Simula SpringerBriefs on Computing 18

Jørgen S. Dokken · Kent-Andre Mardal · Marie E. Rognes · Lars Magnus Valnes · Vegard Vinje *Editors*

Mathematical Modelling of the Human Brain II

From Glymphatics to Deep Learning

simula

OPEN ACCESS

Springer

Simula SpringerBriefs on Computing

Volume 18

About this series

In 2016, Springer and Simula launched the book series *Simula SpringerBriefs on Computing*, which aims to provide introductions to selected research topics in computing. The series provides compact introductions for students and researchers entering a new field, brief disciplinary overviews of the state-of-the-art of select fields, and raises essential critical questions and open challenges in the field of computing. Published by SpringerOpen, all *Simula SpringerBriefs on Computing* are open access, allowing for faster sharing and wider dissemination of knowledge.

Simula Research Laboratory is a leading Norwegian research organization which specializes in computing. Going forward, the book series will provide introductory volumes on the main topics within Simula's expertise, including communications technology, software engineering and scientific computing.

By publishing the *Simula SpringerBriefs on Computing*, Simula Research Laboratory acts on its mandate of emphasizing research education. Books in this series are published by invitation from one of the series editors. Authors interested in publishing in the series are encouraged to contact any member of the editorial board.

Jørgen S. Dokken · Kent-Andre Mardal ·
Marie E. Rognes · Lars Magnus Valnes ·
Vegard Vinje
Editors

Mathematical Modelling of the Human Brain II

From Glymphatics to Deep Learning

simula

Springer

Editors
Jørgen S. Dokken
Simula Research Laboratory
Oslo, Norway

Kent-Andre Mardal
Simula Research Laboratory
Oslo, Norway

Marie E. Rognes
Simula Research Laboratory
Oslo, Norway

Lars Magnus Valnes
Department of Mathematics
University of Oslo
Oslo, Norway

Vegard Vinje
Department of Data Science and Analytics
Norwegian Business School
Oslo, Norway

ISSN 2512-1677 ISSN 2512-1685 (electronic)
Simula SpringerBriefs on Computing
ISBN 978-3-032-00678-3 ISBN 978-3-032-00679-0 (eBook)
https://doi.org/10.1007/978-3-032-00679-0

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Series Foreword

Dear reader,

Scientific research is increasingly interdisciplinary, and both students and experienced researchers often face the need to learn the foundations, tools, and methods of a new research field. This process can be quite demanding, and typically involves extensive literature searches and reading dozens of scientific papers in which the notation and style of presentation varies considerably. Since the establishment of this series in 2016 by founding editor-in-chief Aslak Tveito, the briefs in this series have aimed to ease the process by introducing and explaining important concepts and theories in a relatively narrow field, and to outline open research challenges and pose critical questions on the fundamentals of that field. The goal is to provide the necessary understanding and background knowledge and to motivate further studies of the relevant scientific literature. A typical brief in this series should be around 100 pages and should be well suited as material for a research seminar in a well-defined and limited area of computing.

We publish all items in this series under the SpringerOpen framework, as this allows authors to use the series to publish an initial version of their manuscript that could subsequently evolve into a full-scale book on a broader theme. Since the briefs are freely available online, the authors do not receive any direct income from the sales; however, remuneration is provided for every completed manuscript. Briefs are written on the basis of an invitation from a member of the editorial board. Suggestions for possible topics are most welcome and can be sent to sundnes@simula.no.

March 2023

Dr. Joakim Sundnes
Editor-in-Chief
Simula Research Laboratory
sundnes@simula.no

Dr. Martin Peters
Executive Editor Mathematics
Springer Heidelberg, Germany
martin.peters@springer.com

Series Editor for this Volume

Joakim Sundnes, Simula Research Laboratory, Oslo, Norway

Foreword

Many college students study neuroscience as a means of learning about themselves. The curious student will ask questions such as "How can I speed up learning?" "Why do I have to sleep?" "What is the biological basic of love?" The ambitious student will soon find both hypotheses and literature addressing these fundamental questions. In fact, the answers will reflect the perspectives of the many subfields of neuroscience. Psychological science will describe and attempt to explain the factors and processes underlying memory formation, sleep, and feelings. A basic neuroscientist will ascribe the processes to circuit function and the role of neuromodulators. The human brain project even predicted that we would understand brain function if it were possible to map all neurons and their connections. The latter perspective has fostered general interest in neurocomputational processes and further distanced us from the biological necessity of the living brain. Altogether, college students will find a myriad of answers to their simple questions, based on decades of consideration and careful studies.

One aspect of brain function that has rarely been addressed, however, is the physical restriction of the brain. The brain is a soft organ suspended in cerebrospinal fluid and encased by the skull's rigid bony structure. Its billions of neurons work together to provide its computational power, and yet every single cell has its own requirements for a constant supply of energy metabolites and waste removal. There are limitations to what these cells can tolerate and how they respond to changes in local fluid flow, variations in both hydrostatic and osmotic pressure, and consequent fluctuations in extracellular ion composition. How do neurons deal with the unique viscoelastic properties of the brain, which largely lacks a fibrous extracellular matrix? College students will find a sparse literature and no answers or models. The inherent issues associated with obtaining simple physical measures of pulsatile pressure, tissue resistance, and simple flow patterns make mathematical modeling an invaluable tool in defining the physical restrictions of the living brain. These issues are further complicated by the state of brain activity, that is, the sleep–wake cycle, as well as aging and disease, which can have dramatic effects on glymphatic transport.

This book explores this fundamentally new field that seeks to understand the relationships between the brain's physical state and internal fluid dynamics and their effects on brain function. Little prior literature exists for several of the topics presented in this volume, which truly braves a new field. The curious student can thus begin to make links between memory, the need for sleep, the vagaries of affect and emotion, and the ravages of cognitive decline with aging to the fluid dynamics and biophysics of glymphatic clearance.

Rochester *Maiken Nedergaard*

Preface

It was in the fall of 2013 that we began to develop a strong interest in the mathematical and computational modeling of the brain and brain fluids. What grabbed our attention were the then recently published findings of the Nedergaard Lab pointing to the role of fluid flow and transport in brain clearance, with its potential implications for neurodegenerative disease and sleep. Our scientific background was not in clinical or basal medicine or even in computational neuroscience, but in numerical methods for solving partial differential equations (PDEs), soft tissue mechanics, computational fluid dynamics, scientific computing, PDE-constrained optimization, and numerical linear algebra. But we envisioned that these techniques could provide a new—and hopefully refreshingly different—set of tools to better understand the brain's glymphatic system and its importance.

As we started to explore this research field, we faced numerous numerical and computational challenges associated with modeling the human brain. Thanks to our clinical collaborators Per Kristian Eide, Geir Ringstad, and others at Oslo University Hospital, Rikshospitalet, we had access to a rich and nearly unique data set of magnetic resonance (MR) images from several patient cohorts. While going from image volumes to tessellated brain surfaces was relatively straightforward, thanks to FreeSurfer and its associated software ecosystem, generating computational meshes suitable for finite element methods from these surfaces was not trivial. Our first book in this series, *Mathematical Modelling of the Human Brain: From Magnetic Resonance Images to Finite Element Simulation*, resulted from our experiences and aimed at lowering the entry bar for others interested in exploring this fascinating field.

Since then, the mathematical and computational modeling of the brain's waterscape has evolved and developed substantially. In silico studies now form an integral and integrated pillar complementing experimental and clinical approaches. It is our hope that the chapters of this book can provide an introduction to state-of-the-art tools and/or the domain-specific translation of numerical or machine learning–based techniques for junior and senior researchers interested in venturing further into the PDE- or network-based modeling of the human brain.

This book consists of 10 self-contained and relatively short chapters, each intended as a rapid introduction to a key problem or topic.

- Chapter 1 gives a brief and biased introduction to the physics of brain physiology, describing the brain as a soft, elastic, and porous medium surrounded by pulsat-

ing fluids and featuring a complex interplay between mechanical, chemical, and electrical forces.

- Chapter 2 describes aspects related to creating and marking computational meshes of both the brain and the surrounding cerebrospinal fluid spaces, with an emphasis on resolving complex geometric features and the toolkit SVM-Tk.
- Chapter 3 combines these meshing tools with the FEniCS finite element software to model and simulate the flow of cerebrospinal fluid in the subarachnoid space.
- To illustrate a wider range of available tools, chapter 4 presents an extended meshing and simulation pipeline for the brain and its fluid environment as a fluid–structure interaction problem.
- Chapter 5 turns to the interplay between imaging and modeling molecular enrichment and clearance within the brain and offers a step-by-step guide for the mapping between the signal intensities produced by contrast-enhanced MR images and discrete concentrations.
- Chapter 6 turns to the topic of deep learning and describes a convolutional neural network pipeline for predicting signal intensities from such contrast-enhanced MR images.
- Building on the ideas introduced in chapters 5 and 6, chapter 7 presents a physics-informed neural network model of molecular transport in the brain.
- Chapter 8 presents a more comprehensive PDE-based model of brain molecular transport as an example of multi-compartmental spatial modeling of transport.
- Still within the topic of modeling molecular transport, chapter 9 compares a set of inverse modeling approaches for estimating transport parameters within the brain.
- Finally, chapter 10 gives an introduction to graph-based models of neurotoxic protein spread as a starting point for the mathematical modeling and simulation of the role of glymphatics in neurodegeneration.

We would also like to express our gratitude to all the authors and reviewers contributing to this book and its accompanying software. Finally, we acknowledge funding and support from Stiftelsen Kristian Gerhard Jebsen via the K.G. Jebsen Centre for Brain Fluid Research; the Research Council of Norway via FRIPRO grants #324239 (EMIx), #300305 (SciML), and #301013 (Alzheimers physics); Wellcome via Wellcome Discretionary Award 313298/Z/24/Z; the national infrastructure for computational science in Norway, Sigma2, via grants #NN8049K and #NN9279K; and the European Research Council under grant #101141807 (aCleanBrain) .

Oslo, January 2025 *J. S. Dokken, K.-A. Mardal, M. E. Rognes, L. M. Valnes, V. Vinje*

Contents

1 **From brain physiology to brain physics** 1
Marie E. Rognes and Kent-Andre Mardal and Lars Magnus Valnes and Vegard Vinje

2 **Meshing the intracranial compartments: The cerebellum, cerebrum, brainstem, and cerebrospinal fluid** 13
Lars Magnus Valnes and Kent-Andre Mardal

3 **Segmenting, meshing, and modeling CSF spaces** 35
Martin Hornkjøl, Lars Magnus Valnes and Jørgen S. Dokken

4 **The pulsating brain: An interface-coupled fluid–poroelastic interaction model of the cranial cavity** 47
Marius Causemann, Vegard Vinje, and Marie E. Rognes

5 **Quantifying cerebrospinal fluid tracer concentration in the brain** . . . 63
Bastian Zapf, Lars Magnus Valnes, Kent-Andre Mardal, and Ludmil Zikatanov

6 **Signal increase ratio prediction with CNNs** 77
Marius Zeinhofer and Kent-Andre Mardal

7 **Estimating molecular transport parameters using inverse PDE models** 87
Bastian Zapf, Marius Zeinhofer, and Kent-Andre Mardal

8 **Two-compartment modeling of tracer transport in the brain** 105
Jørgen N. Riseth, Timo Koch, and Kent-Andre Mardal

9 **An introduction to identifying velocity fields from contrast imaging via PDE-constrained optimization** 123
Marie E. Rognes

10 **An introduction to network models of neurodegenerative diseases** . . . 133
Georgia S. Brennan and Alain Goriely

Acronyms

BBB Blood–brain barrier

CNN Convolutional neural network

CSF Cerebrospinal fluid

CSG Constructive solid geometry

DTI Diffusion tensor image

EEG Electroencephalogram

fMRI Functional magnetic resonance imaging

ICP Intracranial pressure

iNPH Idiopathic normal pressure hydrocephalus

ISF Interstitial fluid

MPRAGE Magnetization prepared rapid acquisition gradient echo

MRE Magnetic resonance elastography

OCD Optimal convection–diffusion

OMT Optical flow and optimal mass transport

rOCD Reduced optimal convection–diffusion

ROI Region of interest

SAS Subarachnoid space

SGD Stochastic gradient descent

SIR Signal increase ratio

SLYM Subarachnoid lymphatic-like membrane

TBI Traumatic brain injury

Chapter 1

From brain physiology to brain physics

Marie E. Rognes and Kent-Andre Mardal and Lars Magnus Valnes and Vegard Vinje

Abstract This note provides an accessible introduction to key aspects of the mathematical modeling of brain fluid dynamics and solid mechanics in the context of brain glymphatics. We touch upon the different modes and cycles of brain pulsatility, the brain's elastic properties, the brain's fluid environment and fluid networks, and the chemical–mechanical interplay induced by osmotics, all under both physiological and pathological conditions.

1.1 Introduction

In the early 2010s, groundbreaking discoveries at the interface between neurology and neuroscience dramatically increased the interest in and relevance of the mathematical modeling of human brain mechanics. By tracking the in vivo movement of tracer molecules in the brains of mice, Xie et al. (2013) observed that molecules cleared rapidly in sleeping mice but not in awake mice. Moreover, the tracer movement suggested that the perivascular spaces, that is, the cavities filled with cerebrospinal fluid (CSF) that run alongside the blood vessels, are deeply involved in the transportation processes. Six years later, Fultz et al. (2019) revealed intimate connections between waves of blood flow, CSF flow, and neural activity in and around the sleeping brain. These findings introduced the idea that sleep drives molecular clearance from the human brain, that brain clearance is key to why we sleep, and, moreover, that brain clearance may be fundamentally associated with the presence of toxic metabolites and neurodegeneration.

In the years since, the mathematical modeling and simulation of the physiology and physics of the brain have emerged as key enabling technologies for testing biological hypotheses, quantifying hidden mechanisms, and resolving major controversies within these fields. This chapter gives a qualitative and quantitative introduction to fundamental aspects of the solid mechanics and fluid dynamics of the brain and its

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_1

CSF-filled local environment, both at the tissue-to-organ spatial scale and at time scales associated with physiological pulsations. Note that this chapter should not be read as a systematic or exhaustive review, but rather as an accessible introduction highlighting key aspects of the topic.

1.2 The Pulsating Brain

With every heartbeat, blood flows into the brain via the cerebral arteries.[1] The volume of the (adult human) brain and its vasculature increases by about 1 cm^3 followed by an increase of around 3 mmHg in intracranial pressure (ICP) (Baldent, 2014; Eide et al., 2021; Ringstad et al., 2017; Wagshul et al., 2011). The resulting pressure difference between the brain and spinal compartments drives the CSF downward from the subarachnoid space (SAS) surrounding the brain and the ventricles inside the brain, along the spinal canal. The brain, that is, the brain parenchyma (or brain tissue) and the CSF and its various compartments are shown in Figure 1.1. Next, the venous blood outflow routes catch up, in turn decreasing the volume of the brain, reducing the ICP, and resulting in flow reversal of CSF from the spinal compartment up into the cranium. For a simulation framework for this 1-Hz fluid–structure interaction phenomenon, see, for example, Causemann et al. (2022).

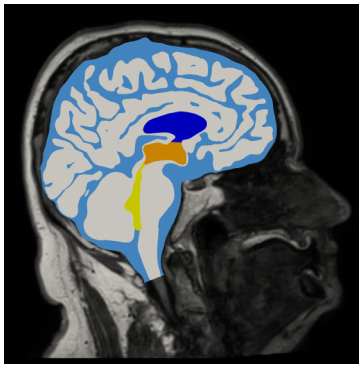


Fig. 1.1 The brain and its fluid surroundings. The colored sagittal section is superimposed on a T1-weighted magnetic resonance image, including the brain parenchyma (light gray) and CSF spaces, showing the subarachnoid space (light blue), fourth ventricle and aqueduct (yellow), third ventricle (orange), and lateral ventricles (dark blue).

The respiratory cycle associated with breathing in (inspiration) and out (exhalation) similarly induces pulsating displacement, pressure, and flow patterns at a frequency of around 0.25Hz in resting humans, with pressure variations on the order of 1 mmHg (Vinje et al., 2019). During natural inspiration, the chest cavity volume increases, reducing the intrathoractic pressure (i.e., pressure in the chest region) and subsequently facilitating venous blood flow from the brain into the heart. This volume is replaced by CSF flowing up from the spinal compartment. Exhalation reverses the

[1] For an introduction to brain anatomy, see, for example, Boron and Boulpaep (2016) and Mardal et al. (2022).

patterns. The picture may, however, involve more complex mechanisms and different breathing techniques inducing different patterns, and the precise mechanisms remain under debate (Lloyd et al., 2020).

Intriguingly, in addition to the cardiac and respiratory cycles, a number of unique wave patterns are associated with the sleeping brain (Lewis, 2021). Using electroencephalography and functional magnetic resonance imaging (blood oxygen level dependent, or BOLD), Fultz et al. (2019) observed large but slow (0.05 Hz) oscillations in CSF flow into the ventricles, strongly anticorrelated with BOLD signal oscillations in the cortical gray matter and in sync with electroencephalographic delta waves. Bojarskaite et al. (2023) also directly demonstrated sleep–wake differences in vasomotion and perivascular dynamics.

1.3 Elastic Properties of the Brain Parenchyma

The brain's composition makes its tissue rheologically complex, with differing elastic properties under different conditions and time scales. Under physiological conditions, that is, during the normal pulsations of the brain associated with cardiac, respiratory, or sleep–wake cycles and the associated time frames of seconds to minutes to hours, the brain can be represented as a linearly elastic or poroelastic medium. That is, assuming an ICP pressure pulsation of 3 mmHg $\approx$ 0.4 kPa would yield low strain (less than 10%) and a linear regime, given a Young's modulus higher than 4 kPa, which is often assumed (Smith and Humphrey (2007)). However, when undergoing large or sudden deformations, the brain's elastic response becomes nonlinear and may result in traumatic brain injury (Griffiths and Budday, 2022). It is estimated that, in the United States, 1 million to 4 million individuals suffer from traumatic brain injury resulting from head injuries in sports alone (Jordan, 2013). In turn, longer time scales (months, years) unveil the viscoelastic features of brain tissue. The excellent reviews by Goriely et al. (2015) and Budday et al. (2020) are good starting points for understanding the nonlinear and viscoelastic properties of the brain, with Su et al. (2023) providing recent insights on the comparison between viscoelastic and poroelastic properties.

In terms of elastic properties, the brain is considered surprisingly soft. How do we quantify this? If u is a vector field that defines the brain displacement relative to a baseline configuration and we model the brain as a linearly elastic medium, then the stress tensor σ is expressed in terms of the strain ε as

$$\sigma(u) = C\varepsilon(u), \quad \varepsilon(u) = \frac{1}{2}\left(\nabla u + \nabla u^T\right), \tag{1.1}$$

where C is a fourth-order stiffness tensor. The nerve fiber tracts making up the brain's white matter define a strongly anisotropic structure, but, interestingly, the elastic response of the brain is reported to not be notably anisotropic (Budday et al., 2017). Therefore, under the assumption of isotropy, the general relation (1.1) reduces

to the isotropic stress–strain relation

$$\sigma(u) = 2\mu\varepsilon(u) + \lambda\nabla \cdot uI, \quad \mu = \frac{E}{2(1+\nu)}, \quad \lambda = \frac{E\nu}{(1-2\nu)(1+\nu)}, \tag{1.2}$$

where μ and λ are the Lamé parameters, expressed here also in terms of the more frequently reported Young's modulus E and Poisson ratio ν. Within this setting, the ICP corresponds to the solid pressure:

$$ICP \approx p_s = -\frac{1}{d}\mathrm{tr}(\sigma),$$

where $d = 1, 2, 3$ and tr are the dimension and trace of σ, respectively.

So, how soft is brain tissue? The literature varies quite a bit. For instance, the early literature review by Smith and Humphrey (2007) reported a range in Young's modulus of 2–60 kPa in mammals. More recent research has suggested smaller values: Budday et al. (2015) reported typical Young's modulus values of 1.4 kPa in gray matter and 1.9 kPa in white matter, along with an incompressibility assumption, that is, $\nu = 0.5$. Part of the reason for the discrepancies could be that the meninges are stiff compared to the parenchyma. In particular, the pia membrane, the innermost membrane surrounding the brain parenchyma, is considered two to three orders of magnitude stiffer (Ozawa et al., 2004). These estimates of the brain's elastic moduli (Budday et al., 2015) rely on classical ex vivo experimental techniques in which, for example, a bovine (cow) brain tissue sample is extracted and subjected to indentation tests. A newer, emerging technique for measuring brain elasticity noninvasively is magnetic resonance elastography (MRE; see Fig. 1.2, (Murphy et al., 2019; Svensson et al., 2021)). Using MRE, Green et al. (2008) identified viscoelastic moduli in the range 2.5–3.1 kPa, with a significantly lower storage modulus in white matter than in gray matter. We note that viscoelastic parameters as measured with MRE are often reported in pascals rather than pascals per second, since the parameters are determined by considering a periodic Helmholtz-type problem in the regime of 1–50 Hz. In this regime, the viscoelasticity parameters scale linearly with frequency (see Green et al. (2008) for details). Using optimal coherence elastography, Gary et al. (2023) found that the (mouse) brain stiffens with age in combination with a decrease in water content.

1.4 Fluid Dynamics in the Brain's Local Environment

Within the cranium, the brain's CSF-filled surroundings include the SAS and the ventricles (Fig. 1.1). The SAS is defined by its outer arachnoid membrane and inner pia membrane and is potentially divided by the subarachnoid lymphatic-like membrane (Møllgård et al., 2023), which closely follows the surface of the brain and widens out into larger cisterns, especially around the cerebellum (Boron and Boulpaep, 2016). The SAS in adult humans is typically 1–5 mm wide, with a volume

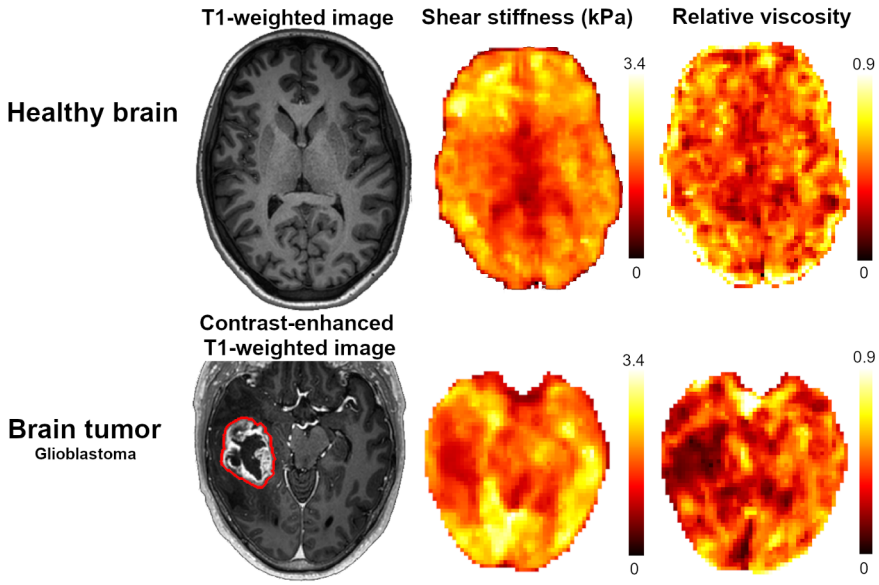


Fig. 1.2: Anatomical T1-weighted images and MRE images for a healthy subject (top) and a patient with a brain tumor. The MRE maps show the magnitude of the shear modulus (shear stiffness in kilopascals) and the shear phase angle (relative viscosity).

of 125 mL (Sakka et al., 2011). The four ventricles include two lateral ventricles, the third and fourth ventricles, with a total volume of ~25 mL. The ~15-mm-wide cerebral aqueduct connects the third and fourth ventricles, while the lateral and median apertures connect the fourth ventricle with the SAS at the cisterna magna. CSF is a clear, colorless fluid that is composed of 99% water; electrolyte ions, including sodium (Na^+), chloride (Cl^-), potassium (K^+), magnesium (Mg^{2+}), and calcium (Ca^{2+}); as well as glucose and plasma proteins, with an osmolarity of ~290 mOsm/L and a pH of 7.33. CSF is thought to be mainly produced in the choroid plexus lining the cerebral ventricles at a rate resulting in total CSF renewal around every six hours in humans (MacAulay et al., 2022; Wichmann et al., 2022). The outflux routes of CSF are still under debate and may vary between species (see, e.g., Hornkjøl et al. (2022) and references therein).

Clinically, CSF flow can be measured in the ventricles, cerebral aqueduct, or craniocervical junction using phase contrast magnetic resonance imaging or other techniques. Typical flow speeds are 3–5 cm/s in the third ventricle (Dreha-Kulaczewski et al., 2015), 4–6 cm/s in the narrower cerebral aqueduct, and 1–3 cm/s at the craniocervical junction (Eide et al., 2021).

Within the SAS, the flow of CSF is well represented by the incompressible Navier–Stokes equations for the velocity v and fluid (or intracranial) pressure p:

$$\rho v_t + v \cdot \nabla v + \nabla \cdot (\mu \varepsilon(v) - pI) = 0, \tag{1.3a}$$

$$\nabla \cdot v = 0, \tag{1.3b}$$

with dynamic viscosity $\mu \approx 0.697 \times 10^{-3}$ Pa s and density $\rho \approx 10^3$ kg/m^3 at body temperature. Often, the low Reynolds and Womersley number approximations may be highly appropriate, in which case the nonlinear Navier–Stokes equations above reduce to the linear Stokes equations by ignoring the term $v \cdot \nabla v$.

1.5 The Brain's Fluid Networks and Permeability

In addition to its fluid surroundings, the brain parenchyma itself is 80% water and permeated by a number of fluid networks. The extracellular space between the brain cells is largely contiguous and filled with interstitial fluid (ISF), a fluid similar in composition to the CSF. The vasculature, filled with blood and separated from the brain tissue by the blood–brain barrier, defines one or several additional fluid networks. In addition, perivascular spaces are hypothesized to enclose the cerebral vasculature within the subarachnoid space and within the parenchyma. The properties of these spaces, including their structure and function, shape and size, and extent of penetration into the brain, are still debated (Bakker et al., 2016; Bohr et al., 2022).

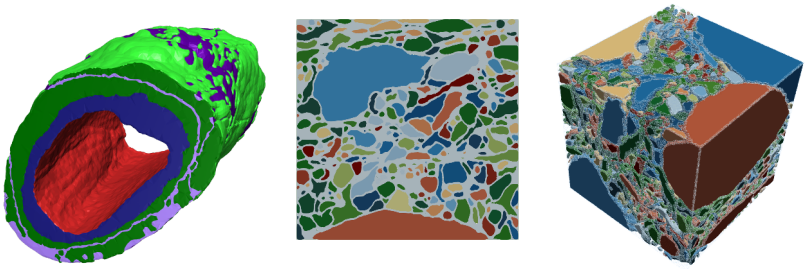


Fig. 1.3: Fluid networks within the brain. Shown here are the perivascular space (dark blue) surrounding an arteriole (left) and the extracellular spaces (in gray) between brain cells in a tissue slice (middle) and a tissue cube (right) (Causemann, 2023).

Interestingly, the brain parenchyma (but not the brain meninges) lacks lymphatic vessels (Aspelund et al., 2015; Louveau et al., 2015). The brain's alternative pathways for solute influx and clearance, termed the brain's *glymphatics*, is an active topic of research (Bohr et al., 2022). The glymphatic concept was introduced by Iliff et al. (2012) in 2012, extending upon previous findings on perivascular pathways (Hadaczek et al., 2006; Rennels et al., 1985) and bulk ISF flow (Abbott, 2004; Cserr and Ostrach, 1974) dating back to the 1970s. The original glymphatic theory stipulates that i) CSF flows in via periarterial spaces, mixes with ISF and flows through the brain

interstitial space, and is cleared via perivenous spaces; ii) interstitial bulk flow depends on transmembrane water movement in astrocytes; and iii) fluid flow is driven by cerebral pulsations.

