorbentien für Wärmetransformationsanwendungen". PhD thesis. Freiburg im Breisgau: Albert-Ludwigs-Universität, 2002.

[Par+14] P. Pardo, A. Deydier, Z. Anxionnaz-Minvielle, S. Rougé, M. Cabassud and P. Cognet. "A review on high temperature thermochemical heat energy storage". In: *Renewable and Sustainable Energy Reviews* 32 (2014), pp. 591–610.

[PMD92] M. E. Perez-Davis, B. I. McKissock and F. DiFilippo. *Thermochemical Energy Storage for a Lunar Base*. National Aeronautics and Space Administration, 1992.

[RTY15] A. Roskilly, P. Taylor and J. Yan. "Energy storage systems for a low carbon future–in need of an integrated approach". In: *Applied Energy* 137 (2015), pp. 463–466.

[Rut84] D. M. Ruthven. *Principles of Adsorption and Adsorption Processes*. New York: Wiley, 1984.

[Sac+15] A. Sachse, K. Rink, W. He and O. Kolditz. *OpenGeoSys Tutorial: Computational Hydrology I: Groundwater Flow Modeling*. Springer-Verlag GmbH, 2015.

[Sac+17] A. Sachse, E. Nixdorf, E. Jang et al. *OGS Tutorial: Computational Hydrology II: Groundwater Quality Modeling*. Springer-Verlag GmbH, 2017.

[Sem+17] S. Semprini, C. Lehmann, S. Beckert et al. "Numerical Modelling of Water Sorption Isotherms of Zeolite 13XBF Based on Sparse Experimental Data Sets for Heat Storage Applications". In: *Energy Conversion and Management* 150 (2017), pp. 392–402.

[Sha+16] H. Shao, P. Hein, A. Sachse and O. Kolditz. *Geoenergy Modeling II*. Springer-Verlag GmbH, 2016.

[SLH79] R. A. Shigeishi, C. H. Langford and B. R. Hollebone. "Solar energy storage using chemical potential changes associated with drying of zeolites". In: *Solar Energy* 23.6 (1979), pp. 489–495.

[TH17] K. Tan and B. Hameed. “Insight into the Adsorption Kinetics Models for the Removal of Contaminants from Aqueous Solutions”. In: *Journal of the Taiwan Institute of Chemical Engineers* 74 (2017), pp. 25–48.

[TN65] C. Truesdell and W. Noll. “The Non-Linear Field Theories of Mechanics”. In: *The Non-Linear Field Theories of Mechanics*. Springer Berlin Heidelberg, 1965, pp. 1–579.

[Tru84] C. Truesdell. *Rational Thermodynamics*. New York, NY: Springer New York, 1984.

[Tso13] E. Tsotsas. “M7 Wärmeleitung und Dispersion in durchströmten Schüttungen”. In: *VDI-Wärmeatlas*. Ed. by VDI e. V. Berlin, Heidelberg: Springer, 2013, pp. 1517–1534.

[Wat+17] N. Watanabe, G. Blöcher, M. Cacace, S. Held and T. Kohl. *Geoenergy Modeling III*. Springer-Verlag GmbH, 2017.

[WC76] W. Wentworth and E. Chen. “Simple thermal decomposition reactions for storage of solar thermal energy”. In: *Solar Energy* 18.3 (1976), pp. 205–214.

[Won+01] W. Wongsuwan, S. Kumar, P. Neveu and F. Meunier. “A review of chemical heat pump technology and applications”. In: *Applied Thermal Engineering* 21.15 (2001), pp. 1489–1519.

[YS02] L. Yong and K. Sumathy. “Review of mathematical investigation on the closed adsorption heat pump and cooling systems”. en. In: *Renewable and Sustainable Energy Reviews* 6.4 (2002), pp. 305–338.

[YWW13] N. Yu, R. Wang and L. Wang. “Sorption thermal storage for solar energy”. In: *Progress in Energy and Combustion Science* 39.5 (2013), pp. 489–514.

Symbols

Throughout the book bold face symbols denote tensors and vectors. Normal face letters represent scalar quantities.