The parenchyma is compartmentalized into several different compartments at different scales, where we mention in particular the intra- and extracellular compartments, vasculature, and perivascular space. To model the interaction between the fluid flow between these compartments and the elastic response of the brain under the influence of physiological pulsations, Tully and Ventikos (2011) proposed applying multiple-network poroelastic theory (MPET). For a given set of networks $1, 2, \ldots, I$, the multiple-network poroelasticity equations describe the brain displacement u and pressure p_i in each fluid network i via the governing equations

$$\rho u_t - \nabla \cdot (\sigma(u) - \sum_{i=1}^{I} \alpha_i p_i I) = f, \tag{1.4a}$$

$$c_i p_{i,t} + \alpha_i \operatorname{div} u - \nabla \cdot K_i \nabla p_i + T(p) = g_i, \quad \mathrm{i} = 1, 2, \ldots, \mathrm{I}, \tag{1.4b}$$

where $\rho \approx 1.046 \times 10^3$ kg/m^3 is the density of the brain tissue, σ is the elastic stress tensor (see Eq (1.1)), α_i is the Biot–Willis coefficient, c_i is a specific storage coefficient, and K_i is hydraulic conductance, the latter three for each network i. In the case $I = 1$, (1.4) reduces to Biot's equations. The permeability of the extracellular space in neuropil (an area of the brain representative of tissue without vasculature and, as such, a model for extracellular space) is relatively low, with estimates on the order of 10 nm^2 (Holter et al., 2017) to 2500 nm^2 (Smith and Humphrey, 2007). Less is known about the specific storage coefficient and Biot–Willis coefficient of the extracellular space, perivascular spaces, and vasculature. The MPET equations and organ-level modeling of brain perfusion and glymphatics have seen a surge of interest since the late 2010s (e.g., Causemann et al., 2022; Corti et al., 2023; Guo et al., 2020; Kraus et al., 2023; Piersanti et al., 2023).

1.6 Fluid Mechanics and Osmotics in the Brain at the Microscale

At the microscale, in the extracellular space and within and between brain cells such as, for example, in astrocyte networks, brain fluid mechanics and electrophysiology become intrinsically intertwined. The electrical activity of brain nerve cells rely on the rapid movement of ions such as sodium and potassium across the neuronal membrane, altering the electrical potential gradient between the extracellular and intracellular spaces. However, the presence of an ionic gradient also induces an osmotic gradient, leading to water movement across the cellular membranes, cellular swelling, and potential water movement within extracellular and intracellular compartments. Understanding and modeling the electro-chemo-mechanical interplay within the brain is an emerging research field, and we refer the interested reader

to, for example, Rasmussen et al. (2020) and Stra et al. (2023) as recent points of departure for the neurophysiology and computational modeling, respectively.

1.7 Brain Fluid Dynamics and Pathology

Changes in CSF dynamics or in the mechanical properties of the brain are also associated with neurological disorders or neurodegenerative disease. A condition associated with abnormal CSF flow is idiopathic normal pressure hydrocephalus, a form of dementia associated with an increase in the ventricular volume. The pathophysiology of idiopathic normal pressure hydrocephalus is disputed, but theories point to alterations in CSF dynamics such as CSF production or absorption (Reeves et al., 2020). Importantly, patients with Alzheimer's disease show altered CSF dynamics (Schubert et al., 2019). Protein accumulations such as Aβ plaques or CSF-τ tangles are hallmarks of Alzheimer's disease (Fjell et al., 2014), and there is frequent buildup of Aβ in vascular spaces (Thal et al., 2008). Such buildup increases arterial stiffness and reduces arterial pulsatility (Hughes et al., 2013), which may be crucial for CSF flow and exchange along perivascular spaces (Iliff et al., 2012). Generally, brain stiffness is sensitive to demyelination, as encountered, for example, in multiple sclerosis (decreased stiffness), dementia (decreased stiffness in Alzheimer's disease), and tumors (Murphy et al., 2019).

Brain fluid dynamics are also altered in cancer and stroke Eide et al. (2024). The blood–brain barrier limits the transport of toxic substances from the blood into the brain (Hammarlund-Udenaes et al., 2014). In patients with glioma (a form of brain cancer), this barrier is partially disrupted, resulting in at least three separate fluid compartments where substances may be exchanged (blood, ISF, and CSF). Furthermore, it has been shown that perivascular spaces are an important invasion pathway for glioma cells (Cuddapah et al., 2014), highlighting the potential role of the glymphatic system for the development of disease. Finally, glymphatic dysfunction has been associated with the pathophysiology of brain edema, blood–brain barrier disruption, neuroinflammation, and neuronal cell death after stroke (Rasmussen et al., 2018).

1.8 Conclusions and Outlooks

Understanding the human brain is arguably one of the greater scientific challenges of our century. Inspired by Hilbert's (1902) mathematical problems, several lists of unsolved problems have been introduced (Adolphs, 2015; Van Hemmen and Sejnowski, 2005). It is striking that most of the challenges listed are related to the computational and cognitive aspects of the brain, specifically how neurons work, while the basic mechanics of the brain's unique physiology have received comparably

less attention. However, the glymphatics concept (Iliff et al., 2012) and its relation to sleep (Xie et al., 2013) and various neurological diseases (Rasmussen et al., 2018) have revitalized the field of brain biomechanics, with research activity increasing year by year.

Acknowledgements We express our sincere gratitude to Siri Flgstad Svensson for the graphical illustrations and captions displayed in Figure 1.2, Marius Causemann for sharing data and images for Section 1.5, and Ada J. Ellingsrud, whose feedback greatly enhanced our presentation.

References

Abbott N (2004) Evidence for bulk flow of brain interstitial fluid: significance for physiology and pathology. Neurochemistry International 45(4):545–552, doi:10.1016/j.neuint.2003.11.006, role of Non-synaptic Communication in Information Processing

Adolphs R (2015) The unsolved problems of neuroscience. Trends in Cognitive Sciences 19(4):173–175, doi:10.1016/j.tics.2015.01.007

Aspelund A, Antila S, Proulx ST, Karlsen TV, Karaman S, Detmar M, Wiig H, Alitalo K (2015) A dural lymphatic vascular system that drains brain interstitial fluid and macromolecules. Journal of Experimental Medicine 212(7):991–999, doi:10.1084/jem.20142290

Bakker ENTP, Bacskai BJ, Arbel-Ornath M, Aldea R, Bedussi B, Morris AWJ, Weller RO, Carare RO (2016) Lymphatic Clearance of the Brain: Perivascular, Paravascular and Significance for Neurodegenerative Diseases. Cellular and Molecular Neurobiology 36(2):181–194, doi:10.1007/s10571-015-0273-8

Baldent O (2014) Imaging of the cerebrospinal fluid circulation, Cambridge University Press, pp 121–138. doi:10.1017/CBO9781139382816.013, editor: Rigamonti, Daniele

Bohr T, et al. (2022) The glymphatic system: Current understanding and modeling. Iscience 25:104987, doi:10.1016/j.isci.2022.104987

Bojarskaite L, et al. (2023) Sleep cycle-dependent vascular dynamics in male mice and the predicted effects on perivascular cerebrospinal fluid flow and solute transport. Nature Communications 14(1):953, doi:10.1038/s41467-023-36643-5

Boron WF, Boulpaep EL (2016) Medical physiology. Elsevier

Budday S, Nay R, de Rooij R, Steinmann P, Wyrobek T, Ovaert TC, Kuhl E (2015) Mechanical properties of gray and white matter brain tissue by indentation. Journal of the Mechanical Behavior of Biomedical Materials 46:318–330, doi:10.1016/j.jmbbm.2015.02.024

Budday S, Sommer G, Birkl C, Langkammer C, Haybaeck J, Kohnert J, Bauer M, Paulsen F, Steinmann P, Kuhl E, Holzapfel G (2017) Mechanical characterization of human brain tissue. Acta Biomaterialia 48:319–340, doi:10.1016/j.actbio.2016.10.036

Budday S, Ovaert TC, Holzapfel GA, Steinmann P, Kuhl E (2020) Fifty Shades of Brain: A Review on the Mechanical Testing and Modeling of Brain Tissue. Archives of Computational Methods in Engineering 27(4):1187–1230, doi:10.1007/s11831-019-09352-w

Causemann M (2023) EMI-meshing: High-quality extracellular-membrane-intracellular meshes of the mouse visual cortex (0.0.1) [Data set]. doi:10.5281/zenodo.8356333

Causemann M, Vinje V, Rognes ME (2022) Human intracranial pulsatility during the cardiac cycle: a computational modelling framework. Fluids and Barriers of the CNS 19(1):1–17, doi:10.1186/s12987-022-00376-2

Corti M, Antonietti PF, Dede L, Quarteroni AM (2023) Numerical modeling of the brain poromechanics by high-order discontinuous Galerkin methods. Mathematical Models and Methods in Applied Sciences 33(08):1577–1609, doi:10.1142/S0218202523500367

Cserr HF, Ostrach L (1974) Bulk flow of interstitial fluid after intracranial injection of Blue Dextran 2000. Experimental Neurology 45(1):50–60, doi:10.1016/0014-4886(74)90099-5

Cuddapah VA, Robel S, Watkins S, Sontheimer H (2014) A neurocentric perspective on glioma invasion. Nature Reviews Neuroscience 15(7):455–465, doi:10.1038/nrn3765

Dreha-Kulaczewski S, Joseph AA, Merboldt KD, Ludwig HC, Gärtner J, Frahm J (2015) Inspiration Is the Major Regulator of Human CSF Flow. Journal of Neuroscience 35(6):2485–2491, doi:10.1523/JNEUROSCI.3246-14.2015

Eide PK, Valnes LM, Lindstrøm EK, Mardal KA, Ringstad G (2021) Direction and magnitude of cerebrospinal fluid flow vary substantially across central nervous system diseases. Fluids and Barriers of the CNS 18(1):16, doi:10.1186/s12987-021-00251-6

Eide PK, Undseth RM, Gjertsen Ø, Valnes LM, Ringstad G, Lindstrøm EK (2024) Significant individual variation in cardiac-cycle-linked cerebrospinal fluid production following subarachnoid hemorrhage. Fluids and Barriers of the CNS 21(1):1–12, doi:10.1186/s12987-024-00587-9

Fjell AM, McEvoy L, Holland D, Dale AM, Walhovd KB (2014) What is normal in normal aging? Effects of aging, amyloid and Alzheimer's disease on the cerebral cortex and the hippocampus. Progress in Neurobiology 117:20–40, doi:10.1016/j.pneurobio.2014.02.004

Fultz NE, Bonmassar G, Setsompop K, Stickgold RA, Rosen BR, Polimeni JR, Lewis LD (2019) Coupled electrophysiological, hemodynamic, and cerebrospinal fluid oscillations in human sleep. Science 366(6465):628–631, doi:10.1126/science.aax5440

Gary RG, Rolland JP, Song W, Nedergaard M, Parker KJ (2023) Fluid compartments influence elastography of the aging mouse brain. Physics in Medicine and Biology 68(9), doi:10.1088/1361-6560/acc922

Goriely A, Geers MGD, Holzapfel GA, Jayamohan J, Jérusalem A, Sivaloganathan S, Squier W, van Dommelen JAW, Waters S, Kuhl E (2015) Mechanics of the brain: perspectives, challenges, and opportunities. Biomechanics and Modeling in Mechanobiology 14(5):931–965, doi:10.1007/s10237-015-0662-4

Green MA, Bilston LE, Sinkus R (2008) In vivo brain viscoelastic properties measured by magnetic resonance elastography. NMR in Biomedicine 21(7):755–764, doi:10.1002/nbm.1254

Griffiths E, Budday S (2022) Finite element modeling of traumatic brain injury: Areas of future interest. Current Opinion in Biomedical Engineering 24, doi:10.1016/j.cobme.2022.100421

Guo L, Vardakis JC, Chou D, Ventikos Y (2020) A Multiple-network Poroelastic Model for Biological Systems and Application to Subject-specific Modelling of Cerebral Fluid Transport. International Journal of Engineering Science 147:103204, doi:10.1016/j.ijengsci.2019.103204

Hadaczek P, Yamashita Y, Mirek H, Tamas L, Bohn MC, Noble C, Park JW, Bankiewicz K (2006) The "perivascular pump" driven by arterial pulsation is a powerful mechanism for the distribution of therapeutic molecules within the brain. Molecular therapy : the journal of the American Society of Gene Therapy 14(1):69–78, doi:10.1016/j.ymthe.2006.02.018

Hammarlund-Udenaes M, et al. (2014) Drug delivery to the brain. AAPS Press, Springer, DOI 10:978–1, doi:10.1007/978-1-4614-9105-7

Hilbert D (1902) Mathematical problems. Bulletin of the American Mathematical Society 8

Holter KE, Kehlet B, Devor A, Sejnowski TJ, Dale AM, Omholt SW, Ottersen OP, Nagelhus EA, Mardal KA, Pettersen KH (2017) Interstitial solute transport in 3D reconstructed neuropil occurs by diffusion rather than bulk flow. Proceedings of the National Academy of Sciences 114(37):9894–9899, doi:10.1073/pnas.1706942114

Hornkjøl M, Valnes LM, Ringstad G, Rognes ME, Eide PK, Mardal KA, Vinje V (2022) CSF circulation and dispersion yield rapid clearance from intracranial compartments. Frontiers in Bioengineering and Biotechnology 10:932469, doi:10.3389/fbioe.2022.932469

Hughes TM, et al. (2013) Pulse wave velocity is associated with β-amyloid deposition in the brains of very elderly adults. Neurology 81(19):1711–1718, doi:10.1212/01.wnl.0000435301.64776.37

Iliff JJ, et al. (2012) A Paravascular Pathway Facilitates CSF Flow Through the Brain Parenchyma and the Clearance of Interstitial Solutes, Including Amyloid β. Science Translational Medicine 4(147):147ra111–147ra111, doi:10.1126/scitranslmed.3003748

Jordan BD (2013) The clinical spectrum of sport-related traumatic brain injury. Nature Reviews Neurology 9:222–230, doi:10.1038/nrneurol.2013.33

Kraus J, Lederer PL, Lymbery M, Osthues K, Schöberl J (2023) Hybridized Discontinuous Galerkin/Hybrid Mixed Methods for a Multiple Network Poroelasticity Model with

Application in Biomechanics. SIAM Journal on Scientific Computing 45(6):B802–B827, doi:10.1137/22M149764X

Lewis LD (2021) The interconnected causes and consequences of sleep in the brain. Science 374(6567):564–568, doi:10.1126/science.abi8375

Lloyd RA, Butler JE, Gandevia SC, Ball IK, Toson B, Stoodley MA, Bilston LE (2020) Respiratory cerebrospinal fluid flow is driven by the thoracic and lumbar spinal pressures. The Journal of Physiology 598(24):5789–5805, doi:10.1113/JP279458

Louveau A, Smirnov I, Keyes TJ, Eccles JD, Rouhani SJ, Peske JD, Derecki NC, Castle D, Mandell JW, Lee KS, Harris TH, Kipnis J (2015) Structural and functional features of central nervous system lymphatic vessels. Nature 523(7560):337–341, doi:10.1038/nature14432

MacAulay N, Keep RF, Zeuthen T (2022) Cerebrospinal fluid production by the choroid plexus: a century of barrier research revisited. Fluids and Barriers of the CNS 19(1):26, doi:10.1186/s12987-022-00323-1

Mardal KA, Rognes M, Thompson T, Valnes L (2022) Mathematical Modeling of the Human Brain: From Magnetic Resonance Images to Finite Element Simulation. Springer, Cham, doi:10.1007/978-3-030-95136-8

Møllgård K, Beinlich FR, Kusk P, Miyakoshi LM, Delle C, Plá V, Hauglund NL, Esmail T, Rasmussen MK, Gomolka RS, et al. (2023) A mesothelium divides the subarachnoid space into functional compartments. Science 379(6627):84–88, doi:10.1126/science.adc8810

Murphy MC, III JH, Ehman RL (2019) MR elastography of the brain and its application in neurological diseases. NeuroImage 187:176–183, doi:10.1016/j.neuroimage.2017.10.008

Ozawa H, Matsumoto T, Ohashi T, Sato M, Kokubun S (2004) Mechanical properties and function of the spinal pia mater. Journal of Neurosurgery: Spine 1(1):122–127, doi:10.3171/spi.2004.1.1.0122

Piersanti E, Rognes ME, Vinje V (2023) Are brain displacements and pressures within the parenchyma induced by surface pressure differences? A computational modelling study. Plos one 18(12):e0288668, doi:10.1371/journal.pone.0288668

Rasmussen MK, Mestre H, Nedergaard M (2018) The glymphatic pathway in neurological disorders. The Lancet Neurology 17(11):1016–1024, doi:10.1016/S1474-4422(18)30318-1

Rasmussen R, O'Donnell J, Ding F, Nedergaard M (2020) Interstitial ions: A key regulator of state-dependent neural activity? Progress in Neurobiology 193:101802, doi:10.1016/j.pneurobio.2020.101802

Reeves BC, Karimy JK, Kundishora AJ, Mestre H, Cerci HM, Matouk C, Alper SL, Lundgaard I, Nedergaard M, Kahle KT (2020) Glymphatic system impairment in alzheimer's disease and idiopathic normal pressure hydrocephalus. Trends in Molecular Medicine 26(3):285–295, doi:10.1016/j.molmed.2019.11.008

Rennels ML, Gregory TF, Blaumanis OR, Fujimoto K, Grady PA (1985) Evidence for a "Paravascular" fluid circulation in the mammalian central nervous system, provided by the rapid distribution of tracer protein throughout the brain from the subarachnoid space. Brain Research 326(1):47–63, doi:10.1016/0006-8993(85)91383-6

Ringstad G, Lindstrøm EK, Vatnehol SAS, Mardal KA, Emblem KE, Eide PK (2017) Non-invasive assessment of pulsatile intracranial pressure with phase-contrast magnetic resonance imaging. PloS one 12(11):e0188896, doi:10.1371/journal.pone.0188896

Sakka L, Coll G, Chazal J (2011) Anatomy and physiology of cerebrospinal fluid. European Annals of Otorhinolaryngology, Head and Neck Diseases 128(6):309–316, doi:10.1016/j.anorl.2011.03.002

Schubert JJ, Veronese M, Marchitelli L, Bodini B, Tonietto M, Stankoff B, Brooks DJ, Bertoldo A, Edison P, Turkheimer FE (2019) Dynamic 11C-PiB PET shows cerebrospinal fluid flow alterations in Alzheimer disease and multiple sclerosis. Journal of Nuclear Medicine 60(10):1452–1460, doi:10.2967/jnumed.118.223834

Smith JH, Humphrey JA (2007) Interstitial transport and transvascular fluid exchange during infusion into brain and tumor tissue. Microvascular Research 73(1):58–73, doi:10.1016/j.mvr.2006.07.001

Su L, Wang M, Yin J, Ti F, Yang J, Ma C, Liu S, Lu TJ (2023) Distinguishing poroelasticity and viscoelasticity of brain tissue with time scale. Acta Biomaterialia 155:423–435, doi:10.1016/j.actbio.2022.11.009

Svensson SF, De Arcos J, Darwish OI, Fraser-Green J, Stors TH, Holm S, Vik-Mo EO, Sinkus R, Emblem KE (2021) Robustness of mr elastography in the healthy brain: Repeatability, reliability, and effect of different reconstruction methods. Journal of Magnetic Resonance Imaging 53(5):1510–1521, doi:10.1002/jmri.27475

Stra MJ, Ellingsrud AJ, Rognes ME (2023) Neural activity induces strongly coupled electro-chemo-mechanical interactions and fluid flow in astrocyte networks and extracellular space – A computational study. PLOS Computational Biology 19(7):1–31, doi:10.1371/journal.pcbi.1010996

Thal DR, Griffin WST, de Vos RA, Ghebremedhin E (2008) Cerebral amyloid angiopathy and its relationship to Alzheimer's disease. Acta neuropathologica 115:599–609, doi:10.1007/s00401-008-0366-2

Tully B, Ventikos Y (2011) Cerebral water transport using multiple-network poroelastic theory: application to normal pressure hydrocephalus. Journal of Fluid Mechanics 667:188–215, doi:10.1017/S0022112010004428

Van Hemmen JL, Sejnowski TJ (2005) 23 problems in systems neuroscience. Oxford University Press

Vinje V, Ringstad G, Lindstrøm EK, Valnes LM, Rognes ME, Eide PK, Mardal KA (2019) Respiratory influence on cerebrospinal fluid flow – a computational study based on long-term intracranial pressure measurements. Scientific Reports 9(1):9732, doi:10.1038/s41598-019-46055-5

Wagshul ME, Eide PK, Madsen JR (2011) The pulsating brain: A review of experimental and clinical studies of intracranial pulsatility. Fluids and Barriers of the CNS 8(1):1–23, doi:10.1186/2045-8118-8-5

Wichmann TO, Damkier HH, Pedersen M (2022) A Brief Overview of the Cerebrospinal Fluid System and Its Implications for Brain and Spinal Cord Diseases. Frontiers in Human Neuroscience 15:850, doi:10.3389/fnhum.2021.737217

Xie L, et al. (2013) Sleep Drives Metabolite Clearance from the Adult Brain. Science 342(6156):373–377, doi:10.1126/science.1241224

Chapter 2

Meshing the intracranial compartments: The cerebellum, cerebrum, brainstem, and cerebrospinal fluid

Lars Magnus Valnes and Kent-Andre Mardal

Abstract Computational modeling can assist researchers to further understand the mechanism in and around the human brain. However, there are significant differences between brain geometries, so macroscopic models can require person-specific meshes. Therefore, we will go through the process of making a patient-specific volumetric mesh from T1-weighted magnetic resonance images. The resulting mesh will include structures such as the cerebellum, subarachnoid space, cisterna magna, and foramen magnum.

2.1 Introduction

The transport and exchange between the brain and the surrounding fluids have been a hot topic of research in the last decade, particularly since the introduction of the glymphatic system and its relevance to dementia (Jessen et al., 2015). Cerebrospinal fluid (CSF) flow in the subarachnoid space (SAS) seems to be of crucial importance for solute transfer within the brain (Hornkjøl et al., 2022) and varies significantly among different diseases (Eide et al., 2021). However, as has become increasingly clear over these last years, the fluid processes are complicated and compartmentalized in the subarachnoid space (Mestre et al., 2018; Møllgård et al., 2023). Additionally, there are pulsations at several different frequencies, such as sleep waves (Bojarskaite et al., 2022; Fultz et al., 2019) and respiration (Vinje et al., 2019), in addition to cardiac pulsations. Our aim here is therefore to build software tools for subject-specific simulations of multiphysics problems where the interactions of the brain (including the cerebellum and brainstem), CSF, and meninges are simulated with methods from scientific computing.

Mardal et al. (2022) discussed techniques for generating surfaces and meshes for FEniCS-based (Alnæs et al., 2015) simulations. The main focus was on meshing based on the default surface reconstructions provided by FreeSurfer (Fischl, 2012),

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_2

namely, the surfaces of the gray and white matter of the cerebrum and corresponding single-physics simulations. Here, we extend the methodology and also consider the CSF of the SAS, cisterna magnum, aqueduct, and foramen magnum, along with the brainstem and cerebellum. Furthermore, we describe the tools to mark volumes, surfaces, and interfaces, along with tools that provide the clean cuts needed for setting boundary conditions at, for instance, the cervicocranial junction in the upper neck. We thus aim to provide, in particular, the software tools for brain-wide simulation of the glymphatic system (Jessen et al., 2015), where the viscous flows of CSF interact with the poroelastic brain within, possibly accounting for multiple compartments of the human brain (Piersanti et al., 2023).

We will also briefly outline new features to the Surface Volume Meshing Toolkit (SVM-Tk) version 2.0.0, which is a Python-based front-end interface to the Computational Geometry Algorithms Library (CGAL) (Fabri et al., 2000). SVM-Tk will be used for correcting surface defects and imperfections with functions designed for brain anatomy and provides a relatively robust mesh generation procedure. It has been used in various publications since its introduction by Mardal et al. (2022) (e.g., Corti et al., 2023; Hornkjøl et al., 2022; Jzsa et al., 2023; Vinje et al., 2023). The documentation for SVM-Tk[1] includes installation instructions, a list of requirements, and an API.

This chapter is divided into three sections, each describing different parts of the workflow for creating volume meshes:

1. Segmenting the intracranial compartments using FreeSurfer to amend and/or supplement the segmentation of the cerebrum (as described by Mardal et al. (2022)).
2. Constructing and repairing surfaces for meshing and a brief description of the updates in SVM-Tk.
3. Creating meshes with sharp edges and unique interface tags needed for boundary conditions in the cervicocranial junction.

We encourage the reader to download and install the dataset and tools pertaining to this chapter, which would allow the reader to test the procedures as we go through them. The dataset `mri2femii-chp2-dataset` can be downloaded from Zenodo (Valnes and Mardal, 2024) together with a configuration file for this chapter. The code and installation procedures are available from Valnes (2024). However, note that FreeSurfer requires a free license to run.[2]

[1] See `https://github.com/SVMTK/SVMTK`.

[2] See `https://surfer.nmr.mgh.harvard.edu/registration.html`.

2.2 Segmenting the Intracranial Compartments

In this section we will explain how to segment the CSF spaces (aqueduct, SAS, cisterna magna) either manually or by T2-weighted magnetic resonance imaging (MRI), along with segmentation of the brainstem, cerebellum, and upper part of the spinal cord.

2.2.1 Initial Segmentation

FreeSurfer is open-source software for MRI-based analysis in widespread use in neuroscience.[3] Our interest here is the semiautomatic segmentation of the brain's structures and regions. An initial segmentation that serves as our starting point is typically provided by the FreeSurfer command `recon-all`. The unfamiliar reader may want to look at Mardal et al. (2022) for a brief introduction, while the review by Fischl (2012) is recommended for more details on FreeSurfer.

The `recon-all` command is typically time-consuming, requiring up to 24 hours, and the segmentation may require manual corrections. Such corrections will not be covered here, since the topic can be quite extensive, and tutorials and courses exist on FreeSurfer.[4]

The FreeSurfer segmentation and surfaces are included in the dataset (Valnes and Mardal, 2024). We can inspect the FreeSurfer output, found in the dataset folder, by executing the following commands:

```
$ export SUBJECT_DIRS=<path_to_dataset>
$ freeview --recon Gonzo
```

This will open the graphic user interface Freeview, which will be used to view the segmentation files. The following are some useful Freeview shortcuts:

- PgUp and PgDn can be used to scroll through the image.
- Alt+C will cycle between loaded images.
- Alt+X, Alt+Y, Alt+Z, and Alt+3 will, respectively, choose sagittal, coronal, axial, and 3D as the primary view.

Figure 2.1 shows the initial segmentation, white matter segmentation, and constructed hemisphere surfaces. Furthermore, we can inspect the different regions by hovering the cursor over regions and reading the names. This allows for simpler investigation of the segmentation, since we can thus find the cerebellum, brainstem, and ventricles of the brain.

[3] See `https://surfer.nmr.mgh.harvard.edu/`.

[4] See `https://surfer.nmr.mgh.harvard.edu/fswiki/FsTutorial`

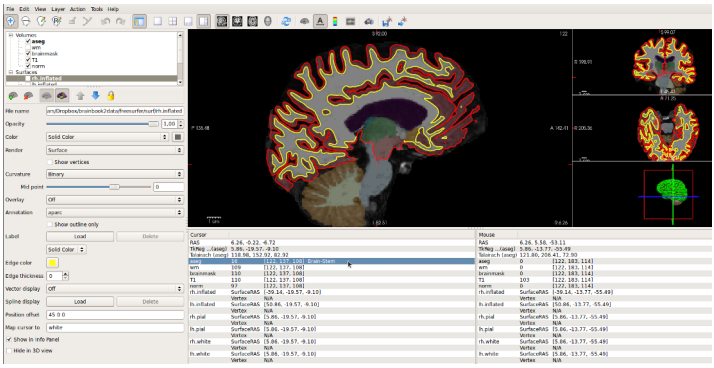


Fig. 2.1: This image shows the Freeview GUI (graphical user interface) with readouts of labels and image intensities observed by hovering over the regions. The yellow and red curves indicate the FreeSurfer output of the white and pial surfaces. The views of the brain correspond to the sagittal view (left), the coronal view (top right), the axial view (middle right), and a 3D representation (bottom right).

2.2.2 Amending the FreeSurfer Segmentation

The surfaces provided by FreeSurfer are 1) the pial surface at the interface between CSF and gray matter and 2) the interface between the gray and white matter (in both hemispheres of the cerebrum). Any other surface that requires a specific tag must be created by the user. Below we address the creation of the surfaces for the internal CSF spaces (ventricles and aqueduct), external SAS CSF space, brainstem, and cerebellum. Note that the various CSF spaces are prone to be thin and are therefore challenging to segment.

2.2.3 The Cerebral Aqueduct

The cerebral aqueduct is a region of interest, particularly for those diagnosed with idiopathic normal-pressure hydrocephalus, since CSF fluid velocities are commonly assessed using phase-contrast MRI in this region (Ringstad et al., 2017; Lindstrm et al., 2018). However, although certain segments are clearly visible, a complete, continuous aqueduct may be challenging to obtain, especially in young people. The size of the aqueduct differs between individuals, and the cross-sectional area ranges from 0.2 to 1.8 mm^2 (Cinalli et al., 2011). Compared to the default 1-mm^3 voxel resolution used for FreeSurfer segmentation, the aqueduct may be too small to segment automatically. Furthermore, the cerebral aqueduct is not labeled with its own value in FreeSurfer segmentation and may be included as a part of the fourth ventricle connecting to the third ventricle. Therefore, if it is missing, the third and

fourth ventricles would appear disconnected in the segmentation, as indicated in Figure 2.3.

Missing aqueducts can often be fixed by including a supplementary T2-weighted image to the `recon-all` command, given the image has sufficient quality and resolution. Otherwise, we can manually segment the cerebral aqueduct using Freeview. Both alternatives are outlined below.

Manual Editing of the Cerebral Aqueduct

We will edit the cerebral aqueduct directly in the segmentation file. We therefore open the segmentation and T1-weighted MRI in Freeview with the following command:

```
$  freeview aseg.mgz:colormap=lut:opacity=0.5 orig.mgz
```

We navigate the cursor to the fourth ventricle in the brainstem and place the marker with a Left Click. We select the action Voxel Edit and optionally set the view to only show an axial MRI slice by selecting 1x1 and axial (see Fig. 2.2). Selecting Voxel Edit will open a window in which we can set the brush value to 15, which is the label value for the fourth ventricle.

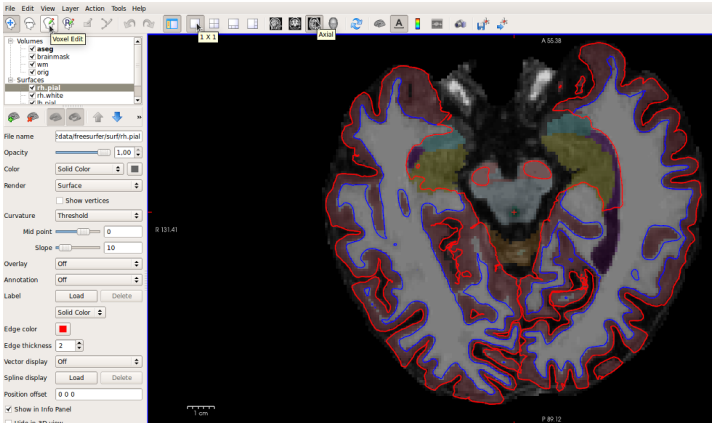


Fig. 2.2: This image shows the Freeview GUI setup for the manual editing of the cerebral aqueduct, with the cursor indicating the step and the red cross marking the cerebral aqueduct.

We have now completed the setup and can begin marking voxels with Left Click, and we can rewind with CTRL + Z. The voxels we want to mark appear dark in the image and are situated inside the brainstem segmentation connecting the third and fourth ventricles.

Registration and Normalization of T2-weighted Images

In the dataset `MRI2FEMii` (Valnes and Mardal, 2024), there is a T2-weighted image that can be used to supplement the segmentation. However, first, the T2-weighted image must be registered to the T1-weighted image.

This can be done automatically during the initialization or by using individual commands from FreeSurfer (with the option `--T2` to specify the T2-weighted image to `recon-all`).

If it has not already been done, the following commands perform the registration:

```
# Convert the T2 image to size 256^3 with voxel size 1 mm^3
mri_convert -odt float -c T2.orignal.mgz T2.conformed.mgz
# bbregister perform within-subject cross-modal registration
bbregister --s ${SUBJECT} --mov T2.original.mgz --lta \
    ↪transforms/T2raw.lta --init-fsl --T2
# Apply transformation to the conformed T2 image.
mri_vol2vol --mov T2.conformed.mgz --targ orig.mgz --lta \
    ↪T2raw.lta --o T2.registered.mgz
```

First, we resample the T2-weighted image to the conformed space of FreeSurfer. This is followed by a registration procedure using `bbregister`. Additionally, we need to change the FreeSurfer environment folder `SUBJECTS_DIR` to the folder that contains the FreeSurfer output. The registration provides the transformation in a file with the suffix `lta`, which we apply to the T2-weighted image to align it with the FreeSurfer output.

Two additional steps can be performed to normalize the values in the T2-weighted image, which gives us the file `T2.norm.mgz`:

```
# Rescale the T2 image.
mri_nu_correct.mni --i T2.registered.mgz --o T2.nu.mgz
mri_normalize T2.nu.mgz T2.norm.mgz
```

These steps are recommended in particular for creating a semiautomatic procedure to apply to many patients. FreeSurfer commonly uses a normalization[5] where the peak in white matter is normalized to be around 110 in T1-weighted images. The normalization procedure does not directly translate to T2-weighted images, but it will normalize the intensities to be within the range from zero to 255. Additionally, note that if you use `recon-all` with `-T2`, the normalization will differ. Therefore, the user is recommended to call `mri_normalize` on `T2.prenorm.mgz` produced by `recon-all`.

[5] See `https://surfer.nmr.mgh.harvard.edu/fswiki/normalization`.

T2-weighted Images Augmentation of the Cerebral Aqueduct

The aqueduct (containing CSF) being mislabeled as the brainstem (label 16) is a common problem. However, with a good T2-weighted image, the aqueduct gives a strong signal (as CSF) and can thus be easily identified. The following script searches for voxels within the brainstem region with a strong signal in the T2-weighted image (more than 40, assuming normalization as above) and sets their label to the fourth ventricle (label 15):

```
# Creates a mask of the brainstem segmentation
mri_binarize --i aseg.mgz --match 16 --o brainstem.mgz
# Creates a mask of the CSF inside the brainstem mask.
mri_binarize --i T2.norm.mgz --min 40 --mask brainstem.mgz \
    ↪--o cerebral.aqueduct.mgz
# Categorize different voxel clusters of the mask
mri_volcluster --in cerebral.aqueduct.mgz --thmin 1 --ocn \
    ↪aqueduct.clusters.mgz
# Create a mask of the largest voxel cluster and set the \
    ↪value to 15
mri_binarize --i aqueduct.clusters.mgz --match 1 --o \
    ↪cerebral.aqueduct.mgz --binval 15
# Combine the mask voxels with the freesurfer segmentation
mri_mask -transfer 15 aseg.modified.mgz cerebral.aqueduct.mgz \
    ↪aseg.modified.mgz
```

We note that a clustering procedure (`mri_volcluster`) is utilized to ensure that the voxels are connected. The result is shown in Figure 2.3.

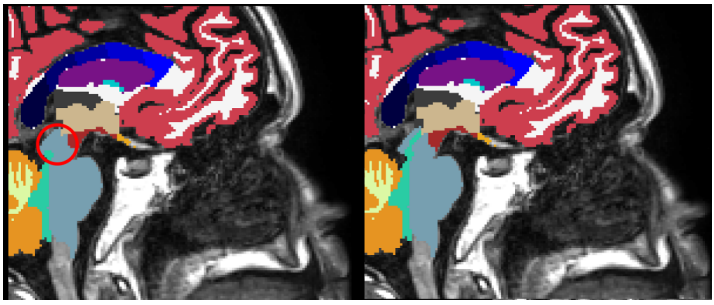


Fig. 2.3: The left panel shows a cerebral aqueduct mislabeled as brainstem (highlighted with a red circle). The right panel shows the cerebral aqueduct after T2-weighted correction.

Segmentation of the SAS using T2-weighted Images

The methodology for segmenting the SAS is similar to for segmenting the cerebral aqueduct using T2-weighted images. However, controlling through visual inspection that the threshold used in the segmentation is appropriate is recommended. The approach is detailed below. Manual inspection of both T1- and T2-weighted images is conveniently performed as follows (see Fig. 2.4):

```
$ freeview -v T1.mgz \
    T2.norm.mgz:colormap=nih:colorscale=40,200
```

To determine the best threshold values, we may adjust the color scale on the left in Freeview. The preset colormap, `nih`, will only show values within the color scale range. Hence, we can investigate different thresholds by adjusting the `min` parameter of the color scale. Setting the `min` parameter to 40 seems to include most of the CSF in the SAS, as shown in Figure 2.4. We then use the same method for the normalized T1-weighted image (`T1.mgz` found in the FreeSurfer output), but we adjust the `max` parameter instead, found to be around 65.

When confident with the thresholds, we use `mri_binarize` to create an initial mask for the CSF of the SAS using the normalized T2 image. Next, we use the output of this command to remove voxels containing tissue or skull, again by using `mri_binarize`. The SAS may contain a large number of small, isolated clusters due to the fact that the space is thin and separated by folds. We therefore combine the SAS with the brain segmentation provided by `aseg.mgz`, using `mri_binarize`. This will ensure that the small SAS clusters are connected with the brain segmentation, which allows us to employ `mri_volcluster` to remove other smaller clusters, such as the vitreous humor (part of the eye). The combination of the brain and SAS segmentation may not fit perfectly, and unmarked voxels might therefore exist inside or at the edge of the mask. We use `mri_morphology` to fill these unmarked voxels to avoid pits in the surface. This will overwrite the original bounding mask, called `dura.mgz`. These steps are done with the following script:

```
# Create a mask of CSF in T2-weighted image.
mri_binarize --i T2.norm.mgz --min 110 --o T2.threshold.mgz
# Subtract any voxels with tissue or skull based on \
    ↪T1-weighted image.
mri_binarize --i T1.mgz --max 65 --mask T2.threshold.mgz --o \
    ↪SAS.mgz
# Merge mask with freesurfer segmentation.
mri_binarize --i aseg.mgz --min 1 --o dura.mgz --merge SAS.mgz
# Categorize different voxel clusters of the mask.
mri_volcluster --in dura.mgz --thmin 1 --ocn Dura.clusters.mgz
# Create a mask of the largest voxel cluster.
mri_binarize --i Dura.clusters.mgz --match 1 --o dura.mgz
# Closes any holes/concave region in the mask.
mri_morphology dura.mgz close 4 dura.mgz
```

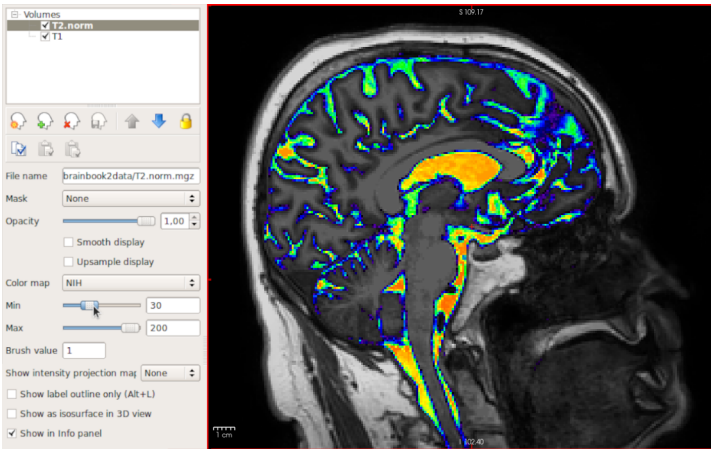


Fig. 2.4: The normalized T2-weighted image with the normalized T1-weighted image in the background.

2.2.4 Extending the Brainstem Segmentation to the Spinal Cord

There are two issues with the brainstem and associated CSF spaces: 1) the FreeSurfer segmentation does not include the full brainstem or spinal cord, as seen in Figure 2.5 (left), and 2) for accurate simulation results, it is preferable to have a smooth interface on which to apply boundary conditions, as seen in Figure 2.5 (right). We note that it is preferable that the cut be normal to the expected flow direction.

The first problem is similar to the problem with the aqueduct described earlier, namely, that a region should be extended along certain signal values if the region is within a surrounding mask. In detail, we start by extracting voxels containing tissue inside mask `dura.mgz`, created in Section 2.2.3, using the threshold of `T1.mgz`. Then, we remove voxels already in the FreeSurfer segmentation (using `mri_binarize`), followed by removing unconnected voxels (using `mri_volcluster`). We set the mask value to 16, labeling it as the brainstem, and transfer the voxels over to the FreeSurfer segmentation file, named `aseg.extended.mgz` (using `mri_mask`). Below we provide a script for generating such a correction:

```
# Creates a mask of tissue inside the dura mask.
mri_binarize --i T1.mgz --min 65 --mask dura.mgz --o \
    ↪brainstem.extended.mgz
```

```
# Subtracts the freesurfer segmentation from the previous mask
mri_binarize --i aseg.mgz --match 0 --mask \
    ↪brainstem.extended.mgz --o brainstem.extended.mgz
# Categorize different voxel clusters of the mask
mri_volcluster --in brainstem.extended.mgz --thmin 1 --ocn \
    ↪brainstem.clusters.mgz
# Create a mask of the largest voxel cluster and set the \
    ↪value to 16
mri_binarize --i brainstem.clusters.mgz --match 1 --o \
    ↪brainstem.extended.mgz --binval 16
# Closes any holes/concave region in the mask.
mri_morphology brainstem.extended.mgz close 4 \
    ↪brainstem.extended.mgz
# Combine the mask voxels with the freesurfer segmentation
mri_mask -transfer 16 aseg.mgz brainstem.extended.mgz \
    ↪aseg.modified.mgz
```

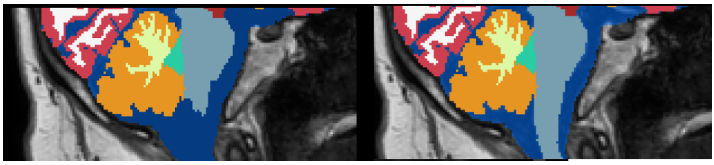


Fig. 2.5: The left panel shows the brainstem not connected with the spinal cord. The right panel shows the brainstem connected with the spinal cord.

2.3 Constructing Surfaces based on Image Segmentation

With the augmented FreeSurfer segmentation, the next step is to create surface triangulations of the various labeled structures. For this process, we will use the FreeSurfer command `mri_binarize`, which produces a binary volume image and/or surface triangulation.

We employ surface smoothing to avoid sharp edges, where `--surf-smooth 0` corresponds to extracting a voxelized surface. The smoothing will cause the surface to no longer match the segmentation, since it will shrink when locally convex and expand when locally concave. This can cause issues when combining surfaces, since they might no longer be connected.

2.3.1 Extracting the SAS Bounding Surface

The SAS consists of a network of narrow passages that can easily collapse due to the smoothing. Therefore, we will not create a surface based on SAS segmentation. Instead, we utilize the fact that the SAS exists inside the bounding mask of `dura.mgz` and create a bounding surface. This means that the SAS can be defined as the difference between the bounding surface and surfaces of the brain segmentation.

```
# Extracting bounding surface based on the inner dura mask.
mri_binarize --i Dura.mgz --match 1 --surf-smooth 5 --surf \
    ↪./bounding_surface.stl
```

2.3.2 Extracting the Cerebellum and Brainstem Surfaces

The cerebellum cortex has a fine-grained surface with sub-resolution sulci and gyri. We neglect the sub-resolution features and consider the surface smooth. We thus utilize the FreeSurfer command `mri_binarize` to create a mask of the cerebellum. We include the fourth ventricle such that the surface union with the brainstem in the next step becomes simpler. We then fill any potential cavities in the mask to avoid problems related to not having simply connected volumes.

```
# Creating a mask of the cerebellum and the fourth ventricle.
mri_binarize --i aseg.modified.mgz --match 7 46 8 47 15 --o \
    ↪cerebellum.mgz
# Fill holes that can result in surface errors.
mri_morphology cerebellum.mgz fill_holes 5 cerebellum.mgz
# Extracting cerebellum surface
mri_binarize --i cerebellum.mgz --match 1 --surf-smooth 5 \
    ↪--surf ./cerebellum.stl
```

We can utilize the same approach as above, but we also use additional segmentation tags for the brainstem and ventricles.

```
# Creating a mask of the brainstem, with thalamus and ventral \
    ↪diencephalon of each hemisphere.
mri_binarize --i aseg.modified.mgz --match 10 28 16 15 14 60 \
    ↪49 --o brainstem.mgz
# Fill holes that can result in surface errors.
mri_morphology brainstem.mgz fill_holes 5 brainstem.mgz
# Extracting brainstem surface
mri_binarize --i brainstem.mgz --match 1 --surf-smooth 5 \
    ↪--surf ./brainstem.stl
```

The following collects all the commands in a bash script:

```
$ bash get_surface --cerebellum --brainstem
```

2.4 Creating Meshes with SVM-Tk

The surfaces created with FreeSurfer are a collection of triangles without a consistent orientation. A consistent orientation is enforced automatically in SVM-Tk using tools from CGAL (Gueziec et al., 1998).

To avoid unphysical behavior in the finite element simulations, we require all cells to be connected through a facet, and not only through a vertex or edge. We utilize the function `validate_mesh` to inspect if there are any hazardous connections in the constructed volume mesh. However, removal of these connections may have a cascading effect, so, instead, it is advisable to have a properly prepared surface mesh before creating the volume mesh. We thus detail below techniques to repair surfaces.

In Section 2.3, we created three surfaces named `brainstem.stl`, `cerebellum.stl`, and `dura.stl`, with the command `get_surface`. We can use this command with the options `--white` ,`--pial`, and `--ventricles` to create the following surfaces: `lhwhite.stl`, `rhwhite.stl`, `lhpial.stl`, `rhpial.stl`, and `ventricles.stl`.

The surface `.stl` files can be loaded into SVM-Tk with the following:

```
import SVMTK as svm
<name> =svm.Surface("<name>.stl")
```

2.4.1 Surface Repairs in SVM-Tk

There are several frequently occurring issues with surfaces generated by FreeSurfer:

1. Multiple surfaces are unconnected.
2. Surfaces may have missing triangles.
3. Triangle aspect ratios vary.

To resolve the first issue, we assume that the surface with the largest area is the primary surface and should be kept. We use `keep_largest_connected_component` to remove any other surfaces. Then, to resolve the second issue, we ensure that there are no missing facets in the surface by calling the function `fill_holes`. Lastly, we resolve the third issue by utilizing the function `isotropic_remeshing`, which will re-triangulate the surface with a provided edge length. Smoothing the surface with `smooth_taubin`, a volume-preserving smoother introduced by Mardal et al. (2022, Chapter 3), is also recommended. These functions will ensure that we have a closed

surface. However, there can be self-intersections in the surface, which can be fixed by using the function `repair_self_intersection`. This function is rather complex, so we refer the reader to the SVM-Tk documentation and examples for details.

2.4.2 Surface Tuning in SVM-Tk

As mentioned, a constructed volume mesh can have hazardous connections due to short distances between non-adjacent/non-connected vertices, which commonly occurs in the folds on the brain surface. This can be fixed by either increasing the mesh resolution or preprocessing the surface at a lower resolution. The first option will naturally increase the computational cost. Thus, in this section we will focus on the second option, which consists of tuning the surfaces to the mesh resolution (see Fig. 2.6).

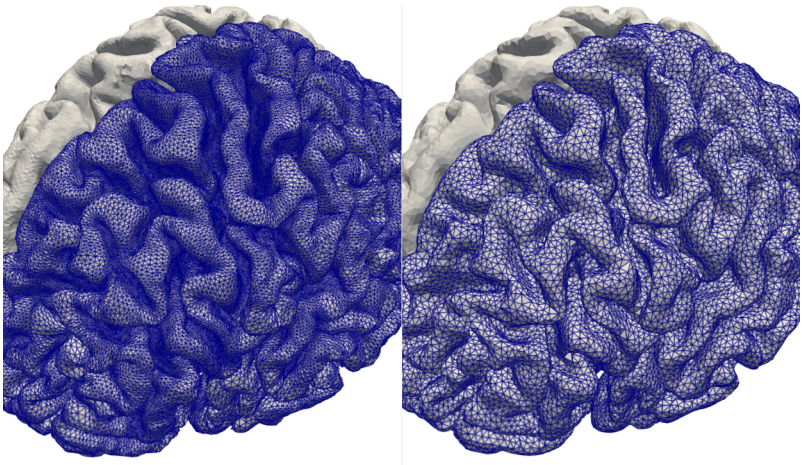


Fig. 2.6: The left panel shows the pial surface processed at a higher resolution. The right panel shows the pial surface processed at a lower resolution. The gaps between the folds are bigger in the right panel, since the processing step will cause larger separations at lower resolution. Processing the surface at the same resolution as the volume mesh will decrease the likelihood of bad edges and vertices.

As a preprocessing step, we would call `isotropic_remeshing` with a larger edge length than that used in Section 2.4.1. Next, we separate the distances between folds by utilizing the function `separate_narrow_gaps`, as shown by Mardal et al. (2022, Chapter 3). The function works by incrementally separating non-adjacent vertices in the negative direction of the surface normal until the closest vertex is connected with an edge.

The functions in this and the previous section have been combined into a single Python function (Valnes, 2024).

2.4.3 Union of Surfaces (White Matter, Brainstem, and Cerebellum)

To obtain a surface that encapsulates the ventricle system, we merge the cerebellum, brainstem, and white matter surface. The merging process works by considering the union sequentially (see Fig. 2.7). Since the cerebellum and white surface are disjointed, we start by taking the union of the cerebellum and brainstem, which is then followed by the white matter. We note that the union command modifies the surface object and should be repaired with functions from Section 2.4.1. We provide the following utility function `sequential_union` for this sequential union in the associated code (Valnes, 2024):

```
brainstem.union(cerebellum)
white.union(brainstem)
```

There are occasions when the ventricle surface is not encapsulated by the combined surfaces. Therefore, in Section 2.4.4 we will describe the method to fix this issue.

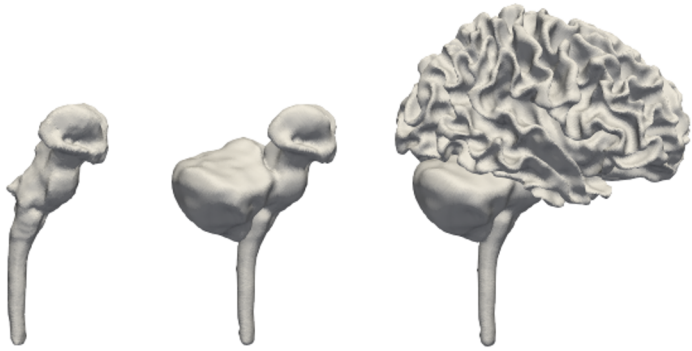


Fig. 2.7: This image shows the process of merging surfaces into a single surface.

2.4.4 Anatomical Correctness

We previously created and repaired several cojoining surfaces, a method that, in principle, can be used to generate meshes. However, the surfaces were created independently, with no knowledge of the other surfaces. Hence, there may be overlapping or crossing surfaces that are clearly anatomically incorrect.

Examples of such flaws are as follows:

- The pial surfaces overlap/cross with the bounding surface of the dura.
- The cerebellum overlaps/crosses with the pial surface.
- The cerebellum overlaps/crosses with the bounding surface of the dura.
- The surface of the ventricles is connected to the SAS through passages other than the cisterna magna.

These flaws will be fixed by employing the SVM-Tk functions `enclose`, `embed`, and `expose` in different cases. As shown in Figure 2.8, `enclose` will incrementally expand until it encloses another surface, `embed` will incrementally contract until it is embedded in another surface, and `expose` will incrementally contract until the surfaces are disjointed. These functions will also separate the surfaces by continuing the incremental process until there is at least an edge length between the surfaces.

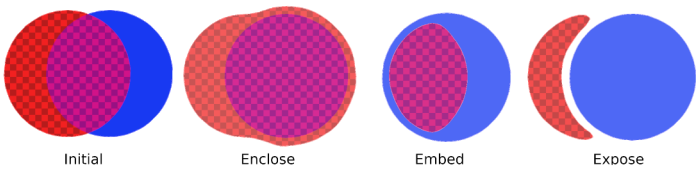


Fig. 2.8: The leftmost panel shows the initial state of two circles, and the other panels show the results after the functions `enclose`, `embed`, and `expose` are applied on the red checkerboard circle. These figures are merely for illustrative purposes to explain the functions.

We suggest the following approach: The first flaw is often caused by the smoothing of the bounding surface, resulting in the pial surfaces protruding. The problem is fixed below by using `enclose` to expand the dura surface locally:

```
svm.enclose(dura,lhpial)
svm.enclose(dura,rhpial)
```

Similarly, to address the second and third flaws, we first separate the cerebellum and the pial surfaces with `expose`:

```
svm.expose(cerebellum,lhpial)
svm.expose(cerebellum,rhpial)
```

Then, we move the cerebellum surface from the dura surface to ensure a CSF pathway around the cerebellum, with `embed`:

```
svm.embed(cerebellum,dura)
```

Finally, we may find the ventricle system connected to the SAS in more locations than through the cisterna magna. In some cases, the ventricle surface may extend far outside the white surface, for example, near the temporal horn of lateral ventricles (see Fig. 2.9). We start by taking the union of the ventricle system and the white surface and use both `enclose` and `embed`:

```
white.union(ventricles)
svm.enclose(white, ventricles)
svm.embed(ventricles, white)
```

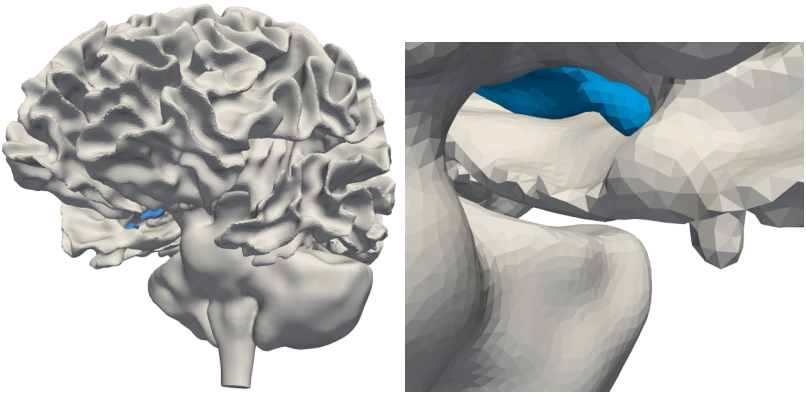


Fig. 2.9: This figure shows the surface of the lateral ventricle in blue, which extends outside the combined surface of the cerebellum, brainstem, and white matter.

2.4.5 Creating the Cisterna Magna Fluid Pathway

To avoid any unintentional pathways between the SAS and the ventricular system, we imposed separations between the surfaces using the functions `enclose` and `embed`. The previous surface operations ensure that there are no unintentional CSF pathways between the SAS and the ventricular system. However, the cisterna magna may have been obstructed, so we need to construct the cisterna magna pathway anew. We do this by using the utility function `make_cisterna_magna`, which will create a connecting cylinder between the ventricular and white surfaces. Then, we subtract the cisterna magna from the white surface, creating a fluid pathway from the SAS to the ventricles.

```
cisterna_magna = make_cisterna_magna(ventricles, white, radius)
white.difference(cisterna_magna)
```

2.5 Creating a Plane Surface for Boundary Conditions

The Monro–Kellie doctrine (Mokri, 2001) states that the volume inside the skull remains constant, implying that the volume changes caused by the pulsating blood flow in arteries and veins is balanced by CSF flow (Baldent, 2014). Lately, the effect of respiration on venous–CSF coupling has also been proposed as a main driver of CSF (Vinje et al., 2019). In any case, the flow through the foramen magnum is crucial and will typically be included as a boundary condition.