Greek Symbols

α_T	Thermal expansion coefficient	K^{-1}
ϕ	Volume fraction	–
$\mathit{\Gamma}$	Surface with dimension one less than $\mathit{\Omega}$	
$\lambda, \boldsymbol{\lambda}$	Thermal conductivity (tensor)	$\mathrm{W\,m^{-1}\,K^{-1}}$
μ	Dynamic viscosity	$\mathrm{Pa\,s}$
$\mathit{\Omega}$	Domain, volume of some dimension	
$\hat{\varrho}$	Mass production	$\mathrm{kg\,m^{-3}\,s^{-1}}$
ϱ	Mass density	$\mathrm{kg\,m^{-3}}$
$\boldsymbol{\sigma}$	Mechanical stress	Pa
τ	Tortuosity	–

Indices

0	Initial value
α	Placeholder for a specific phase
ads	Adsorbate
adv	Advection
air	Air
amb	Ambient conditions
bed	Adsorbent bed
dry	Dry adsorbent, i. e., entirely in desorbed state
eff	Effective property
eq	In equilibrium state
end	Final state
G	Gas phase
in	Inlet

out	Outlet
p	Property of an adsorbent pellet
pore	A pore in the adsorbent pellet
R	Real/intrinsic quantity
S	Solid phase
V	Vapour

Roman Symbols

$\mathbf{1}$	Identity tensor of second rank	
A_m	Specific differential work of adsorption	$\mathrm{J\,kg^{-1}}$
C	Adsorbent loading	–
c_p	Specific isobaric heat capacity	$\mathrm{J\,kg^{-1}\,K^{-1}}$
$\boldsymbol{d}$	Diffusion velocity	$\mathrm{m\,s^{-1}}$
d	Diameter	m
$\boldsymbol{D}$	Diffusivity tensor	$\mathrm{m^2\,s^{-1}}$
D_{comb}	Combined diffusion coefficient	$\mathrm{m^2\,s^{-1}}$
D_{cont}	Continuum diffusion coefficient	$\mathrm{m^2\,s^{-1}}$
D_{Kn}	Knudsen diffusion coefficient	$\mathrm{m^2\,s^{-1}}$
$\hat{e}$	Total internal energy production	$\mathrm{J\,m^{-3}\,s^{-1}}$
h	Specific enthalpy	$\mathrm{J\,kg^{-1}}$
$\mathit{\Delta} h_{\mathrm{ads}}$	Specific enthalpy of adsorption	$\mathrm{J\,kg^{-1}}$
$\dot{H}_{\mathrm{adv}}$	Advective heating power	W
$\mathit{\Delta} H_{\mathrm{adv}}$	Integral amount of advectively transferred heat	J
$\mathit{\Delta} h_{\mathrm{V}}$	Specific enthalpy of evaporation	$\mathrm{J\,kg^{-1}}$
h_{wall}	Wall heat transfer coefficient	$\mathrm{W\,m^{-2}\,K^{-1}}$
$k, \boldsymbol{k}$	Intrinsic permeability (tensor)	$\mathrm{m^2}$
$\dot{\boldsymbol{m}}$	Mass flux density	$\mathrm{kg\,m^{-2}\,s^{-1}}$
M	Molar mass	$\mathrm{kg\,mol^{-1}}$
m	Mass	kg
$\dot{m}$	Mass flux density, component normal to some surface	$\mathrm{kg\,m^{-2}\,s^{-1}}$
$\boldsymbol{n}$	Outwards oriented unit normal vector	–
$\hat{\boldsymbol{p}}$	Direct momentum production	$\mathrm{kg\,m^{-2}\,s^{-2}}$
p	Pressure	Pa
p_{sat}	Saturation/equilibrium vapour pressure	Pa
$\boldsymbol{q}$	Heat flux density	$\mathrm{W\,m^{-2}}$
R	Universal gas constant	$\approx 8.314\,\mathrm{J\,mol^{-1}\,K^{-1}}$
r	Radius, radial coordinate	m
$\mathit{\Delta} s$	Specific entropy of adsorption	$\mathrm{J\,kg^{-1}\,K^{-1}}$
$\hat{\boldsymbol{s}}$	Total momentum production	$\mathrm{kg\,m^{-2}\,s^{-2}}$
T	Temperature	K
t	Time	s
$\hat{u}$	Direct internal energy production	$\mathrm{J\,m^{-3}\,s^{-1}}$
u	Specific internal energy	$\mathrm{J\,kg^{-1}}$
$\boldsymbol{v}$	Velocity	$\mathrm{m\,s^{-1}}$

$\boldsymbol{v}_{\mathrm{Darcy}}$	Darcy/seepage/superficial/empty tower velocity	$\mathrm{m^3\,m^{-2}\,s^{-1}}$
W	Specific adsorbed volume	$\mathrm{m^3\,kg^{-1}}$
$x_{m\mathrm{V}}$	Vapour mass fraction: Vapour mass per mass of moist air	–
$x_{n\mathrm{V}}$	Vapour mole fraction	–

The abstract quantities $\mathcal{G}$, $\mathcal{F}$, $\mathcal{S}$, and $\hat{\mathcal{P}}$, as well as their densities γ, $\boldsymbol{\varphi}$, σ, and $\hat{\pi}$, respectively, occurring in Eqs. (2.3) and (2.4) have been omitted from the list above.