Creating a boundary that is perpendicular to the flow is advantageous, so that we can avoid boundary artifacts. Therefore, we want to clip both the brainstem and dura surface with a plane perpendicular to the centerline. To avoid errors due to differences between centerlines, we define the perpendicular plane using only the brainstem, since it is a smaller surface. Then, we use the plane to clip both surfaces. This is done by utilizing `get_foramen_magnum`, which requires that we define a z-plane. We define the z-plane to be 10 units lower than the cerebellum surface in the z-direction.

```
z0 = cerebellum.span(2)[0] - 10
foramen_magnum = get_foramen_magnum(brainstem, z0)
```

We can store the clip surface as an `.stl`-file, to use later. We cut the mesh with the plane using the function `clip`, as shown in the following code snippet:

```
dura.clip(foramen_magnum)
white.clip(foramen_magnum)
```

2.5.1 Constructing Mesh with Sharp Boundaries

We construct a `Domain` object for constructing the volume mesh:

```
surfaces = [dura,lhpial,rhpial,white,ventricles]
domain   = svm.Domain(surfaces, smap)
```

The variable `smap` is a `SubdomainMap` object, which is used to set the subdomain tag during mesh construction. For a detailed description, see Mardal et al. (2022, Chapter 3) and the SVM-Tk documentation.

The default mesh construction in SVM-Tk does not preserve sharp boundaries, since such boundaries are not present in our brain. However, at artificial boundaries such as the foramen magnum described above, preserving sharpness is an advantage. Therefore, we will show how to keep sharp boundaries in the mesh, with different degrees of complexity. This will be done by preserving sharp edges, which are edges where the angle between adjacent facet normals exceeds a threshold angle, set around 60–90 degrees.

In SVM-Tk, there are two primary methods to add edges to the mesh construction, namely, the `Domain` class functions `add_sharp_border_edges` and `add_border`. The function `add_border` is straightforward, since it works by adding a list of points, representing connected edges, to be included in the mesh. It will also check if the new borders overlap previously added borders, since that could cause the construction to crash.

The function `add_sharp_border_edges` uses a surface and threshold angle as primary inputs and works by finding edges in the surface that exceed the threshold. Then, the sharp edges are added with the function `add_border`. As the geometries become more complex, we can lose track of possible sharp boundaries. Thus, we also have the option to only include sharp edges near a given plane or surface by including either as an argument in `add_sharp_border_edges`. This makes it simpler to preserve edges that were created with the `clip` function. We use a threshold angle of 60 degrees and 84 for `mesh_resolution`.

```
domain.add_sharp_border_edges(white, foramen_magnum, 60)
domain.add_sharp_border_edges(dura, foramen_magnum, 60)
domain.create_mesh(84)
```

The variable `mesh_resolution` is used to divide the minimum bounding radius of the surfaces to obtain the upper bound of the cell size in the mesh construction.

2.5.2 Checking the Quality of the Mesh

We briefly discussed the topic of mesh quality, but how do we measure it? There are multiple methods to evaluate the mesh quality. For instance, ParaViewhas a wide range of checks that can be used for more information. Here, we will consider the dihedral angle for each of the tetrahedrons in the mesh as a mesh quality measure. The dihedral angle is the angle between two triangle planes in the tetrahedron. As a rough starting point, we would like the dihedral angle to be between 10 and 70 degrees. Elements with sharp angles are typically called slivers and are known to cause instabilities in simulations.

Slivers are created when they are the only possible configuration that can satisfy the meshing criteria. Thus, the more constraints we add, that is, surfaces and sharp edges, the more likely it is for slivers to occur. This can be mitigated by using mesh optimization functions such as `exude`, `lloyd`, `perturb`, and `odt`, and SVM-Tk provides a Python wrapper for the corresponding CGAL implementations (Cheng et al., 2000; Du et al., 1999; Du and Wang, 2003; Tournois et al., 2009).

In Figure 2.10(left), we plot the initial distribution of the dihedral angles and, afterward, both `odt` and `exude` optimization. We can see that the optimization functions produce a significant change, but the optimization may also remove troublesome tetrahedrons, causing missing tetrahedrons in the volume mesh, as also shown in

Figure 2.10(right). The function `dihedral_angles` provides the lowest dihedral angle for each cell in the volume mesh.

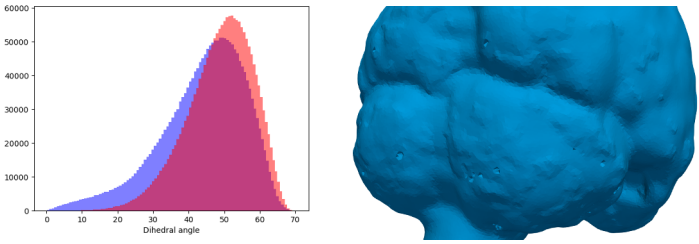


Fig. 2.10: The left panel shows the initial dihedral angle distribution in blue and the distribution after `odt` optimization and `exude` optimization in red. The right panel shows tetrahedrons missing due to the optimization, that is, their removal rather than their restructuring.

2.5.3 Boundary Segmentation

We want to mark the foramen magnum with a different boundary tag, so that boundary conditions become simpler to set. This can be done with the function `boundary_segmentation`, which will segment a given surface of the constructed mesh based on sharp edges. The surface can be specified with a subdomain tag indicative of the surface enclosing the subdomain or by a tuple representing an interface between two subdomains (with zero indicating a non-mesh boundary). This method should be applied before exporting the mesh, since the optimization might remove the interface tag.

```
domain.boundary_segmentations((1, 0), 60)
```

The resulting segmentation is visible in Figure 2.11.

2.6 Concluding Remarks

In this chapter, we have extended the work of Mardal et al. (2022) to also incorporate the cerebellum, brainstem, and cerebrospinal fluid into a patient-specific mesh. This method includes adding markers for the different subdomain and interfaces, making the mesh suitable for multiphysics simulations. We also suggested methods and tools to handle problems that often occur when combining multiple surfaces from MRI, since these often overlap, creating nonphysical features. Finally, the resulting mesh

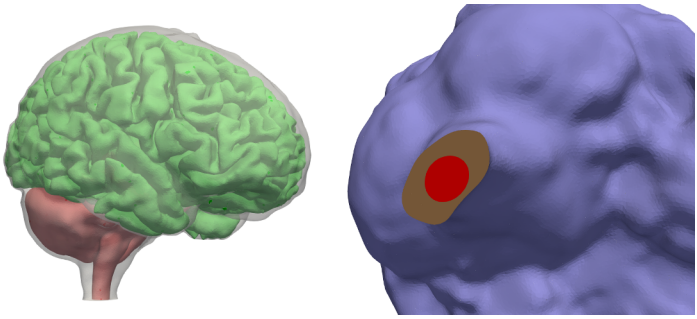


Fig. 2.11: The left panel shows the subdomain markings, with the SAS set on a lower opacity. The right panel shows the planar boundary has a boundary tag different from that of the SAS boundary.

can be converted to different formats and loaded into finite element software such as FEniCS.

References

Alnæs MS, et al. (2015) The fenics project version 1.5. Archive of Numerical Software 3, doi:10.11588/ans.2015.100.20553

Baldent O (2014) Imaging of the cerebrospinal fluid circulation, Cambridge University Press, pp 121–138. doi:10.1017/CBO9781139382816.013, editor: Rigamonti, Daniele

Bojarskaite L, et al. (2022) Sleep cycle-dependent vascular dynamics enhance perivascular cerebrospinal fluid flow and solute transport. bioRxiv doi:10.1101/2022.07.14.500017

Cheng SW, Dey TK, Edelsbrunner H, Facello MA, Teng SH (2000) Sliver exudation. Journal of the ACM (JACM) 47(5):883–904, doi:10.1145/355483.355487

Cinalli G, Spennato P, Nastro A, Aliberti F, Trischitta V, Ruggiero C, Mirone G, Cianciulli E (2011) Hydrocephalus in aqueductal stenosis. Child's Nervous System 27(10):1621–1642, doi:10.1007/s00381-011-1546-2

Corti M, Antonietti PF, Dede L, Quarteroni AM (2023) Numerical modeling of the brain poromechanics by high-order discontinuous Galerkin methods. Mathematical Models and Methods in Applied Sciences 33(08):1577–1609, doi:10.1142/S0218202523500367

Du Q, Wang D (2003) Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations. International journal for numerical methods in engineering 56(9):1355–1373, doi:10.1002/nme.616

Du Q, Faber V, Gunzburger M (1999) Centroidal Voronoi tessellations: Applications and algorithms. SIAM review 41(4):637–676, doi:10.1137/S0036144599352836

Eide PK, Valnes LM, Lindstrøm EK, Mardal KA, Ringstad G (2021) Direction and magnitude of cerebrospinal fluid flow vary substantially across central nervous system diseases. Fluids and Barriers of the CNS 18(1):16, doi:10.1186/s12987-021-00251-6

Fabri A, Giezeman GJ, Kettner L, Schirra S, Schönherr S (2000) On the design of CGAL a computational geometry algorithms library. Software: Practice and Experience 30(11):1167–1202, doi:10.1002/1097-024X(200009)30:11<1167::AID-SPE337>3.0.CO;2-B

Fischl B (2012) FreeSurfer. NeuroImage 62(2):774–781, doi:10.1016/j.neuroimage.2012.01.021

Fultz NE, Bonmassar G, Setsompop K, Stickgold RA, Rosen BR, Polimeni JR, Lewis LD (2019) Coupled electrophysiological, hemodynamic, and cerebrospinal fluid oscillations in human sleep. Science 366(6465):628–631, doi:10.1126/science.aax5440

Gueziec A, Taubin G, Lazarus F, Horn W (1998) Converting sets of polygons to manifold surfaces by cutting and stitching. In: Proceedings Visualization '98 (Cat. No.98CB36276), pp 383–390, doi:10.1109/VISUAL.1998.745327

Hornkjøl M, Valnes LM, Ringstad G, Rognes ME, Eide PK, Mardal KA, Vinje V (2022) CSF circulation and dispersion yield rapid clearance from intracranial compartments. Frontiers in Bioengineering and Biotechnology 10:932469, doi:10.3389/fbioe.2022.932469

Jessen NA, Munk ASF, Lundgaard I, Nedergaard M (2015) The Glymphatic System: A Beginner's Guide. Neurochem Res 40(12):2583–2599, doi:10.1007/s11064-015-1581-6

Jzsa T, Petr J, Payne S, Mutsaerts H (2023) MRI-based parameter inference for cerebral perfusion modelling in health and ischaemic stroke. Computers in Biology and Medicine 166:107543, doi:10.1016/j.compbiomed.2023.107543

Lindstrm EK, Ringstad G, Mardal KA, Eide PK (2018) Cerebrospinal fluid volumetric net flow rate and direction in idiopathic normal pressure hydrocephalus. NeuroImage: Clinical 20:731–741, doi:0.1016/j.nicl.2018.09.006

Mardal KA, Rognes M, Thompson T, Valnes L (2022) Mathematical Modeling of the Human Brain: From Magnetic Resonance Images to Finite Element Simulation. Springer, Cham, doi:10.1007/978-3-030-95136-8

Mestre H, et al. (2018) Flow of cerebrospinal fluid is driven by arterial pulsations and is reduced in hypertension. Nat Commun 9(1):4878, doi:10.1038/s41467-018-07318-3

Mokri B (2001) The monro–kellie hypothesis. Neurology 56(12):1746–1748, doi:10.1212/WNL.56.12.1746

Møllgård K, Beinlich FR, Kusk P, Miyakoshi LM, Delle C, Plá V, Hauglund NL, Esmail T, Rasmussen MK, Gomolka RS, et al. (2023) A mesothelium divides the subarachnoid space into functional compartments. Science 379(6627):84–88, doi:10.1126/science.adc8810

Piersanti E, Rognes ME, Vinje V (2023) Are brain displacements and pressures within the parenchyma induced by surface pressure differences? A computational modelling study. Plos one 18(12):e0288668, doi:10.1371/journal.pone.0288668

Ringstad G, Lindstrøm EK, Vatnehol SAS, Mardal KA, Emblem KE, Eide PK (2017) Non-invasive assessment of pulsatile intracranial pressure with phase-contrast magnetic resonance imaging. PloS one 12(11):e0188896, doi:10.1371/journal.pone.0188896

Tournois J, Srinivasan R, Alliez P (2009) Perturbing slivers in 3D Delaunay meshes. In: Proceedings of the 18th international meshing roundtable, Springer, pp 157–173, doi:10.1007/978-3-642-04319-2

Valnes LM (2024) mri2fem-chp2-code: v1.0.0. doi:10.5281/zenodo.10818263

Valnes LM, Mardal KA (2024) mri2femii-chp2-dataset: v1.0.0. doi:10.5281/zenodo.14536218

Vinje V, Ringstad G, Lindstrøm EK, Valnes LM, Rognes ME, Eide PK, Mardal KA (2019) Respiratory influence on cerebrospinal fluid flow – a computational study based on long-term intracranial pressure measurements. Scientific Reports 9(1):9732, doi:10.1038/s41598-019-46055-5

Vinje V, Zapf B, Ringstad G, Eide PK, Rognes ME, Mardal KA (2023) Human brain solute transport quantified by glymphatic MRI-informed biophysics during sleep and sleep deprivation. Fluids and Barriers of the CNS 20(1):62, doi:10.1186/s12987-023-00459-8

Chapter 3

Segmenting, meshing, and modeling CSF spaces

Martin Hornkjøl, Lars Magnus Valnes and Jørgen S. Dokken

Abstract The goal of this chapter is to demonstrate how one can go from a surface triangulation of the brain to modeling cerebrospinal fluid (CSF) flow. Specifically, the model will include the subarachnoid space, a ventricular system, and the CSF produced by the choroid plexus in the ventricles. The CSF is modeled to adhere to the Stokes equations, which in turn are solved with FEniCS, a finite element framework using an iterative solver. The computed velocity field corresponds well with current experimental and computational studies.

3.1 Introduction

In medical studies and patient care, tracers, such as gadolinium-based contrast agents, are used as a surrogate to monitor fluid flow and infer solute transport in the body. For the specific study of solute transport through the brain, tracers are introduced to the subject through lumbar intrathecal injection. The tracers are transported with the cerebrospinal fluid (CSF) from the spinal subarachnoid space (SAS) to the brain (Ringstad et al., 2017). The transport of the tracers is monitored with T1-weighted magnetic resonance imaging at various points in time. From these images, it has been shown that the CSF flow varies for different pathologies (Eide et al., 2021). Thus, the need for patient-specific modeling is evident, since detailed studies of large cohorts over a long time span are not feasible.

Several computational models have been developed to study the transport of tracers in the brain (Valnes et al., 2020; Zapf et al., 2022). These models have considered the extracellular diffusive transport of tracer in brain tissue. However, the importance of fluid flow in tracer transport has been acknowledged by several studies (Ray et al., 2021; Vinje et al., 2023), where medical images were compared with computational diffusive models. They concluded that extracellular diffusion is not sufficient to explain the transport of tracers in the brain.

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_3

Few studies have considered both the CSF flow and the transport of tracers in the SAS. Vinje et al. (2019) considered the CSF flow in the cerebral aqueduct and the effect of respiration on the CSF flow. One of few such studies, Hornkjøl et al. (2022) considered a multiphysics approach that models both the CSF and brain tissue. These models are more computationally expensive, since they require solving multiple partial differential equations, either in a segregated, iterative approach or as a monolithic system. However, these models might be required to fully understand the underlying physical processes of tracer transport.

In this chapter, we consider a simplified version of the study of Hornkjøl et al. (2022), where we model fluid flow in the SAS and ventricles using patient-specific surfaces obtained through the methods described by Valnes and Mardal (2024). Special care is taken to ensure that the ventricles and SAS are properly resolved in the surfaces that are passed into the Surface-Volume-Meshing Toolkit (Mardal et al., 2022). We will model the fluid flow using Stokes equations, discretized with a finite element method and implemented in FEniCS (Alnæs et al., 2015; Baratta et al., 2023).

3.2 Model of Fluid Flow in CSF Spaces

The CSF is assumed to adhere to the Stokes equations, since the Reynolds number is very low and the flow is laminar (Daversin-Catty et al., 2020; Mestre et al., 2018). The Stokes equations can be written as

$$\mu\nabla^2 u - \nabla p = 0 \text{ in } \Omega_F, \tag{3.1a}$$
$$\nabla \cdot u = g \text{ in } \Omega_F, \tag{3.1b}$$
$$u = 0 \text{ in } \Omega_P, \tag{3.1c}$$
$$u = 0 \text{ on } \Gamma_{FP}, \tag{3.1d}$$
$$\mu\nabla u \cdot n - pn = 0 \text{ on } \partial\Omega_{ps}, \tag{3.1e}$$
$$u = 0 \text{ on } \partial\Omega \setminus \partial\Omega_{ps}, \tag{3.1f}$$

where Ω_F is the CSF region, Ω_P is the brain tissue, and $\mu = 0.8$ mPa is the viscosity of water. The function g in Equation (3.1b) is spatially dependent and represents a source/producer (Weed, 1914) of CSF in the choroid plexus (Ω_{cp}):

$$g = \begin{cases} 0 \text{ in } \Omega_F \setminus \Omega_{cp}, \\ g \text{ in } \Omega_{cp}. \end{cases} \tag{3.2}$$

The choroid plexus is a network of capillaries and cells located within the ventricles (Damkier et al., 2013). We will only consider the lateral ventricles in this chapter. For a more thorough introduction to the ventricular system, see Rognes et al. (2024, Chapter 1).

The velocities measured in the SAS are substantial, with pulsatile velocities of 10–40 μm/s in mice (Bedussi et al., 2018; Mestre et al., 2018). In humans, pulsatile velocities on the order 5cm/s have been measured in the foramen magnum (Bradley et al., 2016). A small pressure gradient of 1–2 mmHg/m has been measured in humans, and computational studies using this value indicate that the bulk velocity is on the order of 1 μm/s in the ventricular system and surface paravascular space (Vinje et al., 2019). These CSF space velocities are in stark contrast to the velocities in the brain parenchyma, which are usually in the range of 0.2 μm/min (Nicholson, 2001; Rosenberg et al., 1980), This motivates the choice of setting the velocity in the parenchyma to zero, as seen in Equations (3.1c) and (3.1d).

Figure 3.1 highlights the regions of importance for the simulations in this chapter. The choroid plexus is in yellow.

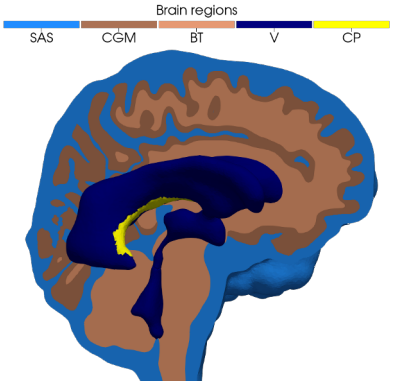


Fig. 3.1: The different regions of the brain, including the SAS, cortical gray matter (CGM), brain tissue (BT), ventricles (V), and choroid plexus (CP). We consider brain tissue and cortical gray matter solids. Fluid is thus only allowed in the SAS, ventricles, and choroid plexus.

The parasagittal sinus is chosen as an outflow pathway, Equation (3.1e), as do Hornkjøl et al. (2022). One can consider other possible fluid outflow pathways, such as the cribriform plate (Hladky and Barrand, 2014). The surface of the parasagittal sinus is visualized in Figure 3.2 where free flow is permitted. The rest of the skull has a no-slip condition imposed by Equation (3.1f).

3.3 Mesh Creation for Fluid Flow Simulations

The aim of this section is to create a mesh consisting of both the brain tissue and the CSF spaces. Most of the methods used in this chapter are described in detail

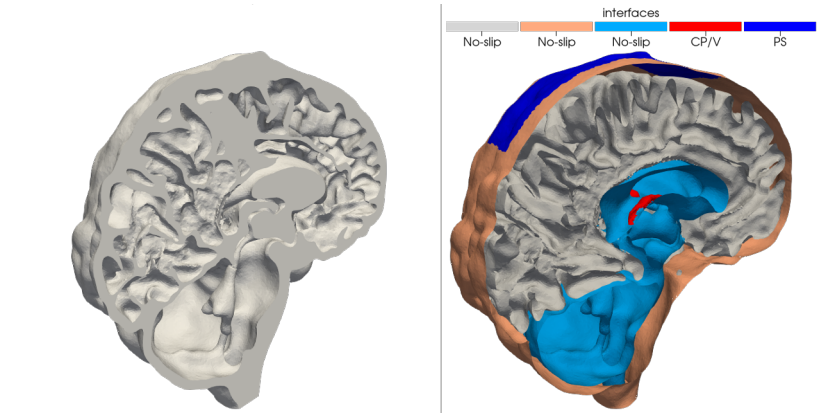


Fig. 3.2: The left figure visualizes the fluid spaces surrounding the brain. The right figure shows the various interfaces considered in the simulations. Here, the outlet is chosen to be the parasagittal sinus, and CP/V indicates the internal interface in the ventricles to the choroid plexus, which produces the CSF.

by Valnes and Mardal (2024, Chapter 2). However, we cover additional steps required to obtain a mesh suitable for fluid flow simulations.

3.3.1 The Choroid Plexus

Since the choroid plexus is the producer of CSF, it must be properly resolved in the mesh. It can either be represented as a boundary condition, where one excludes the region where the choroid plexus is located, or as a source term in the fluid region. However, setting appropriate boundary conditions for the choroid plexus is challenging. Therefore, we model it as part of the CSF and assume that the production of CSF in the choroid plexus is 0.5 L/day (Pardridge, 2016).

The choroid plexus is a thin structure and often not well resolved in magnetic resonance images. This results in FreeSurfer not always creating contiguous segmentation, with the choroid plexus potentially located within the brain tissue rather than in the ventricles. If this occurs, the meshing of the fluid volume will lead to degenerate meshes consisting of small/squashed tetrahedra. These tetrahedra, in turn, lead to unphysical simulation results. To resolve these issues, one can manually segment the choroid plexus in Freeview. However, this is not a trivial task and should be done in collaboration with professionals. For the model in this chapter, we use the `get_surface --choroid-plexus` script described in Valnes and Mardal (2024, Chapter 2.3). See Figure 3.1 for the resulting choroid plexus surface.

3.3.2 Making a Full Brain Mesh by Combining Surfaces

Our goal is to create a mesh of the brain with a bounding SAS. We use the surfaces created in Valnes and Mardal (2024, Chapter 2) as a starting point and make minor adjustments to suit our needs. The full script, as well as all necessary *stl* files, `create_mesh.py`, is found online (Hornkjøl, 2024).

First, we load the following surfaces into SVM-Tk: the white matter `white.stl`, the left hemisphere pial surface `lh.pial.stl`, the hemisphere pial surface `rh.pial.stl`, the dura mater meninges (boundary to the SAS) `dura.stl`, the ventricles `ventricles.stl`, the left choroid plexus `lh.choroid.stl`, and the right choroid plexus `rh.choroid.stl`.

In certain cases, the SAS is so thin that it is hard to resolve in a mesh. To ensure that we include some SAS, we expand the SAS surface by a few millimeters. This can be done with the command `adjust_boundary` in SVM-Tk:

```
bounding_surface.adjust_boundary(2)
```

The value of two is represented in the units of the input mesh, which in our case are millimeters. We additionally call the `repair_surface` method from Valnes and Mardal (2024, Chapter 2.4.1) to fix possible defects in the surfaces. Alternatively, one can set a lower value in `adjust_boundary` to expand the surface less.

The aqueduct is usually the thinnest part of the mesh, connecting the third and fourth ventricles. Therefore, we want to add the aqueduct as a separate subdomain for local refinement in Section 3.3.3, and for this we create the aqueduct surface. This requires knowledge about the coordinates of the aqueduct. We start by opening the ventricle surface file in ParaView and select Hover Points On, which shows the coordinates of whichever point over which we are hovering. We select two points on the surface, indicating the top and bottom of the aqueduct, which we respectively name `p1` and `p2`. Then we create a copy of the ventricles and utilize the function `clip_aqueduct`. The function finds perpendicular cuts near `p1` and `p2`, which are subsequently used to clip the ventricular surface to create the aqueduct surface. The result is shown in Figure 3.3, and the code to conduct this process is the following:

```
aqueduct = svm.Surface(ventricles)
p1 = svm.Point_3(-7.0,-23.0, 4.0)
p2 = svm.Point_3(-6.9,-30.2,-5.6)
aqueduct = clip_aqueduct(venticles,p1,p2)
```

With the necessary surfaces constructed, we can divide the mesh into six subdomains, namely, the SAS, the cortical gray matter, the remaining brain tissue, the ventricles, the choroid plexus, and the aqueduct. The code with descriptions is found in Hornkjøl (2024), and a detailed explanation can be found in Mardal et al. (2022, Chapter 3 and 4.1).

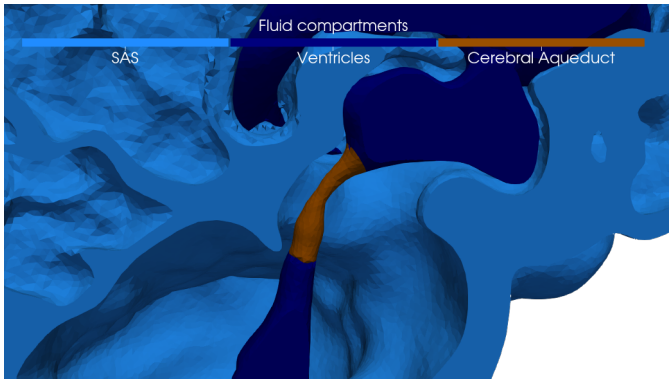


Fig. 3.3: The cerebral aqueduct marked as a separate subdomain.

3.3.3 Mesh Resolution and Refinement

When using a mesh for an application, it is important to consider its resolution. This is especially important in thin regions, such as the cerebral aqueduct and the SAS. If we consider 2D slices through such a thin domain, and observe that all vertices are on the fluid-tissue or fluid-skull boundary, the flow will effectively be stopped. To avoid this, one can refine the mesh in these areas. There are multiple ways of making a finer mesh. Either, one can let SVM-Tk handle the mesh resolution when generating the mesh, or one can refine the mesh after it has been created. As long as the surfaces are sufficiently resolved, it is often beneficial to use external tools (other than SVM-Tk) to refine the mesh, as:

1. SVM-Tk does not use distributed computing to generate the mesh.
2. SVM-Tk does not support adaptive mesh refinement.

For adaptive and parallel mesh refinement, one can use the Plaza and Carey (2000) algorithm, which has been implemented in both FEniCS Alnæs et al. (2015) and DOLFINx Baratta et al. (2023). A script for FEniCS can be found in (Mardal et al., 2022, Chapter 4.5), while a script for a similar refinement routine in DOLFINx can be found in the file `refine_mesh.py` at Hornkjøl (2024).

3.4 Implementation of the Finite Element Formulation

In this section, we cover how to implement the equations in Section 3.2 in DOLFINx Baratta et al. (2023). First, we will show how to extract the CSF spaces as a separate grid, which we will use for fluid flow simulation.

3.4.1 Loading Meshes and Marking Modification

SVM-Tk can generate `.mesh` files, which can be converted into `.xdmf` files using `meshio` Schlömer (2024). The details on how to use meshio for this conversion, see Hornkjøl (2024). Another approach is to directly interface DOLFINx and SVM-Tk by reading in the data directly from the `svmtk.Domain` object. We utilize the function (unpack_mesh) to extract the necessary components.

```
points, cells, markers, facets, f_markers = unpack_mesh(domain)
```

We can then construct the mesh in DOLFINx after storing it as `.npz`.

```
filename = "brain_with_CSF.npz"
numpy.savez(filename, points=points, cells=cells,
      cell_tags=markers, facets=facets, facet_tags=f_markers)
```

The following code distributes the over the number of processes the Python script is initiated with

```
domain, ct, ft = read_from_svmtk_npz(MPI.COMM_WORLD, filename)
```

Here `ct` is a `dolfinx.mesh.MeshTags`-object containing the subdomain markers from SVM-Tk, while `ft` is a `dolfinx.mesh.MeshTags`-object containing the surface markers.

In Section 3.2 we defined that there should be an outflow through the parasagittal sinus. However, this was not annotated in the `facet_tags` that we loaded from SVM-Tk. We use `extend_facet_marker_with_outlet` from Hornkjøl (2024) to mark the parasagittal sinus as an outflow boundary.

```
ft = extend_facet_marker_with_outlet(
  domain, ft, x_bounds=(-28, 4), y_bounds=(-100, 11), z_bound=40)
```

3.4.2 Extraction of Subdomains

In this chapter, we do not want to model mechanics within the brain tissue. Therefore, we extract the the fluid volume using `dolfinx.mesh.create_submesh`.

```
fluid_markers = (1, 4, 5, 6) # Markers for the fluid regions
fluid_cells = ct.indices[np.isin(ct.values, fluid_markers)]
mesh, c_map, v_map, _ = dolfinx.mesh.create_submesh(
  domain, ct.dim, fluid_cells)
cell_tags = transfer_meshtags_to_submesh(
  domain, ct, mesh, v_map, c_map)
facet_tags = transfer_meshtags_to_submesh(
  domain, ft, mesh, v_map, c_map)
```

The variable `mesh` will be used as the computational domain in the subsequent sections. Additionally, we transfer the markers from the full mesh to the submesh using `transfer_meshtags_to_submesh` in Hornkjøl (2024), as the markers are needed for the boundary conditions and for specifying g in (3.2).

3.4.3 Solving the Stokes Problem

Once the mesh has been read in to DOLFINx, we start by creating appropriate inf-sup stable finite element spaces for the velocity and pressure.

```
cell = mesh.basix_cell()
P2 = element("Lagrange", cell, 2, shape=(3, ))
P1 = element("Lagrange", cell, 1)
taylor_hood = mixed_element([P2, P1])
W = dolfinx.fem.functionspace(mesh, taylor_hood)
```

We define an integration measure over all cells in the SAS as `dx`, which can be restricted to any of the markers we have read in from SVM-Tk.

```
dx = ufl.Measure("dx", domain=mesh, subdomain_data=cell_tags)
```

The shape argument in the constructor of `P2` indicates how many components the velocity field should have. Next, we compute the production value of the CSF that we want to apply in the choroid plexus. As stated previously, we use the estimates by Pardridge (2016) and convert it into the appropriate unit for our simulations.

```
cp_marker = 5 # Integer id for choroid plexus
comm = mesh.comm
choroid_plexus_volume = dolfinx.fem.form(1*dx(cp_marker))
vol = comm.allreduce(dolfinx.fem.assemble_scalar(
  choroid_plexus_volume), op=MPI.SUM)
g_source = dolfinx.fem.Constant(mesh, production_value/vol)
```

The variational form of the Stokes equation (3.1a) and (3.1b) is obtained in the standard way, using multiplication of a test and trial function and integration by parts. The outlet condition(3.1e) is not included in the variational form, as it is a natural boundary condition. We represent the variational form in the unified form language (UFL) Alnæs et al. (2014).

```
mu = dolfinx.fem.Constant(mesh, water_viscosity)
(u, p) = ufl.TrialFunctions(W)
(v, q) = ufl.TestFunctions(W)
a = mu * ufl.inner(ufl.grad(u), ufl.grad(v)) * dx
a -= ufl.div(v) * p * dx
a -= q * ufl.div(u) * dx
L = -g_source * q * dx(cp_marker)
```

We apply the no-slip boundary conditions (3.1d) and (3.1f) as shown below:

```
W0 = W.sub(0)
V, _ = W0.collapse()
no_slip = dolfinx.fem.Function(V)
bcs = []
mesh.topology.create_connectivity(2, 3)
for marker in noslip_markers:
  facets = facet_tags.find(marker)
  fixed_dofs = dolfinx.fem.locate_dofs_topological(
    (W0, V), 2, facets)
  bcs.append(dolfinx.fem.dirichletbc(no_slip, fixed_dofs, W0))
```

Due to the complexity of the SAS spaces, the condition number of the arising linear system is very high, which in turn leads to degradation of the convergence of an iterative solver. To mitigate this, we assemble a standard preconditioner for Stokes equation Mardal and Winther (2011). The following code-snippet shows how to assemble the preconditioner.

```
P = mu * ufl.inner(ufl.grad(u), ufl.grad(v)) * dx
P += (1.0 / mu) * p * q * dx
p_compiled = dolfinx.fem.form(P)
P = dolfinx.fem.petsc.assemble_matrix(p_compiled, bcs=bcs)
P.assemble()
```

We use the minimal residual (MINRES) Krylov subspace method preconditioned with BoomerAMG Henson and Yang (2002) to efficiently solve the linear system of equations iteratively

```
opts = {"ksp_type": "minres", "pc_type": "hypre",
  "pc_hypre_type": "boomeramg",
  "ksp_error_if_not_converged": True,
  "ksp_atol": 1e-6, "ksp_rtol": 1e-6}
problem = dolfinx.fem.petsc.LinearProblem(
  a, L, bcs=bcs, petsc_options=opts)
problem.solver.setOperators(problem.A, P)
wh = problem.solve()
```

Note that we have attached the preconditioner to the linear problem in the second to last line of the snippet above.

3.5 Results

The fluid domain consists of 2 453 870 tetrahedra and 546 542 vertices. The velocity function space has 11 518 224 degrees of freedom, while the pressure space has 546 542 degrees of freedom. The linear problem is solved with 1155 iterations of the MINRES method, using the preconditioned residual norm for the stopping criterion. The reason for this high-iteration count is that the inf-sup condition breaks

down in complex, anisotropic geometries Chizhonkov and Olshanskii (2000). In fact, the condition number of the preconditioned matrix is estimated to be 927 000. Further work is needed to improve the convergence of the iterative solver, such as using a more advanced preconditioner, such as balancing domain decomposition by constraints preconditioner Li and Widlund (2006). The streamlines of the velocity field are shown in Figure 3.4. We observe peak magnitudes of $\sim$ 3.3 mm/s in the thinnest part of aqueduct, and $\sim 2-4$ μm/s in the SAS. Figure 3.4 shows a cross

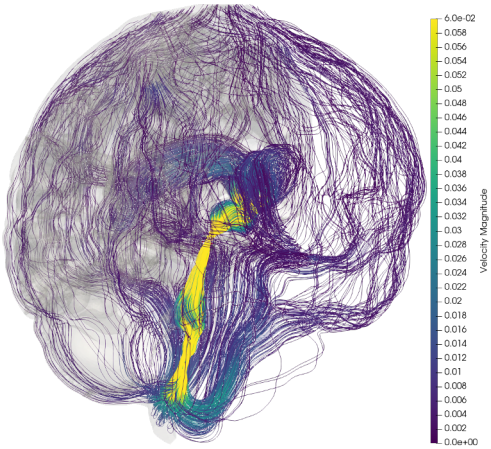


Fig. 3.4: The magnitude of the CSF velocity field on our mesh computed with the Stokes equations. The mesh is cut such that the third and fourth ventricle and the aqueduct can be seen. The plot is scaled such that the cutoff maximum velocity is 0.06 mm/s so that the velocity field outside of the ventricles are visible.

section of the magnitude of the solution. The velocity field has a source in the choroid plexus and moves through the ventriclular systems where it exits into the SAS at the cisterna magna. From there, the velocity field spreads upward through the SAS and out of the parasagittal sinus. The velocities found with this model are similar too those found in experimental studies with mice Bedussi et al. (2018); Mestre et al. (2018) and matches well with previous simulations Vinje et al. (2019). The peak memory usage was 120 GB RAM, with the simulation being partitioned over 20 CPUs.

3.6 Conclusions

In this chapter, we have shown how to go from surface triangulations of the brain to a fluid flow simulation in the CSF spaces. The velocity field found is in agreement with previous work and experimental studies. The work done on here can

be extended to explore the transport of molecules to the brain. This is done in Hornkjøl et al. (2022), which, using the methods described in this chapter, creates a brain mesh and calculates the velocity field caused by CSF production in the choroid plexus. The paper further extends this work by modeling molecular transport with the convection-diffusion equation. With our assumption of zero velocity in the parenchyma, molecular transport is driven purely by diffusion in this region, while in the CSF spaces both diffusion and convective flow are present. Other effects such as dispersion can be added by i.e. modifying the diffusion coefficient. This work showed that the SAS has an important role in clearing solutes from the intracranial compartment. Another possible extension of the work presented here, which is not explored in Hornkjøl et al. (2022), is to challenge the assumption of zero velocity in the parenchyma by treating it as a porous material and solving the Biot-Stokes equation to see how that affects the velocity field and molecular transport.

References

Alnæs MS, Logg A, Ølgaard KB, Rognes ME, Wells GN (2014) Unified form language: A domain-specific language for weak formulations of partial differential equations. ACM Trans Math Softw 40(2), doi:10.1145/2566630

Alnæs MS, et al. (2015) The fenics project version 1.5. Archive of Numerical Software 3, doi:10.11588/ans.2015.100.20553

Baratta IA, et al. (2023) DOLFINx: The next generation FEniCS problem solving environment. doi:10.5281/zenodo.10447666

Bedussi B, Almasian M, de Vos J, Van Bavel E, Bakker EN (2018) Paravascular spaces at the brain surface: Low resistance pathways for cerebrospinal fluid flow. Journal of Cerebral Blood Flow & Metabolism 38(4):719–726, doi:10.1177/0271678X17737984

Bradley WG, Haughton V, Mardal KA (2016) Cerebrospinal fluid flow in adults. In: Handbook of Clinical Neurology, vol 135, Elsevier, pp 591–601, doi:10.1016/B978-0-444-53485-9.00028-3

Chizhonkov EV, Olshanskii MA (2000) On the domain geometry dependence of the LBB condition. ESAIM: Mathematical Modelling and Numerical Analysis 34(5):935–951, doi:10.1051/m2an:2000110

Damkier HH, Brown PD, Praetorius J (2013) Cerebrospinal Fluid Secretion by the Choroid Plexus. Physiological Reviews 93(4):1847–1892, doi:10.1152/physrev.00004.2013, pMID: 24137023

Daversin-Catty C, Vinje V, Mardal KA, Rognes ME (2020) The mechanisms behind perivascular fluid flow. Plos one 15(12):e0244442, doi:10.1371/journal.pone.0244442

Eide PK, Valnes LM, Lindstrøm EK, Mardal KA, Ringstad G (2021) Direction and magnitude of cerebrospinal fluid flow vary substantially across central nervous system diseases. Fluids and Barriers of the CNS 18(1):16, doi:10.1186/s12987-021-00251-6

Henson VE, Yang UM (2002) BoomerAMG: A parallel algebraic multigrid solver and preconditioner. Applied Numerical Mathematics 41(1):155–177, doi:10.1016/S0168-9274(01)00115-5, developments and Trends in Iterative Methods for Large Systems of Equations - in memorium Rudiger Weiss

Hladky SB, Barrand MA (2014) Mechanisms of fluid movement into, through and out of the brain: evaluation of the evidence. Fluids and Barriers of the CNS 11(1), doi:10.1186/2045-8118-11-26

Hornkjøl M (2024) mhornkjol/mri2fem-ii-chapter-3-code: v1.0.0. doi:10.5281/zenodo.10808334

Hornkjøl M, Valnes LM, Ringstad G, Rognes ME, Eide PK, Mardal KA, Vinje V (2022) CSF circulation and dispersion yield rapid clearance from intracranial compartments. Frontiers in Bioengineering and Biotechnology 10:932469, doi:10.3389/fbioe.2022.932469

Li J, Widlund O (2006) BDDC Algorithms for Incompressible Stokes Equations. SIAM Journal on Numerical Analysis 44(6):2432–2455, doi:10.1137/050628556
Mardal KA, Winther R (2011) Preconditioning discretizations of systems of partial differential equations. Numerical Linear Algebra with Applications 18(1):1–40, doi:10.1002/nla.716
Mardal KA, Rognes M, Thompson T, Valnes L (2022) Mathematical Modeling of the Human Brain: From Magnetic Resonance Images to Finite Element Simulation. Springer, Cham, doi:10.1007/978-3-030-95136-8
Mestre H, et al. (2018) Flow of cerebrospinal fluid is driven by arterial pulsations and is reduced in hypertension. Nat Commun 9(1):4878, doi:10.1038/s41467-018-07318-3
Nicholson C (2001) Diffusion and related transport mechanisms in brain tissue. Reports on progress in Physics 64(7):815, doi:10.1088/0034-4885/64/7/202
Pardridge WM (2016) CSF, blood-brain barrier, and brain drug delivery. Expert Opinion on Drug Delivery 13(7):963–975, doi:10.1517/17425247.2016.1171315, pMID: 27020469
Plaza A, Carey G (2000) Local refinement of simplicial grids based on the skeleton. Applied Numerical Mathematics 32(2):195–218, doi:10.1016/S0168-9274(99)00022-7
Ray LA, et al. (2021) Quantitative analysis of macroscopic solute transport in the murine brain. Fluids and Barriers of the CNS 18(1):55, doi:10.1186/s12987-021-00290-z
Ringstad G, Vatnehol SAS, Eide PK (2017) Glymphatic MRI in idiopathic normal pressure hydrocephalus. Brain 140(10):2691–2705, doi:10.1093/brain/awx191
Rognes ME, Mardal KA, Valnes LM, Vinje V (2024) From brain physiology to brain physics. In: Dokken JS, et al. (eds) Mathematical modeling of the human brain (vol II): from glymphatics to deep learning, Springer
Rosenberg G, Kyner WT, Estrada E (1980) Bulk flow of brain interstitial fluid under normal and hyperosmolar conditions. AJP-Renal Physiology 238(1), doi:10.1152/ajprenal.1980.238.1.F42
Schlömer N (2024) meshio: Tools for mesh files. doi:10.5281/zenodo.1288334
Valnes LM, Mardal KA (2024) Meshing the intracranial compartments: Cerebellum, cerebrum, brainstem and cerebrospinal fluid. In: Dokken JS, et al. (eds) Mathematical modeling of the human brain (vol II): from glymphatics to deep learning, Springer
Valnes LM, Mitusch SK, Ringstad G, Eide PK, Funke SW, Mardal KA (2020) Apparent diffusion coefficient estimates based on 24 hours tracer movement support glymphatic transport in human cerebral cortex. Scientific Reports 10(1):1–12, doi:10.1038/s41598-020-66042-5
Vinje V, Ringstad G, Lindstrøm EK, Valnes LM, Rognes ME, Eide PK, Mardal KA (2019) Respiratory influence on cerebrospinal fluid flow – a computational study based on long-term intracranial pressure measurements. Scientific Reports 9(1):9732, doi:10.1038/s41598-019-46055-5
Vinje V, Zapf B, Ringstad G, Eide PK, Rognes ME, Mardal KA (2023) Human brain solute transport quantified by glymphatic MRI-informed biophysics during sleep and sleep deprivation. Fluids and Barriers of the CNS 20(1):62, doi:10.1186/s12987-023-00459-8
Weed LH (1914) Studies on cerebro-spinal fluid. No. IV: the dual source of cerebro-spinal fluid. The Journal of medical research 31(1):93
Zapf B, Haubner J, Kuchta M, Ringstad G, Eide PK, Mardal KA (2022) Investigating molecular transport in the human brain from MRI with physics-informed neural networks. Scientific Reports 12(1):15475, doi:10.1038/s41598-022-19157-w

Chapter 4

The pulsating brain: An interface-coupled fluid–poroelastic interaction model of the cranial cavity

Marius Causemann, Vegard Vinje, and Marie E. Rognes

Abstract The brain's biomechanics are driven by a dynamic interplay between the tissue, blood, cerebrospinal fluid (CSF), and interstitial fluid and exhibit complex fluid flow and displacement patterns. Despite being essential to normal brain function, our understanding of intracranial dynamics is still limited, and various diseases are associated with impaired CSF flow and elevated intracranial pressure. Here, we present a computational model of cardiac-induced pulsatile motion inside the human cranial cavity. The CSF flow in the subarachnoid space and ventricular system is modeled using the time-dependent Stokes equations and coupled with Biot's poroelasticity equations in the brain tissue, thus integrating all major intracranial constituents into the modeling approach. Employing the pulsatile inflow of blood into brain tissue as a driver of motion, the model enables us to study the dynamics of the entire intracranial system. In addition to the modeling aspects, we showcase the creation of volumetric multidomain meshes from surface meshes using fTetWild, use the FEniCS extension Multiphenics to solve monolithic multiphysics problems, and generate dynamic, interactive standalone animations of the obtained results with PyVista and K3D Jupyter.

4.1 Introduction

The human cranium is often considered a near-rigid vault in which three main components interact: the vasculature (arteries, capillaries, and veins), compartments filled with cerebrospinal fluid (CSF), and the brain parenchyma (Baldent, 2014). Since all these constituents are largely incompressible, any addition of volume leads to a rapid increase of intracranial pressure (ICP) and subsequent fluid flux. Indeed, the temporal change of cerebral blood volume during the cardiac cycle induces a complex periodic pattern of CSF and blood flow and tissue motion (Wagshul et al., 2011). More specifically, during systole, the arterial blood flow into the brain exceeds

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_4

the venous outflow, thereby expanding the brain's volume, causing a rise in ICP and inducing CSF flow both within the cranium and through the spinal canal into the spinal subarachnoid space. Subsequently, diastolic venous outflow dominates the vascular dynamics, leading to a decrease in ICP and a reversal of CSF flow (Baldent, 2014).

To model this brain–CSF interplay, we divide the cranial vault into two three-dimensional domains: the brain parenchyma, denoted by Ω_p, and the surrounding CSF-filled spaces, labeled Ω_f (Fig. 4.1). The two domains share a common interface $\Sigma = \Omega_f \cap \Omega_p$ with a normal vector $\mathbf{n}$, pointing from Ω_f to Ω_p on Σ (and outward on the boundary $\partial\Omega$). Further, Γ_{skull} is the outer boundary of the CSF space where the rigid skull encloses the cranial cavity. The lower boundary of the domain is split into two parts: the caudal continuation of the spinal cord, labeled Γ_{SC}, and Γ_{SAS}, which denotes the boundary to the spinal SAS.

4.1.1 The Brain Parenchyma as a Poroelastic Medium

We represent the brain tissue as a linear poroelastic medium with a single fluid network describing the extracellular CSF/interstitial fluid (ISF) space. The equations of linear poroelasticity describe the macroscopic interaction between the deformation of an elastic solid and a fluid occupying the pore space and enforce momentum conservation of the elastic skeleton and fluid mass conservation within the medium. These equations can be represented with three fields: the displacement $\mathbf{d}$, fluid (pore) pressure p^p, and the additional total pressure $\phi = \alpha p^p - \lambda \operatorname{div} \mathbf{d}$ (Lee et al., 2017; Oyarzúa and Ruiz-Baier, 2016). With the infinitesimal strain tensor $\varepsilon(\mathbf{d}) = \frac{1}{2}(\nabla \mathbf{d} + \nabla \mathbf{d}^T)$ and a volume source term g, the equations are

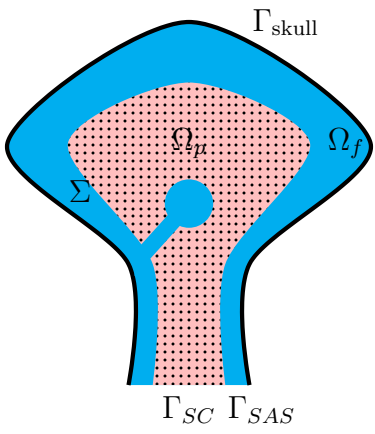


Fig. 4.1 The brain parenchyma (Ω_p, pink) and the CSF-filled spaces (Ω_f, blue). The circular CSF-filled space at the center represents the ventricular system, connected to the cranial subarachnoid space through a narrow opening. The interface of both domains is denoted by Σ. Additionally, the boundaries Γ_{skull} at the skull, Γ_{SC} at the spinal cord, and Γ_{SAS} at the spinal SAS are highlighted.

$$-\mathbf{div}\Big(2\mu_s\varepsilon(\mathbf{d}) - \phi\mathbf{I}\Big) = 0 \qquad \text{in } \Omega_p \times (0,T), \tag{4.1a}$$

$$\phi - \alpha p^p + \lambda \operatorname{div} \mathbf{d} = 0 \qquad \text{in } \Omega_p \times (0,T), \tag{4.1b}$$

$$\left(c + \frac{\alpha^2}{\lambda}\right)\partial_t p^p - \frac{\alpha}{\lambda}\partial_t \phi - \operatorname{div}\left(\frac{\kappa}{\mu_f}\nabla p^p\right) = g \qquad \text{in } \Omega_p \times (0,T), \tag{4.1c}$$

where κ represents the permeability, c the specific storage coefficient, and α the Biot–Willis coefficient. The identity operator is $\mathbf{I}$. The linear isotropic solid matrix is parameterized with the Lamé constants μ_s and λ, while μ_f denotes the viscosity of the permeating fluid. Note that Equation (4.1a) states the conservation of momentum and extends the equations of linear elasticity with the fluid pressure acting on the elastic solid. Further, Equation (4.1c) is motivated by the conservation of fluid mass and combines a Darcy flow model with the effect of the solid deformation. The model was originally proposed to describe geomechanics (Biot, 1941) but has since been widely adopted in biomechanical applications.

4.1.2 Stokes Flow in CSF Spaces

The flow in the CSF compartments can be characterized as a free fluid flow with a low Reynolds number and is hence described by the time-dependent Stokes equations for the CSF velocity $\mathbf{u}$ and fluid pressure p^f. Under these assumptions, the equations read as follows:

$$\rho_f \partial_t \mathbf{u} - \mathbf{div}\Big(2\mu_f\varepsilon(\mathbf{u}) - p^f\mathbf{I}\Big) = 0 \qquad \text{in } \Omega_f \times (0,T), \tag{4.2a}$$

$$\operatorname{div} \mathbf{u} = 0 \qquad \text{in } \Omega_f \times (0,T), \tag{4.2b}$$

with constant CSF density ρ_f and constant CSF viscosity μ_f.

4.1.3 Net Blood Flow as a Driver of Pulsatility

We assume a brain-wide expansion and contraction of blood vessels caused by the net flow of blood into the brain parenchyma as the main driver of motion and model it by a prescribed pulsatile source term g in (4.1c). Hence, we consider the net blood flow as a source term in the CSF/ISF compartment. While this might seem an oversimplification at first, it can be justified by the similarity of an inflow of blood and ISF on the macroscopic level: both lead to a volumetric expansion of the brain parenchyma and an increase in pore pressure. The arterial and venous flow in and out of the cranium is measurable by modern imaging techniques and allows computation of the net cerebral blood flow over the cardiac cycle (Baldent, 2014). However, for simplicity, we let g be a spatially uniform sinusoidal curve in time with frequency f and amplitude A:

$$g(t) = A \cdot \sin(2\pi f t). \tag{4.3}$$

4.1.4 Tissue–CSF Interaction Coupling Conditions

In addition to modeling the mechanics of both domains, we use suitable interface conditions to account for the interaction between the brain tissue and the surrounding CSF. We enforce continuity of the normal flux on the interface (4.4a), momentum conservation (4.4b), and the balance of total normal stress (4.4c). The last interface condition, (4.4d), is the Beavers–Joseph–Saffman condition, which relates the jump in tangential velocities across the interface to the shear stress on the free-flow side of the interface (Beavers and Joseph, 1967; Saffman, 1971). Accordingly, we require the following equations to hold on the interface Σ between the porous parenchyma and the CSF-filled spaces:

$$\mathbf{u} \cdot \mathbf{n} = \left(\partial_t \mathbf{d} - \frac{\kappa}{\mu_f} \nabla p^p \right) \cdot \mathbf{n} \qquad \text{on } \Sigma \times (0,T), \tag{4.4a}$$

$$\left(2\mu_f \varepsilon(\mathbf{u}) - p^f \mathbf{I} \right) \mathbf{n} = \left(2\mu_S \varepsilon(\mathbf{d}) - \phi \mathbf{I} \right) \mathbf{n} \qquad \text{on } \Sigma \times (0,T), \tag{4.4b}$$

$$-\mathbf{n} \cdot \left(2\mu_f \varepsilon(\mathbf{u}) - p^f \mathbf{I} \right) \mathbf{n} = p^p \qquad \text{on } \Sigma \times (0,T), \tag{4.4c}$$

$$-\tau_i \cdot \left(2\mu_f \varepsilon(\mathbf{u}) - p^f \mathbf{I} \right) \mathbf{n} = \frac{\gamma \mu_f}{\sqrt{\kappa}} (\mathbf{u} - \partial_t \mathbf{d}) \cdot \tau_i \qquad \text{on } \Sigma \times (0,T), \quad i = 1,2. \tag{4.4d}$$

Complementing the normal $\mathbf{n}$, we define orthogonal vectors tangent to the interface τ_i $(i = 1, 2)$, with $\gamma > 0$ as the slip rate coefficient, a dimensionless constant that depends on the microstructure of the porous medium.

4.1.5 Boundary and Initial Conditions

Assuming a rigid skull, we set no-slip conditions on the skull boundary Γ_{skull}:

$$\mathbf{u} = 0 \qquad \text{on } \Gamma_{\text{skull}} \times (0,T).$$

For the spinal cord boundary Γ_{SC}, we assume no displacement and no flux:

$$\mathbf{d} = 0 \quad \text{and} \quad \frac{\kappa}{\mu_f} \nabla p^p \cdot \mathbf{n} = 0 \qquad \text{on } \Gamma_{\text{SC}} \times (0,T).$$

Finally, we impose a zero-traction condition at the spinal SAS, allowing for the pulsatile motion of CSF in and out of the domain:

$$\left(2\mu_f \varepsilon(\mathbf{u}) - p^f \mathbf{I}\right) \cdot \mathbf{n} = 0 \qquad \text{on } \Gamma_{\text{SAS}} \times (0, T).$$

We assume that the system is initially at rest, with an initial pore pressure p_0:

$$\begin{aligned} &\mathbf{u} = \mathbf{0} && \text{on } \Omega_f \times \{0\}, \\ &\mathbf{d} = \mathbf{0}, \quad p^p = p_0 && \text{on } \Omega_p \times \{0\}. \end{aligned}$$

4.2 Multidomain Mesh Generation for a Brain–CSF Model

Current medical imaging technologies offer high-resolution spatial images of the whole cranium. Often, after preprocessing and segmentation of the image, a surface mesh in STL format of each of the relevant parts can be obtained. We demonstrate here how to generate a *multidomain* mesh (a mesh with several subdomains and a matching interface) suitable for multiphysics simulations with FEniCS with such surfaces meshes as a starting point. We showcase the meshing tool fTetWild and demonstrate its capability to compute boolean operations on surface meshes (Hu et al., 2020). We have found that fTetWild is particularly well suited for mesh generation from imaging data due to its strong robustness (with no assumptions on mesh manifoldness, watertightness, and the absence of self-intersections), since the algorithm constructs and meshes an envelope of user-defined accuracy around the triangles contained in the input surface meshes.

4.2.1 Defining Domains by Surfaces

Solving the tissue–CSF interaction model on a realistic 3D domain exceeds the computational resources of a standard laptop. To allow the reproduction of our results without access to supercomputing resources, we work with a low-resolution, idealized representation of the human cranium. Specifically, we construct an idealized geometry from the following parts (Fig. 4.2):

- An innermost sphere representing the ventricles,
- A second, larger sphere representing the parenchymal tissue,
- An outermost sphere for the skull,
- A thin cylinder representing the spinal cord,
- A thicker cylinder for the CSF-filled space surrounding the spinal cord, and
- A small, thin cylinder connecting the ventricle with the SAS for the aqueduct.

We generate these surface meshes (shown for the ventricle and the aqueduct here) using PyVista and save them in the STL format:

```
import pyvista as pv
stl_directory = Path("mesh/stls/")
ventricle = pv.Sphere(radius = 0.02)
aqueduct = pv.Cylinder(
    center=(0, 0.03, -0.03), direction=(0, 1, -1),
    radius=0.004, height=0.06).triangulate()
aqueduct.save(stl_directory / "aqueduct.stl")
ventricle.save(stl_directory / "ventricle.stl")
```

4.2.2 Creating the Geometric Representation

Next, we specify a *constructive solid geometry tree*, describing the order and type of boolean operations carried out on the surface meshes:

- First, we take the union of brain tissue components, with the parenchyma and brainstem.
- Then, we subtract the ventricle and aqueduct parts from the tissue.
- Finally, we take the union of the outermost sphere and cylinder with the result of the previous steps.

The code is as follows:

```
import wildmeshing as wm
tetra = wm.Tetrahedralizer(epsilon=0.002, edge_length_r=0.05,
                           coarsen=False)
tetra.load_csg_tree(json.dumps(
    {"operation":"union",
     "right":
        {
        "operation":"difference",
            "left":{"operation":"union",
                "left": str(stl_directory / "parenchyma.stl"),
                "right": str(stl_directory / "cord.stl")},
            "right":{"operation":"union",
                "left": str(stl_directory / "aqueduct.stl"),
                "right":str(stl_directory / "ventricle.stl")},
        },
     "left":
        {"operation":"union",
            "left": str(stl_directory / "skull.stl"),
            "right":str(stl_directory / "canal.stl")},
        }
))
tetra.tetrahedralize()
point_array, cell_array, marker = tetra.get_tet_mesh()
```

The resulting multidomain geometry and mesh are visualized in Figure 4.2. The maximum distance between the input surfaces and the surface of the generated volumetric mesh can be controlled using the `epsilon` parameter (specifying the relative envelope size), and the desired relative edge length is specified with `edge_length_r` .

Moving to a more complex mesh requires only a change in the input surface meshes and the mesh resolution but does not introduce additional difficulties. As an example, a mesh generated by the described workflow based on the surface meshes used by Causemann et al. (2022) is shown in Figure 4.3. The corresponding code is included in the example notebooks.[1]

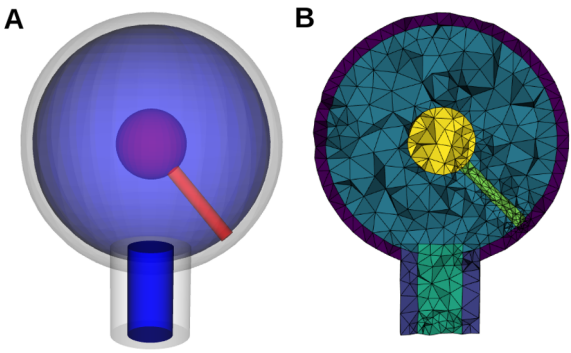
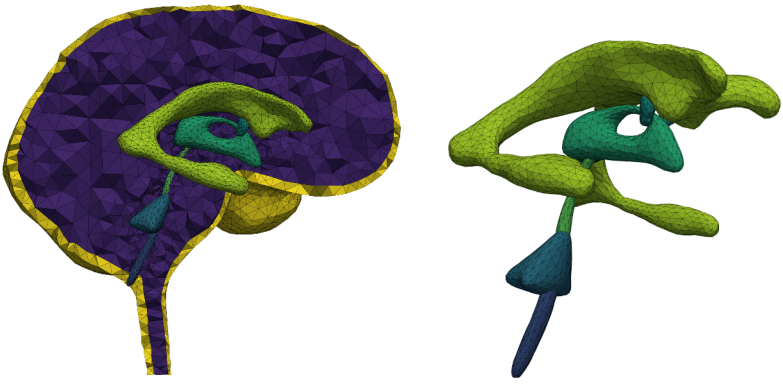


Fig. 4.2 (A) An idealized 3D cranial geometry consisting of a ventricle and aqueduct (red), the parenchyma and brainstem (blue), and the surrounding CSF space (gray/transparent). (B) The resulting multidomain mesh generated by fTetWild.



(a) (b)

Fig. 4.3: a) Example of an image-derived mesh generated with fTetWild and based on data from Causemann et al. (2022). b) The ventricular system of the image-derived mesh.

[1] See `https://zenodo.org/records/10818803`.

4.2.3 Marking the Interface and Boundaries

fTetWild returns a marker function with different identifiers for each of the subdomains but does not generate facet markers for internal interfaces or exterior boundaries. We therefore use FEniCS to loop over all facets in the mesh, check the subdomain IDs of the corresponding cells, and identify the internal interfaces. Similarly, we mark the external boundaries (Γ_{Skull}, Γ_{SAS}, and Γ_{SC}) with a combination of the facet location and subdomain ID of the corresponding subdomain.

4.3 Discretization and Implementation of the Fluid–poroelastic Problem

Following Ruiz-Baier et al. (2022), we discretize the model in space using a monolithic finite element formulation and employ an implicit Euler scheme for time integration. Here, we state the variational form of the time-discretized problem and discuss its implementation with Multiphenics (Ballarin, 2023), a FEniCS extension that provides key functionality for subdomain restricted variables.

4.3.1 Variational Form of the Fluid–poroelastic Interface Problem

In light of the boundary conditions defined in Section 4.1.5, we define the following function spaces:

$$\mathbf{H}^{\mathbf{1}}_{\text{SK}}(\Omega_f) = \{\mathbf{v} \in H^1(\Omega_f)^d \colon \mathbf{v}\,|_{\Gamma_{\text{skull}}} = 0\},$$
$$\mathbf{H}^{\mathbf{1}}_{\text{SC}}(\Omega_p) = \{\mathbf{v} \in H^1(\Omega_p)^d \colon \mathbf{v}\,|_{\Gamma_{\text{SC}}} = 0\},$$

where $H^1(\Omega_i)$ is the standard first-order Sobolev space of scalar functions with derivatives in $L^2(\Omega_i)$ equipped with the classical Sobolev norm on the domain Ω_i for $i \in \{P, F\}$.

After applying the implicit Euler method for time discretization with a fixed step size $\Delta t > 0$ for N_t time steps, we obtain the following semi-discrete weak form of the coupled problem: given suitable initial conditions $\mathbf{u}_0$, p_0^f, $\mathbf{d}_0$, and p_0^p, find $\mathbf{u}_n \in \mathbf{H}_{\text{SK}}(\Omega_f)$, $p_n^f \in L^2(\Omega_f)$, $\mathbf{d}_n \in \mathbf{H}_{\text{SC}}(\Omega_p)$, $p_n^p \in H^1(\Omega_p)$, and $\phi_n \in L^2(\Omega_p)$ for $n \in \{1, 2, \ldots, N_t\}$ such that

$$
\begin{aligned}
a^f(\mathbf{u}_n, \mathbf{v}) + b_1^f(\mathbf{v}, p_n^f) + b_2^\Sigma(\mathbf{v}, p_n^p) + b_3^\Sigma(\mathbf{v}, \Delta t^{-1}\mathbf{d}_n) &= F_n^f(\mathbf{v}) && \forall \mathbf{v} \in \mathbf{H}_{\mathrm{SK}}(\Omega_f)\\
b_1(u_n, q^f) &= 0 && \forall q^f \in L^2(\Omega_f)\\
b_3^\Sigma(\mathbf{u}_n, \mathbf{w}) + b_4^\Sigma(\mathbf{w}, p_n^p) + a_1^P(\mathbf{d}_n, \mathbf{w}) + b_1^p(\mathbf{w}, \phi_n) &= F_n^p(\mathbf{w}) && \forall \mathbf{w} \in \mathbf{H}_{\mathrm{SC}}(\Omega_p)\\
-b_2^\Sigma(\mathbf{u}_n, q^p) - b_4^\Sigma(\Delta t^{-1}\mathbf{d}_n, q^p) & \\
+a_2^p(p_n^p, q^p) - b_2^p(\Delta t^{-1}\phi_n, q^p) &= G_n(q^p) && \forall q^p \in H^1(\Omega_p)\\
b_1^p(\mathbf{d}_n, \psi) + b_2^p(\psi, p_n^p) - a_3^p(\phi_n, \psi) &= 0 && \forall \psi \in L^2(\Omega_p)
\end{aligned}
$$

with the variational forms

$$
\begin{aligned}
a^f(\mathbf{u}, \mathbf{v}) &= \rho_f \int_{\Omega_f} \Delta t^{-1}\mathbf{u}\cdot\mathbf{v} + 2\mu_f \int_{\Omega_f} \epsilon(\mathbf{u}) : \epsilon(\mathbf{v}) + \frac{\gamma\mu_f}{\sqrt{\kappa}}\langle P_T\mathbf{u}, P_T\mathbf{v}\rangle_\Sigma\\
b_1^f(\mathbf{v}, q^f) &= -\int_{\Omega_f} q^f \ \mathrm{div}\ \mathbf{v},\\
b_2^\Sigma(\mathbf{v}, q^p) &= \langle q^p, \mathbf{v}\cdot\mathbf{n}\rangle_\Sigma\\
b_3^\Sigma(\mathbf{v}, \mathbf{d}) &= -\frac{\gamma\mu_f}{\sqrt{\kappa}}\langle P_T\mathbf{v}, P_T\mathbf{d}\rangle_\Sigma,\\
b_4^\Sigma(\mathbf{w}, q^p) &= -\langle q^p, \mathbf{w}\cdot\mathbf{n}\rangle_\Sigma,\\
a_1^p(\mathbf{d}, \mathbf{w}) &= 2\mu_S \int_{\Omega_p} \epsilon(\mathbf{d}) : \epsilon(\mathbf{w}) + \frac{\gamma\mu_f}{\sqrt{\kappa}}\langle P_T\Delta t^{-1}\mathbf{d}, P_T\mathbf{v}\rangle_\Sigma,\\
b_1^p(\mathbf{w}, \psi) &= -\int_{\Omega_p} \psi \ \mathrm{div}\ \mathbf{w},\\
a_2^p(p^p, q^p) &= \left(c + \frac{\alpha^2}{\lambda}\right)\int_{\Omega_p} \Delta t^{-1} p^p q^p + \int_{\Omega_p} \frac{\kappa}{\mu_f}\nabla p^p \cdot \nabla q^p,\\
b_2^p(\psi, q^p) &= \frac{\alpha}{\lambda}\int_{\Omega_p} \psi q^p,\\
a_3^p(\phi, \psi) &= \frac{1}{\lambda}\int_{\Omega_p} \phi\psi
\end{aligned}
$$

and the linear functionals

$$
\begin{aligned}
F_n^f(\mathbf{v}) &= b_3^\Sigma(\mathbf{v}, \mathbf{d}_{i-1}/\Delta t) + \rho_f \int_{\Omega_f} \mathbf{u}_{i-1}/\Delta t \cdot \mathbf{v},\\
F_n^p(\mathbf{w}) &= -b_3^\Sigma(\mathbf{w}, \mathbf{d}_{i-1}/\Delta t),\\
G_n(q^p) &= -b_4^\Sigma(\mathbf{d}_{i-1}/\Delta t, q^p) - b_2^p(\phi_{i-1}/\Delta t, q^p) + \int_{\Omega_p} g q^p + \left(c + \frac{\alpha^2}{\lambda}\right) p_{i-1}^p q^p.
\end{aligned}
$$

Here, P_T denotes the projection on the interface tangent and $\langle\cdot,\cdot\rangle_\Sigma$ the duality pairing between the trace space $H^{1/2}(\Sigma)$ and its dual.

Using a Taylor–Hood-type discretization, we approximate the tissue displacement, fluid velocity, and pore pressure with continuous piecewise quadratic polynomials and use continuous piecewise linear functions for the total pressure and fluid pressure. Note that any other pair of Stokes-stable finite elements also results in a stable scheme (Ruiz-Baier et al., 2022).

4.3.2 Implementation

To facilitate the finite element implementation, we rely on Multiphenics, a Python library providing additional features for block systems and function spaces on subdomains in FEniCS. While such features are also available and under further development in FEniCS(x), Multiphenics' robustness and convenient interface make it particularly useful for multiphysics problems. Concretely, we can define our function spaces for fluid velocity $\mathbf{u}$, fluid pressure p^f, displacement $\mathbf{d}$, pore pressure p^p, and total pressure ϕ as follows:

```
from multiphenics import *
W = FunctionSpace(mesh, "Lagrange", 1)
W2 = FunctionSpace(mesh, "Lagrange", 2)
V = VectorFunctionSpace(mesh, "Lagrange", 2)
H = BlockFunctionSpace([V, W, V, W2, W],
              restrict=[fluidrestriction,
                  fluidrestriction,
                  porousrestriction,
                  porousrestriction,
                  porousrestriction])

trial_functions = BlockTrialFunction(H)
u, pF, d, pP, phi = block_split(trial_functions)

test_functions = BlockTestFunction(H)
v, qF, w, qP, psi = block_split(test_functions)
```

When defining a `BlockFunctionSpace`, Multiphenics allows specifying spatial restrictions using the `restrict` argument with a `MeshRestriction`. The `MeshRestriction` holds the information on the spatial extent of the desired function space and is generated from the subdomain markers provided by fTetWild.

Next, we implement the individual terms for each of the blocks of the monolithic form (with the example shown here for $a^f(\mathbf{u}, \mathbf{v})$) and define the linear system:

```
# Create convenience functions for recurring terms
eps = lambda u: sym(grad(u))
P_t = lambda u: u - dot(u, n) * n

def tang_interf(u, v):
    return inner(P_t(u("+")), P_t(v("+")))*ds_Sigma
```

```
# Define the individual blocks of the variational form
def a_f(u, v):
    return rho_f*dot(u/dt, v)*dxF \
        + 2*mu_f*inner(eps(u), eps(v))*dxF \
        + (gamma*mu_f/sqrt(kappa))*tang_interf(u, v)

...

lhs = [[a_f(u, v), b_1_f(v, pF), b_3_Sig(v, d/dt), b_2_Sig(v, pP)
                                        , 0],
       [b_1_f(u, qF), 0, 0, 0, 0],
       [b_3_Sig(u, w), 0, a_1_p(d, w), b_4_Sig(w, pP), b_1_p(w,
                                            phi)],
       [b_2_Sig(u, qP), 0, b_4_Sig(d/dt, qP), -
        a_2_p(pP, qP), b_2_p(phi/dt, qP)],
       [0, 0, b_1_p(d, psi), b_2_p(psi, pP), -a_3_p(phi, psi)]]

rhs = [F_f_n(v), 0, F_p_n(w), -G_n(qP), 0]
```

Multiphenics offers high-level functions for assembling the block system, setting boundary conditions, and solving the block system. Using the multifrontal direct solver MUMPS (Amestoy et al., 2000), we first assemble the block system and then set up a `PETScLUSolver` solver, which will store the factorization computed in the first `solve` call and hence substantially reduce the cost of subsequent time steps. Additionally, we define the relevant Dirichlet boundary conditions and the time-dependent source term g before starting the main time-stepping loop, in which we update the source term, reassemble the right-hand side, account for the boundary conditions, and finally call the `solve` function of the direct solver:

```
AA = block_assemble(lhs, keep_diagonal=True)
solver = PETScLUSolver(AA, "mumps")

bc_d = DirichletBC(H.sub(2), Constant((0,0,0)),bm, spinal_sas_id)
bc_u = DirichletBC(H.sub(0), Constant((0,0,0)),bm, skull_id)
bcs = BlockDirichletBC([bc_d, bc_u])

g = Expression("A*sin(2*M_PI*f*t)", f =1, t=0, A=0.01, degree=0)
times = np.linspace(0, T, num_steps + 1)

for t in times:
    g.t = t
    FF = block_assemble(rhs)
    bcs.apply(FF)
    solver.solve(block_function.block_vector(), FF)
    block_function.block_vector().block_function().apply("to
                                        subfunctions")
```

4.4 Visualization with PyVista and K3D

PyVista is a Python library providing high-level access to the Visualization Toolkit (VTK) for powerful visualization of complex 3D datasets (Sullivan and Kaszynski, 2019). Here, we showcase its use in conjunction with K3D Jupyter to generate time-dependent, interactive animations as standalone HTML pages, which are easy to embed in Web applications and share with colleagues. Compared to creating animations with GUI-based software such as ParaView, scripting the visualizations allows for their regeneration after changes of the model or input data. Additionally, it provides the freedom to use the rich Python ecosystem to manipulate the data. While scripting 3D animations is possible with other packages (e.g., Mayavi, ParaView, or VTK), PyVista is especially compelling due to its broad VTK-based functionality and intuitive interface.

Since PyVista does not natively support time-dependent datasets, we store the data corresponding to each time step in a new array of a Pyvista `UnstructuredGrid` object:

```
grid = pv.read("mesh/mesh.xdmf")
with meshio.xdmf.TimeSeriesReader("results.xdmf") as reader:
    points, cells = reader.read_points_cells()
    for k in range(reader.num_steps):
        t, point_data, cell_data = reader.read_data(k)
        for var, data in point_data.items():
            grid[f"{var}_{k}"] = data
```

Next, we extract the porous and fluid subdomains and apply the `clip` filter to show the interior of the domain. To visualize the flow field in the CSF-filled domains, we compute arrow glyphs for each time step and scale them with maximum velocity:

```
fluid = grid.extract_cells(grid["subdomains"] == fluid_id)
fluid_clip = fluid.clip()
arr_max = max([np.linalg.norm(fluid_clip[f"u_{t}"], axis=1).max()
               for t in time_indices])
arrows = []
for t in tqdm(time_indices):
    arr = fluid_clip.glyph(orient=f"u_{t}", scale=f"u_{t}",
                           factor=0.1/arr_max, tolerance=.005)
arrows.append(arr)
```

To use K3D, we convert the PyVista objects into K3D objects and add them to a `k3d.plot`. For that purpose, we use K3D's `vtk_poly_data` function, generate an initial object, and set the time-dependent properties (point/vertex location, color attribute) with a Python dictionary mapping the timepoint. We demonstrate this for the arrow glyphs:

```
animation_t = np.linspace(0, 10, len(time_idx))
k3d_arrow = k3d.vtk_poly_data(arrows[0],
                              color=hexcolor("white"),
```

```
                                    side="double")
k3d_arr.vertices = {animation_t[t]: arrows[t].points for t in
                                      time_idx}

pl = k3d.plot(...)
pl += k3d_arrows
pl += ...
pl.display()
```

A screenshot of the resulting interactive figure is shown in Figure 4.4, and the interactive figure is available in the accompanying notebooks.

4.5 Post-processing Aqueduct Flow and ICP

The change in ICP over the cardiac cycle as well as the aqueduct stroke volume (total volume of CSF moved back and forth through the aqueduct during each cycle) are important clinical indicators of intracranial pulsatility (Wagshul et al., 2011). Having defined the interface between the fourth ventricle and the aqueduct during the meshing step, we can compute the flow rate as a surface integral over the corresponding internal facets:

```
u_time_series = [...]
dS = Measure("dS", domain=mesh, subdomain_data=boundary_marker)
n = FacetNormal(mesh)
aq_flow = []
for u in u_time_series:
    aq_flow.append(assemble(inner(u,n("+"))*dS(aqueduct_V4_id)))
```

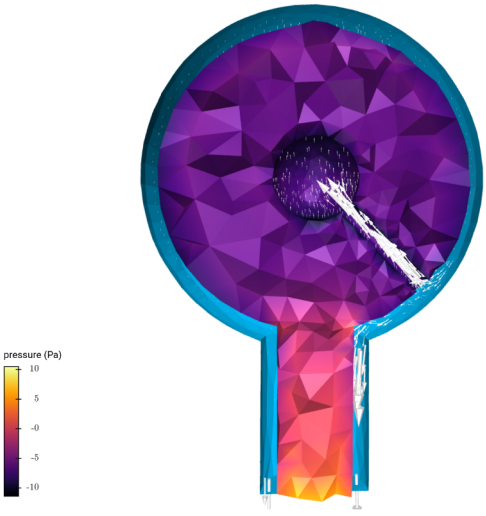


Fig. 4.4 Screenshot of an interactive figure at $t = 0.18$ s generated with K3D. The visualization shows the pore pressure in the brain tissue (color coded), with the arrows indicating the CSF velocities in the CSF-filled spaces in response to arterial blood inflow. The blue surface in the back represents the outer surface of the CSF-filled spaces.

We observe peak aqueduct flow rates of about 0.03 ml/s, and after a few cycles the aqueduct flow synchronizes with the net blood inflow into the brain tissue (Fig. 4.5a).

To compute the ICP and investigate ICP gradients, we place two virtual probes in the CSF-filled spaces: one in the middle of the ventricular sphere $x_V = (0, 0, 0)$ and one at the outer end of the aqueduct cylinder in the SAS ($x_{AQ} = (0, 0.055, -0.055)$):

```
ICP_time_series = [...]
ventricle_mid = Point(0,0,0)
icp_ventricle = [p(ventricle_mid) for p in ICP_time_series]
```

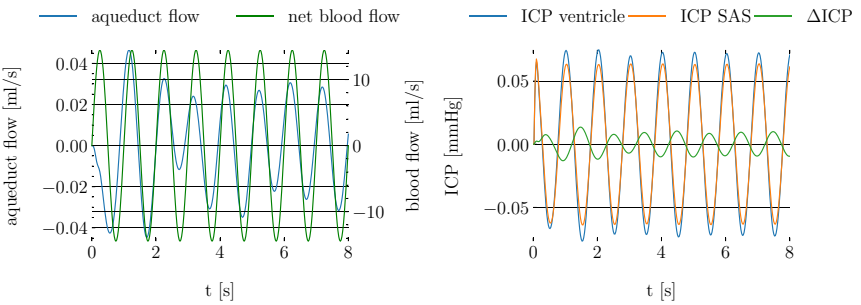


Fig. 4.5: a) The aqueduct CSF flow over multiple cardiac cycles. b) The ICP measured in the ventricular sphere and in the SAS at the outer end of the aqueduct and the pressure difference between the two virtual probe points (ΔICP).

Compared with clinically reported values for temporal ICP changes of 5–10 mmHg (Wagshul et al., 2011), we find relatively small pressure oscillations in both the ventricles and the SAS (less than 0.1 mmHg, Fig. 4.5b). This discrepancy is likely caused by the spinal outflow boundary condition: without additional resistance or an increase in pressure in the spinal SAS, CSF freely leaves the cranium and hence ICP increases only slightly, indicating that accounting for spinal resistance is crucial for physiological ICP values.

4.6 Summary and Outlook

In this chapter, we have presented a model of the dynamic interplay between blood flow, tissue motion, and CSF flow in the human cranium during the cardiac cycle. In addition, we demonstrated key technical steps to conduct multiphysics simulations on complex, image-derived geometries. In particular, we have shown how to generate volumetric multidomain meshes from surface meshes using fTetWild, use the FEniCS extension Multiphenics to set up monolithic forms with function spaces

defined only on subdomains, and create dynamic, interactive standalone animations of the results with PyVista and K3D.

However, we only briefly touched upon the model results and their physiological relevance. Exploring these in more depth, Causemann et al. (2022) has provided a detailed comparison of clinical measurements and results obtained with a refined version of the same model using an image-derived 3D geometry. Further, Unnerbäck et al. (2018) and Domogo et al. (2023) explored the determinants of the ICP curve morphology, whereas Vinje et al. (2019) investigated the respiratory and cardiac contributions to CSF flow and ICP dynamics. Additionally, Linninger, Andreas A. and Tangen, Kevin and Hsu, Chih-Yang and Frim, David (2016) have reviewed the physiology and modeling approaches on CSF dynamics. Finally, measuring blood flow, CSF flow oscillations, and electrophysiological activity in humans during sleep, Fultz et al. (2019) found important links between CSF dynamics and neural and hemodynamic rhythms. Exploring the coupling between these phenomena presents a multitude of research directions related to the modeling of the brain's biomechanics.

References

Amestoy P, Duff I, L'Excellent JY (2000) Multifrontal parallel distributed symmetric and unsymmetric solvers. Computer Methods in Applied Mechanics and Engineering 184(2):501–520, doi:10.1016/S0045-7825(99)00242-X

Ballarin F (2023) Multiphenics - easy prototyping of multiphysics problems in FEniCS. `https://multiphenics.github.io/`

Baldent O (2014) Imaging of the cerebrospinal fluid circulation, Cambridge University Press, pp 121–138. doi:10.1017/CBO9781139382816.013, editor: Rigamonti, Daniele

Beavers GS, Joseph DD (1967) Boundary Conditions at a Naturally Permeable Wall. Journal of Fluid Mechanics 30(1):197–207, doi:10.1017/S0022112067001375

Biot MA (1941) General Theory of Three-Dimensional Consolidation. Journal of Applied Physics 12(2):155–164, doi:10.1063/1.1712886

Causemann M, Vinje V, Rognes ME (2022) Human intracranial pulsatility during the cardiac cycle: a computational modelling framework. Fluids and Barriers of the CNS 19(1):1–17, doi:10.1186/s12987-022-00376-2

Domogo AA, Reinstrup P, Ottesen JT (2023) Mechanistic-mathematical modeling of intracranial pressure (ICP) profiles over a single heart cycle. The fundament of the ICP curve form. Journal of Theoretical Biology 564:111451, doi:10.1016/j.jtbi.2023.111451

Fultz NE, Bonmassar G, Setsompop K, Stickgold RA, Rosen BR, Polimeni JR, Lewis LD (2019) Coupled electrophysiological, hemodynamic, and cerebrospinal fluid oscillations in human sleep. Science 366(6465):628–631, doi:10.1126/science.aax5440

Hu Y, Schneider T, Wang B, Zorin D, Panozzo D (2020) Fast Tetrahedral Meshing in the Wild. ACM Trans Graph 39(4), doi:10.1145/3386569.3392385

Lee JJ, Mardal KA, Winther R (2017) Parameter-Robust Discretization and Preconditioning of Biot's Consolidation Model. SIAM Journal on Scientific Computing 39(1):A1–A24, doi:10.1137/15M1029473

Linninger, Andreas A and Tangen, Kevin and Hsu, Chih-Yang and Frim, David (2016) Cerebrospinal Fluid Mechanics and Its Coupling to Cerebrovascular Dynamics. Annual Review of Fluid Mechanics 48(1):219–257, doi:10.1146/annurev-fluid-122414-034321

Oyarzúa R, Ruiz-Baier R (2016) Locking-Free Finite Element Methods for Poroelasticity. SIAM Journal on Numerical Analysis 54(5):2951–2973, doi:10.1137/15M1050082

Ruiz-Baier R, Taffetani M, Westermeyer HD, Yotov I (2022) The Biot-Stokes coupling using total pressure: Formulation, analysis and application to interfacial flow in the eye. Computer Methods in Applied Mechanics and Engineering 389:114384, doi:10.1016/j.cma.2021.114384

Saffman PG (1971) On the Boundary Condition at the Surface of a Porous Medium. Studies in Applied Mathematics 50(2):93–101, doi:10.1002/sapm197150293

Sullivan CB, Kaszynski AA (2019) PyVista: 3D plotting and mesh analysis through a streamlined interface for the Visualization Toolkit (VTK). Journal of Open Source Software 4(37):1450, doi:10.21105/joss.01450

Unnerbäck M, Ottesen JT, Reinstrup P (2018) ICP curve morphology and intracranial flow-volume changes: a simultaneous ICP and cine phase contrast MRI study in humans. Acta Neurochirurgica 160(2):219–224, doi:10.1007/s00701-017-3435-2

Vinje V, Ringstad G, Lindstrøm EK, Valnes LM, Rognes ME, Eide PK, Mardal KA (2019) Respiratory influence on cerebrospinal fluid flow – a computational study based on long-term intracranial pressure measurements. Scientific Reports 9(1):9732, doi:10.1038/s41598-019-46055-5

Wagshul ME, Eide PK, Madsen JR (2011) The pulsating brain: A review of experimental and clinical studies of intracranial pulsatility. Fluids and Barriers of the CNS 8(1):1–23, doi:10.1186/2045-8118-8-5

Chapter 5

Quantifying cerebrospinal fluid tracer concentration in the brain

Bastian Zapf, Lars Magnus Valnes, Kent-Andre Mardal, and Ludmil Zikatanov

Abstract Medical imaging utilizing contrast tracers has in recent years led to a new understanding of molecular transport dynamics and pathways in the living brain, the glymphatic system. In humans, intrathecal contrast enhanced magnetic resonance imaging (MRI) has been used as a major tool in clinical studies of the glymphatic system. While such image sequences have allowed novel qualitative insights, quantitative analysis requires careful image preprocessing. This chapter explains the computational steps needed to obtain quantitative information from a publicly available time series of cerebrospinal fluid contrast-enhanced MRI. All steps are accompanied by code snippets that allow the reader to reproduce the presented results. Finally, we show how to incorporate the preprocessed data into physical simulations of molecular transport in the human brain.

5.1 Introduction

In recent years, imaging the dynamics of intrathecally injected cerebrospinal fluid (CSF) tracers has led to a new understanding of molecular transport in the brain, both in mice (Iliff et al., 2013; Yang et al., 2013) and in humans (Eide et al., 2018; Ringstad et al., 2017; Ringstad and Eide, 2020). As an illustrative example, the images shown in Figure 5.1 demonstrate that CSF tracer reaches all regions of the brain during an imaging period of ~50 hours.

However, only few studies (Filippi and Watts, 2022; Valnes et al., 2020; Vinje et al., 2023) have so far quantified the concentrations of CSF tracer in the brain. This is partly because tracer quantification, in principle, requires special magnetic resonance (MR) sequences that yield MR images known as T1 *maps* (Filippi and Watts, 2022). Alternatively, estimates of the tracer concentration can be obtained from T1-*weighted MR images* by using parameters of the sequence used for the MR imaging (MRI)

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_5

acquisition and manually normalizing the images (Valnes et al., 2020; Vinje et al., 2023).

The difference between these image types is that the voxel values in T1 *map* images correspond directly to a physical quantity, namely, the (voxel–averaged) longitudinal relaxation time (also called T_1 time) of the atomic nuclei making up the tissue in a given voxel. In T1-weighted images, the voxel values are related to this quantity in a nonlinear manner defined by the parameters of the MRI acquisition sequence. For a more detailed description, the interested reader is referred to, for example, Pooley (2005) and Taylor et al. (2016).

In this chapter, we describe all steps, assumptions, and software tools needed to estimate CSF tracer concentration in the brain from a time series of either type of images taken after injection.

In detail, we show how to use FreeSurfer version 7.1.0 (Fischl, 2012) and Python scripts to carry out the following:

- Resample a series of input images to the FreeSurfer standard, that is, 256^3 voxels of size 1 mm $\times$ 1 mm $\times$ 1 mm.
- Create a registration template and register all images to the template.
- Estimate CSF tracer concentration from T1 maps.
- Estimate CSF tracer concentration from T1-weighted images.
- Incorporate the tracer estimates into finite element simulations.

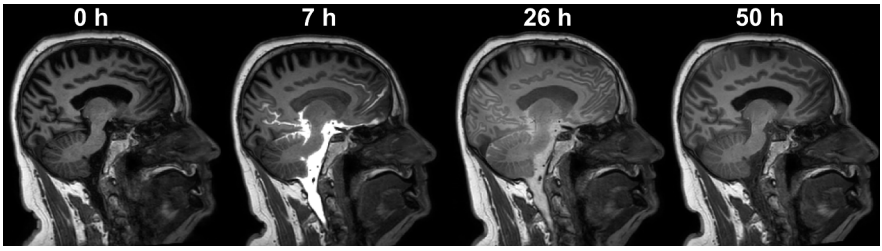


Fig. 5.1: The image shows T1-weighted MR images taken after intrathecal injection of CSF tracer in the example subject. Regions enriched with tracer appear brighter on the image.

5.2 Preliminaries: Software Configurations

For the MRI processing code presented in this chapter to be executable, we assume that FreeSurfer has been installed (see Mardal et al. (2022, Chapter 2) for installation

instructions). The scripts described in this chapter can be obtained by cloning the GitHub repository linked by Zapf et al. (2024a), which also provides installation and usage instructions.

In this chapter, we will assume that all paths are relative to the `csftracermodeling` directory; that is, we run the command `cd csftracermodeling` before executing any of the commands described in this chapter. Next, we assume that FreeSurfer has been run on the MRI data and that the baseline and contrast-enhanced MR images are extracted from DICOM format into a folder `./data/freesurfer/PREREG/` in `.mgz` format. We further assume that the FreeSurfer output folder `./data/freesurfer/mri/` contains the segmentation file `aseg.mgz`. These data are available for download, as described in the README file of Zapf et al. (2024a) For convenience, the final output files that are being created when performing all the steps in this chapter (CSF tracer concentrations, finite element mesh) are included in the data source of (Zapf et al., 2024b).

5.3 Registration of an MRI Time Series

Since the positions of the subjects in the MRI machine differ between acquisitions, we need to align all the images so that it is meaningful to compare intensities in a voxel with indices (ijk) between images taken at different timepoints. This process is called registration, and FreeSurfer provides the program `mri_robust_register`[1] (Reuter et al., 2010) to perform this task. The algorithm finds the affine, that is, rigid translation and rotation, transformation that, when applied to a given input image, minimizes the mismatch to a given target image. We limit ourselves to affine maps in this setting because it is the same subject imaged over a short time span.

In principle, this algorithm could be used to register all images in a time series to a baseline image (pre-contrast). However, the CSF tracer leads to a significantly increased MR signal in most of the cranial compartment compared to the baseline. While the FreeSurfer registration algorithm (Reuter et al., 2010) uses normalization to account for signal differences between the input and target image during registration, the unusual signal differences induced by the CSF tracer can cause registration problems. We hence create an unbiased registration template, which reduces biases in longitudinal studies (Reuter et al., 2012). This template is built from a voxel-wise median of all the images in the time series, registered such that all the images can be registered as well as possible to this template.

First, however, we resample all the images to the FreeSurfer standard format (256^3 voxels) because some MR images can have higher resolutions, significantly increasing the computational time of the registration algorithm, while others may have lower resolutions. We collect all the necessary commands for the registration in a single

[1] See `https://surfer.nmr.mgh.harvard.edu/fswiki/mri_robust_register`.

script, which can be found under `./scripts/data-processing/register.sh`. Here, we give a detailed description of the crucial steps in this script.

To start, we loop over all images in the input folder that are in the `mgz` format and use `mri_convert`[2] from FreeSurfer to resample the images to the FreeSurfer standard:

```
for inputfile in ${INPUTFOLDER}/*.mgz; do
    filename=$(basename $inputfile)
    outputfile=${OUTPUTFOLDER}/${filename}
    mri_convert --conform -odt float ${inputfile} ${outputfile}
done
```

With the option `--conform`, the images are resampled to the FreeSurfer standard voxel size 1 mm^3. The flag `-odt` is shorthand for `--out_data_type`. It is needed because the voxel intensities are not integer values after resampling.

Now, the FreeSurfer command `mri_robust_template`[3] can be used to create the registration template. This program computes a median image of all input images to avoid any bias toward a specific image. This is an iterative process in which several registrations must be computed, and the program can therefore take several minutes to finish.

The command is used as follows:

```
mri_robust_template --mov ${RESAMPLED_IMAGES} --maxit 10 \
    ↪--template ${TEMPLATE} --satit --inittp 1 --fixtp
```

The option `--satit` tells FreeSurfer to automatically determine how outlier voxels should be treated during registration. With `--inittp 1`, the algorithm uses the first image (pre-contrast) as the initial value for the template, and, with `--fixtp`, all the other images are registered to this image. This is useful because we have run the FreeSurfer command `recon-all` (see Mardal et al. (2022, Chapter 3)) with the baseline image (pre-contrast), so all the segmentation files and surfaces generated by FreeSurfer align with this image. With `--inittp 1 --fixtp`, the registration template is automatically aligned with the segmentation and surface files.

Registration is an iterative process, and we specify the maximum number of iterations with `--maxit 10`. Setting this value above the default of six iterations can increase the accuracy of the registration.

Finally, it should be noted that this program can require more memory than is available on personal computers if there are many images in the time series. In this case, the flag `--subsample 200` can be used to resample the images to a resolution of 200^3 voxels (or less if necessary). This reduces the memory required by the program, enabling us to run it on regular laptops.

[2] See `https://surfer.nmr.mgh.harvard.edu/fswiki/mri_convert`.

[3] See `https://surfer.nmr.mgh.harvard.edu/fswiki/mri_robust_template`.

We can now register all the images to the template using FreeSurfer's `mri_robust_register` program:

```
for inputfile in ${INPUTFOLDER}/*.mgz; do
  filename=$(basename $inputfile)
  mri_robust_register --mov ${inputfile} --dst ${TEMPLATE} \
      ↪--mapmov ${OUTPUTFOLDER}/${filename} --maxit 10 --lta \
      ↪./data/freesurfer/LTA/${filename}.lta --iscale --satit
done
```

This command aligns the input image `--mov ${inputfile}` with the target image `--dst ${TEMPLATE}` and stores it to `--mapmov ${OUTPUTFOLDER}/${filename}`. With the argument `--lta`, we specify a storage path to store the affine registration matrix, which can be useful for later analysis. To visually check the registration results, open all images in Freeview to check if they are well aligned to the template using the command

```
$  freeview ${OUTPUTFOLDER}/*
```

It is strongly recommended to always inspect the registration output visually to ensure that the images are properly registered to the baseline image.

5.4 Quantifying Tracer from Contrast-enhanced MRI

The presence of CSF tracer leads to a decrease of the spin–lattice relaxation time T_1 in enriched regions. In T1-weighted images, this increases the MR signal in enriched voxels and makes them appear brighter in the image.

The relaxation time $T_1((i,j,k),t)$ in a voxel (i,j,k) at time t after CSF tracer injection decreases with the contrast agent concentration $c((i,j,k),t)$ (mmol/L) (e.g., Taylor et al., 2016), as follows:

$$\frac{1}{T_1((i,j,k),t)} = \frac{1}{T_1((i,j,k),0)} + r_1 c((i,j,k),t), \tag{5.1}$$

where $T_1((i,j,k),0)$ denotes the T_1 relaxation time at baseline (before injection). The *relaxivity constant* r_1 differs between tracers and depends on other factors, such as the temperature and magnetic field strength. The imaging sequence under consideration here was acquired using the CSF tracer gadobutrol at magnetic field strength 3 T. For these parameters, we can use the measurement value $r_1 = 3.2$ L/mmol s in water at 37 °C from Rohrer et al. (2005).

5.4.1 Quantifying Tracer from T1 Maps

Rearranging (5.1) shows that the concentration of the tracer in every voxel is

$$c((i,j,k),t) = \frac{1}{r_1}\left(\frac{1}{T_1((i,j,k),t)} - \frac{1}{T_1((i,j,k),0)}\right) \tag{5.2}$$

if measurements of T1 are available at all timepoints, as for Watts et al. (2019). In this case, assuming that the files in `./data/freesurfer/REGISTERED` are T1 maps, we can easily compute the concentration with the Python script `estimatec.py`, as described in the accompanying README file under `scripts/data-processing` from Zapf et al. (2024a). By default, the script assumes that the files are named following the convention `YYYYMMDD_HHMMSS.mgz` and finds the baseline map $T_1((i,j,k),0)$ by sorting the files in the input folder. A confirmation prompt asks the user to confirm this selection, and, if it is not correct, the baseline can be manually specified with the optional argument `--baseline`. For convenience, the script will rename the created files in the format `HH.MM.mgz`, where `HH` and `MM` are the numbers of hours and minutes after the baseline image was taken, respectively.

However, T1 maps are acquired with MRI protocols that are not always included in clinical CSF tracer studies with humans Vinje et al. (2023). As a remedy, in the next section we present an estimation of the tracer concentration from T1-weighted images, since these are typically available in such studies.

5.4.2 Quantifying Tracer from T1-weighted Images

Despite the name T1-*weighted image*, the MR signal $S((i,j,k),t)$ in a voxel (i,j,k) of this image is not a quantitative measure of the relaxation time T_1. The relation between the MR signal $S((i,j,k),t)$ and the relaxation time $T_1((i,j,k),t)$ is nonlinear and depends on the details of the MRI protocol used in the image acquisition. As a remedy, we describe how to proceed, following Vinje et al. (2023), and estimate the relaxation times $T_1((i,j,k),t)$ after tracer injection from normalized T1-weighted voxel intensities $S((i,j,k),t)$ relative to the baseline image. We first show how to normalize the images using a manually defined region of interest (ROI). We further need the baseline T1 map $T_1((i,j,k),0)$. If this is not available, we start by showing how to create a synthetic baseline map $T_1((i,j,k),0)$ based on literature values.

Defining a Normalization Region using Freeview

Due to variation in the MRI machine over time, the T1-weighted MR signal can change over time, even in brain regions where no physical change in the tissue is expected. To compensate for this effect, we use an ROI in the brain where the tissue

can be assumed not to change during the imaging period. We then normalize every image with the median MR signal in this region.

To create an ROI, we start by opening the baseline image (the first image in the folder `./data/freesurfer/REGISTERED/`) in Freeview:

```
$ freeview ./data/freesurfer/REGISTERED/20230213_073508.mgz
```

We choose the fat in the posterior part of the orbit (behind the eye), since it can be assumed that this tissue is not enriched by CSF tracer and does not undergo changes during the imaging period of about three days (Eide et al., 2021). First, we move the red, green, and blue planes in the 3D window in Freeview such that they all intersect with the fat tissue (see the 3D panel on the right of Fig. 5.2). For the MRI under consideration, locating the cursor at voxel indices (92, 128, 168) as in Figure 5.2 is a possible choice, since all the planes there cut through the posterior part of the orbit.

To start marking the reference ROI, click File→New Volume, keep the default options, and name the new volume *refroi*. After this empty volume is created, the MR image is overcast, and we change the color map of the volume refroi to ”Heat” or ”Jet” with ”Maximum” set to 1 such that the ROI can be displayed on top of the MR image. We next click Action→Voxel edit and set the ”Brush value” to 1. In every slice window, we mark a few bright, white voxels in the posterior part of the orbit; see Figure 5.2. Note that the ROI should not cover nerve fibers, visible in black close to the ROI and indicated by the blue arrows in Figure 5.2. Finally, the volume can be saved to `./data/freesurfer/mri/refroi.mgz` by clicking File→Save Volume.

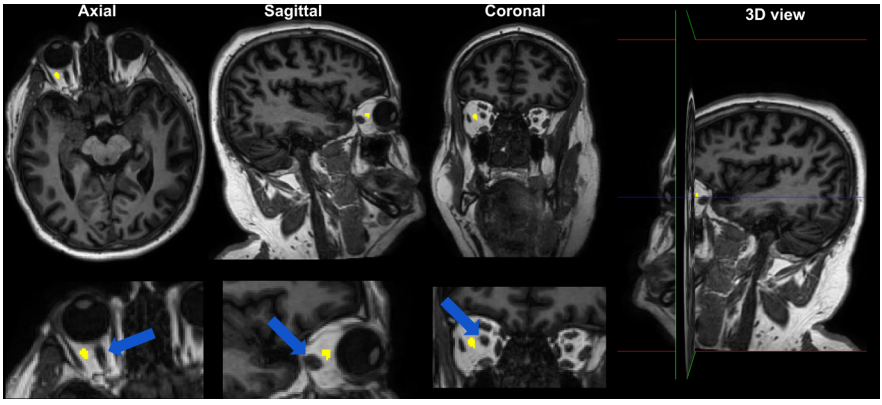


Fig. 5.2: This image shows the ROI in the posterior part of the orbit in yellow, together with the baseline image (visualized using Freeview). The blue arrows indicate the optical nerve, which we do not want to include in the ROI.

Normalization

Using the ROI created in the subsection above, we normalize every image with the median MR signal in this ROI as

$$S((i,j,k),t) = \frac{I((i,j,k),t)}{\text{median}_{(a,b,c)\in \texttt{ROI}}\{I((a,b,c),t)\}}, \tag{5.3}$$

where $I((i,j,k),t)$ denotes the T1-weighted image signal in voxel (i,j,k) at time t and $S((i,j,k),t)$ is the normalized T1-weighted signal. (5.3) is implemented in the Python script `./scripts/data-processing/normalize.py`, which can be used to automatically normalize all images at time t. Running this script also prints the median T1-weighted MR signal values for the baseline image and later images to the terminal. With the ROI stored under `./data/freesurfer/mri/refroi.mgz`, we can see that the median varies from 90, 157 at the baseline to 75, 767, 74, 486, 71, 435, and 69, 087 at the later imaging timepoints. These variations demonstrate the necessity of normalizing the images.

Creating Brain Masks and Synthetic T1 Maps

As a first approximation, we assume that the relaxation time T_1 is constant in the whole brain parenchyma. To define which image voxels are considered to belong to the brain, we use the FreeSurfer segmentation output file found under `./data/freesurfer/mri/aseg.mgz`. The segmented regions are shown in color, with the baseline T1-weighted image in the left of Figure 5.3. The integer voxel values in this image represent the assignment of voxels to 44 distinct brain regions for this subject. The correspondence between labels and anatomical regions can be found in the FreeSurfer documentation,[4] where we can see that the labels 4, 5, 14, 15, 24, 43, and 44 identify CSF compartments such as the ventricles. Since all other nonzero labels denote brain tissue, we can create a binary mask for the brain parenchyma by identifying which of the voxels are labeled with one of these numbers. We assign the value 0 to these voxels, and we set all other labels to 1 to create a binary mask for the parenchyma tissue. In the same way, we can similarly create a synthetic T1 map by assigning a surrogate value for T1 taken from the literature to the voxels not labeled as CSF. We implement this in the `Python` script `make_brainmask.py`. For usage instructions, see the README file under `scripts/data-processing` from Zapf et al. (2024a). Running this script stores a brain mask and a synthetic T1 map with constant value $T_1 = 1$ s throughout the brain to `./data/freesurfer/mri/`.

[4] See `https://surfer.nmr.mgh.harvard.edu/fswiki/FsTutorial/AnatomicalROI/FreeSurferColorLUT`.

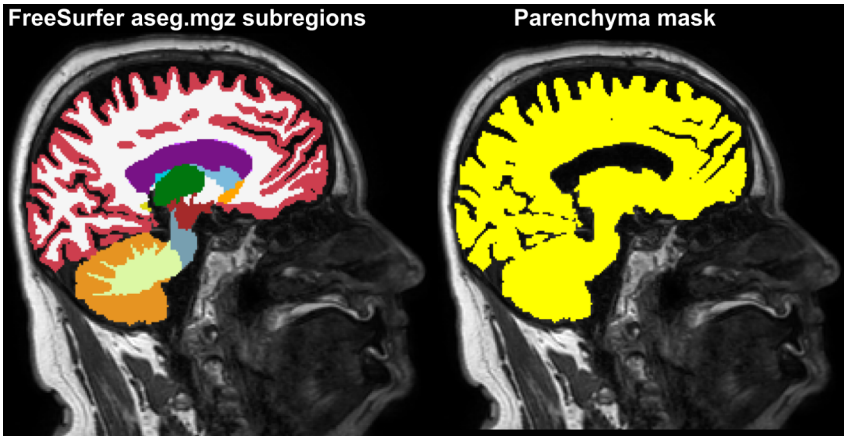


Fig. 5.3: Left: FreeSurfer segmentation (colored) of the brain. Right: The binary mask (yellow) created by excluding voxels labeled as CSF, such as the lateral ventricle (in purple in the left image).

Estimating T_1 from T1-weighted Images

With the synthetic baseline T1 map we have created in Section 5.4.2, we have everything needed to estimate the unknown relaxation time $T_1((i,j,k),t)$ at imaging times t after intrathecal tracer injection. Following Valnes et al. (2020), we use the relation

$$T_1((i,j,k),t) = f^{-1}\left(\frac{S((i,j,k),t)}{S((i,j,k),0)} f(T_1((i,j,k),0))\right), \tag{5.4}$$

where $S(i,0)$ denotes the normalized MR signal intensity before tracer injection and the function f is specific to the MRI acquisition protocol used. The T1-weighted MR images under consideration here were obtained using the Magnetization Prepared Rapid Acquisition Gradient Echo (also referred to as MPRAGE in the literature) sequence, for which we can use the approximation $f(T_1) = A\exp(-bT_1)$ for $T_1 \in [0.2\,\mathrm{s}, 4.5\,\mathrm{s}]$ with $A = 0.95$ *and* $b = 1.48\,\mathrm{s}^{-1}$.[5] With this approximation, we can simplify (5.4) to

$$T_1((i,j,k),t) = T_1((i,j,k),0) - \frac{1}{b}\log\frac{S((i,j,k),t)}{S((i,j,k),0)}. \tag{5.5}$$

Since we are primarily interested in computing the CSF tracer concentration, we do not compute and store the T_1 estimates separately but, rather, evaluate (5.5) as a step in our concentration computation in the next section.

[5] These numerical values for A, b are obtained by fitting $f(T)$ for $T \in [0.2\,\mathrm{s}, 4.5\,\mathrm{s}]$ to the supplementary Eq. (S11) of Valnes et al. (2020).

Computing the Concentration from T1-weighted Images

We combine (5.2) and (5.5) into a Python function that efficiently computes the concentration in all image voxels, using `numpy`. However, one must be careful because computing (5.5) can lead to unreasonably high or low estimates of T_1 due to noise in the T1-weighted MRI. As Valnes et al. (2020) does, we consider T_1 values outside the interval $[0.2, 4.5]$ s to be unreasonable and a consequence of noise, and we set higher/lower threshold values to the upper/lower interval bounds, respectively. The full script is found under `./scripts/data-processing/estimatec.py` and can be called as described in the accompanying README file under `scripts/data-processing` from Zapf et al. (2024a). Figure 5.4 compares the normalized MR signal to the CSF tracer concentration as computed with the constant T_1 value for three timepoints after injection.

5.5 MRI-informed Tracer Modeling

We now ask how well the brain-wide CSF tracer distribution we have computed in the previous section can be described by physics-based models in the form of partial differential equations. To this end, we revisit the model for molecular dynamics in the brain introduced by Mardal et al. (2022, Chapter 1), simplifying the model to include solute transport by diffusion only:

$$u_t - \operatorname{div} D \nabla u = 0 \qquad \text{in } (0,T] \times \Omega, \tag{5.6a}$$

$$u = u_d \qquad \text{on } (0,T] \times \partial\Omega, \tag{5.6b}$$

$$u(0,\cdot) = u_0 \qquad \text{in } \Omega, \tag{5.6c}$$

where u_0 denotes the initial condition, u_d the boundary conditions, and D is the diffusion tensor for the CSF tracer. If a baseline diffusion tensor image is available, it can be used to estimate the diffusion tensor of the tracer, as described by Valnes et al. (2020). Here, we will make a simplifying assumption and use a scalar diffusion coefficient $D_w = 10^{-3}\,\mathrm{mm^2 s^{-1}}$ for water in the brain, which corresponds to $D = 1.3 \times 10^{-4}\,\mathrm{mm^2 s^{-1}}$ for the CSF tracer gadobutrol (Valnes et al., 2020).

Solving these equations then requires specifying both u_0 and u_d. Mardal et al. (2022, Chapter 6) showed how to solve these equations with the modeling assumptions $u_0 = 0$ and $u_d = 1$, with the domain Ω being the full left hemisphere of the brain. Using the concentration data obtained in the previous section, we can now improve upon these assumptions and define the boundary condition as a linear interpolation between image acquisitions:

$$u_d(x,t) = c(x,t_i) + \frac{c(x,t_{i+1}) - c(x,t_i)}{t_{i+1} - t_i}(t - t_i) \quad \text{for} \quad t_i \le t \le t_{i+1}, \tag{5.7}$$

where $t_i = \{0, \ldots, T\}$ is the image acquisition times (relative to the first image after injection) and $c(x, t_i)$ is the CSF tracer concentration. Here, we consider a time span from $t = 0$ (image before injection) to the image taken at $T \approx 2$ days after injection. In this case, the initial condition is simply $u_0 = 0$ in Ω.

We solve Equation (5.6) numerically with the finite element method as described by Mardal et al. (2022, Chapter 3). This requires FEniCS to be installed (see the instructions given in the accompanying code repository Zapf et al. (2024a)).

We use the finite element mesh found under `./data/freesurfer/meshes/lh.xml`. This mesh has been generated as described by Mardal et al. (2022, Chapter 4).[6] Since we want to reuse the simulation code of Zapf et al. (2024c), we can benefit from adopting an object-oriented programming approach and wrapping the simulation code into a class. We call this class `Model`, and the numerical simulation of (5.6) can be computed by the class function `Model.forward()`.

The boundary condition (5.7) is implemented in the class method `Model.boundary_condition()`. Therein, the finite element functions describing the measured concentrations $c(x, t_{i+1}), c(x, t_i)$ are obtained by transformation between mesh coordinates (RAS space) and voxel indices (T1 space). How this is done in Python is briefly described in the script `./examples/mri2fenics.py`. The reverse operation of mapping arbitrary FEniCS functions defined on a brain mesh back to the MRI volume format is described under `/examples` by Zapf et al. (2024a).

The simulation can now be performed by executing the simulation script `./scripts/forward-model/diffusion.py`. By default, this script uses all images within the first three days after injection as the boundary condition to perform the simulations. After the simulation is done, the result can be inspected by running

```
$ paraview simulation_outputs/movie.pvd
```

The stored FEniCS functions $c(x, t_i)$ at imaging times i can be converted to the MRI format as described in `./scripts/forward-model/README.md` (Zapf et al., 2024a). A sagittal slice of the simulated concentration is shown together with the data in Figure 5.4. While the simulation underestimates the data at ~ 7 hours, the model overestimates the concentration significantly after ~ 26 hours. This suggests that diffusion as a sole transport mechanism is insufficient to explain the data, and more detailed models are investigated by Zapf et al. (2024c).

5.6 Concluding Remarks

In this chapter, we described a methodology to obtain quantitative maps of CSF tracer from qualitative MR images. This approach allows us to estimate that roughly one-

[6] For convenience, we provide the scripts used to generate this mesh with instructions under `./scripts/mesh-generation/`.

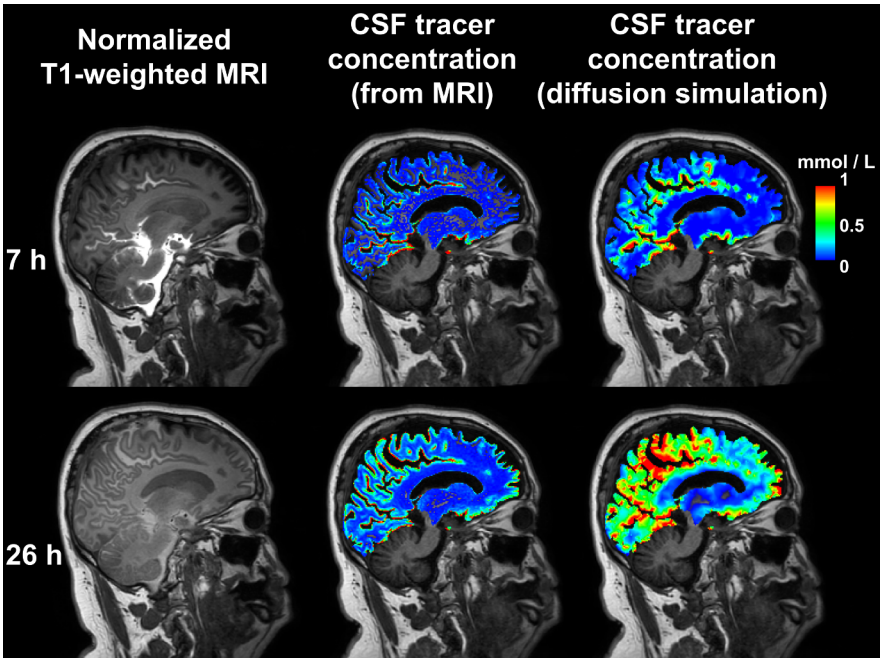


Fig. 5.4: The image shows the following for different timepoints after intrathecal injection of CSF tracer, from left to right: T1-weighted MRI, concentration of CSF tracer in the left hemisphere as computed with a synthetic T1 map, and concentration as simulated using the diffusion equation and subject-specific boundary conditions. This visualization was obtained using Freeview.

quarter of the injected tracer actually enters the brain (Vinje et al., 2023), opening up possibilities for quantitative, physics-based modeling as described herein. In this study, we manually defined the normalization region, which makes the preprocessing step subjective and time-consuming. This is due to the fact that the MR images we are working with are enriched with tracer, which makes the existing normalization tools nontrivial to employ in a consistent manner. Developing automated normalization algorithms—perhaps by combining tools such as FreeSurfer with data-driven machine learning approaches—is an interesting prospect for future research. Given the unique insights that CSF tracer-enriched MRI in humans can provide, more research groups are expected to adopt this technique in upcoming years. An automated algorithm to normalize and process the images acquired in these studies would significantly simplify data analysis and enhance the scientific insights obtainable from the images.

References

Eide PK, Vatnehol SAS, Emblem KE, Ringstad G (2018) Magnetic resonance imaging provides evidence of glymphatic drainage from human brain to cervical lymph nodes. Scientific Reports 8(1):1–10, doi:10.1038/s41598-018-25666-4

Eide PK, Vinje V, Pripp AH, Mardal KA, Ringstad G (2021) Sleep deprivation impairs molecular clearance from the human brain. Brain 144(3):863–874, doi:10.1093/brain/awaa443

Filippi CG, Watts R (2022) Imaging of Glymphatic Flow and Neurodegeneration, Springer, Cham, pp 849–860. doi:10.1007/978-3-030-82367-2_71

Fischl B (2012) FreeSurfer. NeuroImage 62(2):774–781, doi:10.1016/j.neuroimage.2012.01.021

Iliff JJ, et al. (2013) Brain-wide pathway for waste clearance captured by contrast-enhanced MRI. The Journal of clinical investigation 123(3):1299–1309, doi:10.1172/JCI67677

Mardal KA, Rognes M, Thompson T, Valnes L (2022) Mathematical Modeling of the Human Brain: From Magnetic Resonance Images to Finite Element Simulation. Springer, Cham, doi:10.1007/978-3-030-95136-8

Pooley RA (2005) Fundamental Physics of MR Imaging. RadioGraphics 25(4):1087–1099, doi:10.1148/rg.254055027

Reuter M, Rosas HD, Fischl B (2010) Highly accurate inverse consistent registration: A robust approach. NeuroImage 53(4):1181–1196, doi:10.1016/j.neuroimage.2010.07.020

Reuter M, Schmansky NJ, Rosas HD, Fischl B (2012) Within-subject template estimation for unbiased longitudinal image analysis. NeuroImage 61(4):1402–1418, doi:10.1016/j.neuroimage.2012.02.084

Ringstad G, Eide PK (2020) Cerebrospinal fluid tracer efflux to parasagittal dura in humans. Nature Communications 11(1):354, doi:10.1038/s41467-019-14195-x

Ringstad G, Vatnehol SAS, Eide PK (2017) Glymphatic MRI in idiopathic normal pressure hydrocephalus. Brain 140(10):2691–2705, doi:10.1093/brain/awx191

Rohrer M, Bauer H, Mintorovitch J, Requardt M, Weinmann HJ (2005) Comparison of Magnetic Properties of MRI Contrast Media Solutions at Different Magnetic Field Strengths. Investigative Radiology 40(11):715–724, doi:10.1097/01.rli.0000184756.66360.d3

Taylor AJ, Salerno M, Dharmakumar R, Jerosch-Herold M (2016) T1 Mapping: Basics Techniques and Clinical Applications. JACC: Cardiovascular Imaging 9(1):67–81, doi:10.1016/j.jcmg.2015.11.005

Valnes LM, Mitusch SK, Ringstad G, Eide PK, Funke SW, Mardal KA (2020) Apparent diffusion coefficient estimates based on 24 hours tracer movement support glymphatic transport in human cerebral cortex. Scientific Reports 10(1):1–12, doi:10.1038/s41598-020-66042-5

Vinje V, Zapf B, Ringstad G, Eide PK, Rognes ME, Mardal KA (2023) Human brain solute transport quantified by glymphatic MRI-informed biophysics during sleep and sleep deprivation. Fluids and Barriers of the CNS 20(1):62, doi:10.1186/s12987-023-00459-8

Watts R, Steinklein J, Waldman L, Zhou X, Filippi C (2019) Measuring Glymphatic Flow in Man Using Quantitative Contrast-Enhanced MRI. American Journal of Neuroradiology 40(4):648–651, doi:10.3174/ajnr.A5931

Yang L, et al. (2013) Evaluating glymphatic pathway function utilizing clinically relevant intrathecal infusion of CSF tracer. Journal of Translational Medicine 11(1):1–9, doi:10.1186/1479-5876-11-107

Zapf B, Dokken JS, Riseth J (2024a) csftracermodeling/csftracermodeling: Submitted code. doi:10.5281/zenodo.10807300

Zapf B, Valnes LM, Mardal KA, Zikatanov L (2024b) mri2fem-chp5-dataset: v1.0.0. doi:10.5281/zenodo.14629938

Zapf B, Zeinhofer M, Mardal KA (2024c) Inverse transport parameter estimation. In: Dokken JS, et al. (eds) Mathematical modeling of the human brain (vol II): from glymphatics to deep learning, Springer

Chapter 6

Signal increase ratio prediction with CNNs

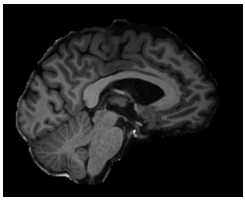
Marius Zeinhofer and Kent-Andre Mardal

Abstract Cerebrospinal fluid (CSF) clearance plays a critical role in the brain's waste removal system. In this chapter, we introduce a supervised deep learning pipeline designed to predict the signal increase ratio (SIR) in the CSF of the human brain 24 hours after the injection of an intrathecal contrast agent. The SIR quantifies the relative increase in magnetic resonance imaging (MRI) signal intensity compared to baseline measurements taken before the tracer injection, thereby providing a visualization of CSF dynamics over time. Our deep learning pipeline enables the prediction and visualization of these dynamics without the need for contrast agent administration, reducing the risk of side effects and lowering both costs and time requirements by relying solely on a baseline MRI scan. This chapter serves a dual purpose: it provides an introduction to deep learning–based image analysis and demonstrates the application of this technology using the nontrivial example of predicting the SIR in CSF.

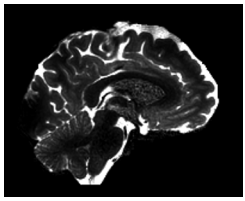
6.1 Introduction

Waste clearance in the human brain is a complex process, and malfunctioning clearance is associated with various pathologies such as Alzheimer's disease. While it is commonly accepted that the cerebrospinal fluid (CSF) plays a crucial role in waste clearance, the precise underlying principles are the topic of active research (Rasmussen et al., 2018). One way to investigate clearance processes in the CSF is to intrathecally inject a magnetic resonance (MR) imaging (MRI) contrast agent—for example, gadobutrol—and to subsequently perform several MR scans. One can use the voxel-wise signal increase ratio (SIR) to visualize tracer concentration over time to study the clearance process. However, this is time- and resource consuming and not free of side effects.

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_6



(a) Baseline.



(b) Tracer after 24 hours.

Fig. 6.1: This figure shows an exemplary training pair for the prediction of the SIR from a baseline image without tracer.

In this chapter, we therefore present a deep learning–based predictor that, given brain MR data without contrast-enhancing tracer, predicts the SIR 24 hours after injection. More precisely, we train a neural network on pairs of 136 T1-weighted human brain MR scans that all provide examples of a tracer-free scan as the input to the neural network and the SIR at 24 hours, computed using a second scan at 24 hours. We assume the MRI data are registered and anonymized, but all other preprocessing is described in detail in the following sections. This chapter also serves as a *brief* introduction to deep learning–based medical image analysis and is accompanied by a GitHub repository available from Zeinhofer (2024). A typical training pair can be seen in Figure 6.1.

6.2 Deep Learning Preliminaries

In this section we provide an introduction to convolutional neural networks (CNNs) and neural network training using stochastic gradient descent (SGD), since our SIR prediction described in the subsequent section relies on a voxel-wise prediction of a CNN.

6.2.1 CNNs

Generally, neural networks are parametric functions that, given a set of trainable parameters θ, can process an input X to produce an output Y, which we write as

$$Y = \mathcal{NN}_\theta(X).$$

In computer vision tasks, X is typically an image or volumetric data represented as a matrix or tensor[1], and the shape of Y depends on the task; in our case Y is

[1] In deep learning terminology, a tensor is a multidimensional array and thus generalizes scalars, vectors, and matrices.

itself an image that contains voxel-wise SIR values. The notation θ for the parameters is an abbreviation, and θ consists of a list of all the trainable weights of the network, typically a collection of weight matrices and bias vectors. The structure of neural networks is highly problem dependent, and specialized architectures have been developed for specific applications.

CNNs are a network architecture popular for computer vision tasks such as image classification and segmentation. The core idea of CNNs is to exploit invariance properties typically presented in natural images.

This can be explained more precisely by looking into how the convolution operation—the decisive building block of CNNs—works. Convolutions act on images and are *linear, local*, and *translation-invariant* operations. In the simplest case, given a gray-scale image represented as a matrix X, a convolution operates on X via

$$H_{ij} = \sum_{k,l=-\nu}^{\nu} W_{k,l} X_{i+k,j+l} + b,$$

where the learnable parameters are the scalar bias b and the matrix W, which is referred to as the convolutional kernel or filter. The number $\nu \in \mathbb{N}$ is the kernel width, and the matrix H is the result of the convolution. Note that the locality of the convolution is embodied by the choice of a small ν and the translation invariance by the fact that W does not depend on i, j. Convolutions are easily extended to three-dimensional data or colored images.

To transition from local information to image-wide context, several convolutions are daisy chained, with intermediate nonlinear up- and downsampling layers. Convolutions do not comprise the only building blocks of CNNs, but they convey the main intuition of the models: first, local, translation-invariant information is aggregated and subsequently combined to produce image-wide context. For a thorough introduction to CNNs, we refer the reader to Zhang et al. (2021).

For our image segmentation example, we use the U-net architecture that was originally proposed by Ronneberger et al. (2015) and, with minor adaptions, remains among the best architectures for image segmentation (see also Isensee et al. (2021)).

6.2.2 SGD

In the following, we illustrate neural network training using an abstract regression problem with a least squares formulation. Given a dataset $(X^{(i)}, Y^{(i)})_{i=1,\dots,N}$ of image pairs represented by matrices[2] of size (h, w) and a neural network $\mathcal{NN}_\theta$ that processes input to target shapes, we measure the neural network's predictions via the mean-squared error:

[2] One can use any other numerical data.

$$\mathrm{MSE}^{(i)}(\theta) = \frac{1}{h \cdot w} \sum_{k,l}^{h,w} \frac{1}{2} (\mathcal{NN}_\theta(X^{(i)})_{kl} - Y^{(i)}_{kl})^2.$$

To obtain accurate predictions on the whole dataset and unseen data, the objective of neural network training is to minimize the loss function

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} \mathrm{MSE}^{(i)}(\theta) \tag{6.1}$$

that averages the mean-squared error over all training samples in order to find well-performing parameters θ. The minimization of L is usually carried out using gradient-based optimization methods. Typically, naive gradient descent is intractable, since both the number N of samples is large and a single training example can be large in terms of memory requirements. One resorts to SGD, where, instead of utilizing L to perform a gradient descent, one uses randomly drawn subsets $\mathcal{B}$ of $\{1, \dots, N\}$, so-called *minibatches*. Precisely, given a current state of parameters θ_k, this means one uses

$$L_{\mathcal{B}}(\theta_k) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \mathrm{MSE}^{(i)}(\theta_k)$$

to compute the gradient $\nabla L(\theta_k)$ by $\nabla L_{\mathcal{B}}(\theta_k)$, which is utilized to update θ_k. A parameter update is of the form

$$\theta_{k+1} = \theta_k - \eta \nabla L_{\mathcal{B}}(\theta_k),$$

where $\eta > 0$ is the step size of the gradient update. In deep learning terminology the step size is called the *learning rate*. Partitioning $\{1, \dots, N\}$ in $\lfloor \frac{N}{|\mathcal{B}|} \rfloor$ random disjoint steps of size $|\mathcal{B}|$ and updating θ accordingly is referred to as a *training epoch*.

In practice, slightly more advanced optimization schemes are preferred over the vanilla SGD described above. They share a similar philosophy concerning stochasticity—maintaining the minibatch training strategy—but often incorporate additional intuition such as momentum and per-parameter learning rates. A precise description exceeds the scope of this chapter, and we refer the reader instead to Zhang et al. (2021). For our exemplary implementation we use the Adam algorithm (Kingma and Ba, 2014), which is widely used in deep learning, although more principled versions exist (e.g., Zaheer et al., 2018).

6.2.3 Train–test Split and Hyperparameter Tuning

When evaluating the trained model, it is essential to benchmark it on data that were not used in the training process in order to quantify how the model behaves on unseen data. Consequently, one splits the total amount of data into a *training set* and a *test set* and only uses the former in training. This separation of data is crucial, as can

be seen in Figure 6.2, where the loss on the training set is significantly smaller than on the test set. Note that after roughly 50 epochs only the training loss improves. This indicates that, at this point, the model is prone to *overfitting* the data; that is, it memorizes particularities of the training data as opposed to learning general rules.

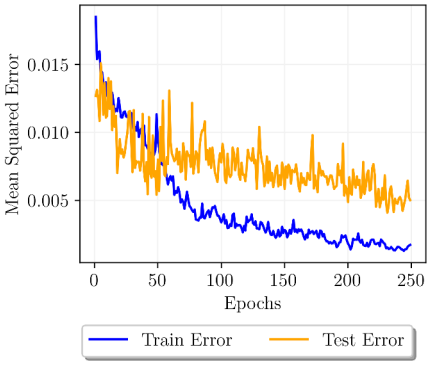


Fig. 6.2: Illustration of the training and test error during training. Depicted is the mean-squared error to the baseline over a training process of 250 epochs.

Typically, to achieve optimal results, it is not enough to perform a single model training process. Instead, the model's architecture, the optimizer, the minibatch size, the learning rate schedule, and so forth, all need to be fine-tuned. This process is referred to as *hyperparameter tuning*. It can be computationally expensive and requires experience.

6.3 Application: Tracer Prediction

In this section we discuss our concrete problem of employing a CNN to predict the SIR after 24 hours.

6.3.1 Data Preprocessing

Our starting point is a collection of T1-weighted MR brain images of patients before and 24 hours after intrathecal tracer injection. At this stage, we assume the images to be properly registered. Before setting up the machine learning pipeline, we perform the following preprocessing steps. Note that this preprocessing is not part of the provided repository, since it requires access to the full set of three-dimensional data, which cannot be disclosed for privacy reasons.

1) In a first step we normalize the MRI intensities by dividing through the value of orbital fat, behind the eye, as was done by Eide et al. (2021). This serves to mitigate variances in the data caused by the MRI machine.

2) Then, we extract a sagittal slice of the volumetric brain data, which we obtain at the 128th voxel of the first dimension of the data. This step mainly serves to reduce computational cost. From here on, we work with images instead of volumetric data. Note that, given sufficient computational resources, volumetric data can be handled in the same fashion as image data.

3) The quantity we aim to predict is the SIR, so, given the image data $X^{0\mathrm{h}} \in \mathbb{R}^{h\times w}$ before tracer injection and $X^{24\mathrm{h}} \in \mathbb{R}^{h\times w}$ 24 hours after tracer injection with height h and width w, we compute the SIR by the formula

$$\mathrm{SIR}_{ij} = \frac{X^{24\mathrm{h}}_{ij} - X^{0\mathrm{h}}_{ij}}{X^{0\mathrm{h}}_{ij} + \epsilon}.$$

We added $\epsilon = 0.1$ to the denominator to prevent division by zero. To reduce outliers in the data, we restrict the SIR to a ratio between zero and four. This choice is not necessarily physiologically grounded but leads to visually appealing results. As a formula, we have

$$\mathrm{SIR}_{ij} = \max(\min(\mathrm{SIR}_{ij}, 4), 0).$$

4) After the steps above, we convert all input data fed into the neural network to lie in the range $[0, 1]$. In this case, this requires converting gray-scale images $X^{0\mathrm{h}}$ that are typically represented as integers between zero and 255 to floating point values. This is carried out by a simple division by 255. Note that we will still denote the resulting converted image with values within $[0, 1]$ by the SIR.

Carefully preprocessing data is crucial for a successful machine learning pipeline, and care should be taken in handling and preprocessing the data. Data that are not properly preprocessed can lead to significantly worse model predictions. Furthermore, the concrete preprocessing steps in many cases form a part of the modeling assumptions, as in our case when we implicitly deem SIR increases beyond a value of four to be irrelevant and added $\epsilon = 0.1$ to the denominator.

6.3.2 Predicting SIR Formulated as a Regression Problem

The previously described data preprocessing pipeline provides us with data pairs

$$(X^{(i)}, Y^{(i)})_{i,\dots,N} = (X^{(i)}_{0\mathrm{h}}, \mathrm{SIR}^{(i)})_{i,\dots,N},$$

where N corresponds to the number of observations/patients available. Hence the task for our neural network $\mathcal{NN}_\theta$ is to predict the SIR given the brain data before tracer injection. For a given input $X_{0\text{h}}^{(i)}$, we write

$$\hat{Y}^{(i)} = \mathcal{NN}_\theta(X^{(i)})$$

for the estimated SIR prediction that our neural network produces. To train the neural network and to measure its performance, we train it using the mean-squared error, which is computed as

$$\text{MSE}^{(i)}(\theta) = \frac{1}{h \cdot w} \sum_{k,l}^{h,w} \frac{1}{2}((\text{SIR}^{(i)})_{k,l} - \mathcal{NN}_\theta(X^{(i)})_{k,l})^2.$$

The full data mismatch term measured over all data points, that is, the loss function, thus becomes

$$L(\theta) = \text{MSE}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \text{MSE}^{(i)}(\theta).$$

6.4 Implementation Details

There are several popular software libraries to choose from for deep learning applications. Among the most popular are TensorFlow (Abadi et al., 2016), PyTorch (Paszke et al., 2019), and JAX (Bradbury et al., 2018), and these libraries all provide efficient implementations of tensor operation and automatic differentiation. Further, the libraries enable computation on specialized hardware, such as GPUs and TPUs, and facilitate scaling to several such devices for large-scale applications.

In this chapter we use PyTorch 2.0.0, and we briefly comment on how one should structure the program using PyTorch. The description is intended to be high level. For the details, we encourage the reader to consult the corresponding repository.

Data Handling

We assume the preprocessed and ready-to-use data to be stored on disk in any user-defined logic. To allow PyTorch to access the data the recommended way, we subclass the PyTorch class `torch.utils.data.Dataset` and implement integer-based access to input–target pairs of the data. Precisely, we need to implement the methods `getitem` and `len`. Using further PyTorch functionality, `torch.utils.Dataloader`, we turn the dataset into an iterator suitable to perform minibatch training.

Neural Network Implementation

PyTorch provides all the building blocks commonly used for defining neural networks. Deep neural networks are typically thought to consist of blocks, which are a form of logical unit—in the easiest case, just a convolutional or fully connected layer. In PyTorch, these blocks are represented through the class `torch.nn.Module`, and any block or network architecture should subclass `torch.nn.Module`. In the constructor, one needs to register trainable parameters as member variables. This can happen either directly or by defining a member variable that is itself of type `torch.nn.Module`. Further, one needs to implement a method called `forward` that describes how input to the neural network is handled.

Training

As described earlier, stochastic minibatch training is the gold standard for neural network training. In the implementation this is reflected as a nested for loop where the inner loop constitutes one training epoch, and hence the training consists of one minibatch pass through the dataset and the outer loop repeating that for a given number of epochs. This training loop structure is omnipresent in deep learning, although some frameworks may hide it behind a single function call. The following code illustrates a typical training loop in PyTorch.

```
for epoch in range(epochs):
    for batch_idx, (inputs, targets) in enumerate(train_loader):
        inputs = inputs.to(device=DEVICE).unsqueeze(1)
        targets = targets.to(device=DEVICE)
        predictions = model(inputs).squeeze()
        loss = loss_fn(predictions, targets)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
```

Notably, the data are first loaded to the available accelerator `DEVICE`, which is either `'cuda'` or `'cpu'`. The data are propagated through the model, and `loss.backward()` invokes the automatic differentiation engine that computes the necessary gradients. It is important to invoke `optimizer.zero_grad()` before computing gradients, since PyTorch by default accumulates gradient computations.[3] In the above code `optimizer` is an instance of `torch.optim.Adam` and is the built-in version of the Adam optimizer in PyTorch. The function `loss_fn` is an implementation of the mean-squared error introduced earlier and is available in PyTorch via `torch.nn.MSELoss()`.

[3] This can be relevant to applications where the desired batch size does not fit in GPU memory.

Getting Started

To become acquainted with the implementation, we recommend starting with the training of the CNN on the exemplary dataset provided. This dataset consists of 30 images, a public subset of the full proprietary dataset. The small dataset has the advantage that it does not require a GPU for the training to finish in a reasonable amount of time. To execute the training, execute the following terminal command:

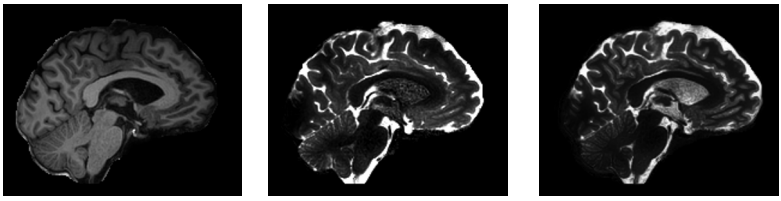
```
$ python train_mse.py --data_dir 'Dataset/' --out_folder \
    'out/'
```

where one substitutes the desired path for *'Dataset/* and *'out/'*. If one follows the structure of the provided repository, these arguments need not be specified, since they are set to working default values. One should then be able to see the value of the loss function as an output on the console while the training loops through the specified number of epochs.

6.5 Results

Figure 6.3 presents three pictures: the baseline MR image before tracer injection, the true measured tracer distribution after 24 hours, and the neural network prediction. Note that this image belongs to the *test set*, meaning that it was not used during training. For the figures in this section, we employed the full proprietary dataset consisting of 136 images, rather than a publicly available subset.

We conclude that even with the simple setup presented here, using only a small dataset, a neural network can already produce reasonable-looking predictions. However, we stress that this chapter has an introductory character and a more careful analysis is necessary before employing an analog pipeline in practice.



(a) Baseline. (b) Tracer after 24 hours. (c) Network prediction.

Fig. 6.3: This image shows a baseline image without tracer, the SIR after 24 hours, and the corresponding neural network prediction.

6.6 Conclusion

This chapter demonstrates the feasibility of using CNNs to predict CSF tracer dynamics in MRI, even when working with a limited dataset. The approach presented provides realistic predictions of the SIR and offers a method to visualize CSF dynamics without the need for tracer administration. In addition to outlining the pipeline's application, the chapter introduces key concepts related to CNNs, covering basic algorithmic components and architectural considerations. The methodology described here can serve as a reference for similar medical imaging tasks.

References

Abadi M, et al. (2016) TensorFlow: A System for Large-Scale Machine Learning. In: Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation, USENIX Association, OSDI'16, pp 265–283

Bradbury J, et al. (2018) JAX: composable transformations of Python+NumPy programs. URL `http://github.com/google/jax`

Eide PK, Vinje V, Pripp AH, Mardal KA, Ringstad G (2021) Sleep deprivation impairs molecular clearance from the human brain. Brain 144(3):863–874, doi:10.1093/brain/awaa443

Isensee F, Jaeger PF, Kohl SA, Petersen J, Maier-Hein KH (2021) nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. Nature Methods 18(2):203–211, doi:10.1038/s41592-020-01008-z

Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. Proceedings of the 3rd International Conference on Learning Representations doi:10.48550/arXiv.1412.6980

Paszke A, et al. (2019) Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems 32, doi:10.48550/arXiv.1912.01703

Rasmussen MK, Mestre H, Nedergaard M (2018) The glymphatic pathway in neurological disorders. The Lancet Neurology 17(11):1016–1024, doi:10.1016/S1474-4422(18)30318-1

Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, Springer, pp 234–241, doi:10.1007/978-3-319-24574-4_28

Zaheer M, Reddi S, Sachan D, Kale S, Kumar S (2018) Adaptive methods for nonconvex optimization. In: Bengio S, et al. (eds) Advances in Neural Information Processing Systems, Curran Associates, Inc., vol 31

Zeinhofer M (2024) Mariuszeinhofer/tracer_preds: Sir prediction with cnns. doi:10.5281/zenodo.10809187

Zhang A, Lipton ZC, Li M, Smola AJ (2021) Dive into deep learning. arXiv preprint arXiv:210611342 doi:10.48550/arXiv.2106.11342

Chapter 7

Estimating molecular transport parameters using inverse PDE models

Bastian Zapf, Marius Zeinhofer, and Kent-Andre Mardal

Abstract The introduction of the glymphatic system in 2012 unleashed a massive research interest in the fluid-mechanical aspects of the brain. Much of the research stems from medical imaging data of solute transport in alive brains in both animals and humans. Yet, the underlying physics of the fundamental processes are not fully understood. In this chapter, we utilize cerebrospinal fluid contrast-enhanced magnetic resonance (MR) imaging (MRI) data to test if a physical model including diffusion and reaction can explain how molecules are distributed brain-wide in humans. We apply two different methods, namely, physics-informed neural networks and an adjoint-based finite element method approach, to determine unknown model parameters from the MRI data. By incorporating executable code snippets, this chapter allows the reader to carry out and understand the implementation of these two approaches on their own machine using the associated publicly available MRI data.

7.1 Introduction

In Chapter 5 (Zapf et al., 2024b) we estimated the concentration of intrathecally injected cerebrospinal fluid (CSF) tracer in the brain from the so-called glymphatic magnetic resonance imaging (MRI) protocol (Iliff et al., 2013; Ringstad et al., 2017, 2018). The imaging involved several MR images taken over a time span of several days. These MRI studies indicated that the data were likely not explained by extracellular diffusion alone, but rather were governed by additional processes as proposed by the glymphatic concept (Iliff et al., 2012). In rodents, it has been observed that MRI contrast is transported 10–25 times faster than expected from extracellular diffusion alone (Ray et al., 2021), and 10–20 times faster in sleep than awake (Xie et al., 2013). This factor appears to be smaller in humans and has been estimated to range from a factor of up to 1.3 (Valnes et al., 2020) to 3.5 (Vinje et al., 2023), depending on the model and time frame of the MRI data. Both finite

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_7

elements and neural networks yield similar estimates (Zapf et al., 2022). Given that impaired clearance of molecules from the brain could play a central role in the development of neurological diseases (Rasmussen et al., 2018) and that the exact physical mechanisms have not yet been established (Bohr et al., 2022; Daversin-Catty et al., 2020; Holter et al., 2017; Sharp, 2023), image-based computational modeling of physiological processes is urgently needed.

However, assessing the relative significance and plausibility of different transport mechanisms via physics-based simulations is challenging, since the relevant physiological parameters are partially unknown in humans (Croci et al., 2019). As a remedy, one can instead attempt to estimate the unknown parameters from MRI data.

This chapter focuses on diffusion–clearance models, mathematically formulated as a diffusion–reaction equation. Diffusion–advection models are covered by Rognes (2024). The clearance is related to the multicompartment models of Riseth et al. (2023), in the sense that the MR contrast is cleared via either the perivascular or vascular routes. We note that the apparent diffusion in our model may include dispersion.

Our model for the diffusion and clearance of molecules is described by a partial differential equation (PDE) with unknown coefficients. We use two different variants of *PDE-constrained optimization* (Hinze et al., 2008) to determine them from MRI data. The first approach is based on the finite element method (FEM) via the simulations implemented by Zapf et al. (2024b, Chapter 5). As an alternative approach, we also demonstrate how to use the recently popularized physics-informed neural networks (PINNs) (Raissi et al., 2019) to solve this problem. We discuss how and why the two approaches apply different mathematical formulations of the physiological modeling problem. The respective mathematical formulations lead to different numerical problems that need to be solved, and we investigate the influence of numerical hyperparameters on the outcomes of the two methods.

7.2 Problem Formulation

We consider the CSF tracer concentration $c^d(t_i)$ in the brain estimated from MRI by Zapf et al. (2024b) at the discrete timepoints $t_i \in \{7, 26, 50\}$ hours and model these data using a function $c(t, x)$ that fulfills the diffusion–reaction equation

$$\frac{\partial}{\partial t} c(t,x) = D\Delta c(t,x) - rc(t,x) \quad \text{in } \Omega \times (0, 50\,\text{h}), \tag{7.1}$$

with unknown apparent diffusion coefficient $D > 0$ and a reaction rate $r > 0$. We note that, for simplicity, D and r are constants throughout the domain. The process of extracellular diffusion is well documented (Nicholson, 2001), and it is known that diffusion in the nanometer-thick extracellular matrix is slowed down by roughly a

factor of three compared to free diffusion. To determine these two unknown scalar numbers, we define the following PDE-constrained optimization problem. Find the parameters D, r such that

$$\mathcal{J}(c, c^d) = \sum_{i=1,2,3} \int_\Omega \left(c^d(t_i, x) - c_{D,r}(t_i, x) \right)^2 \tag{7.2}$$

is minimized under the constraint that $c_{D,r}(t, x)$ fulfills (7.1) for given parameters D, r. How this problem is solved with FEM, using an adjoint-based approach, is discussed in Section 7.4 and with PINNs in Section 7.5.

In principle, the domain Ω on which the optimization problem (7.1)–(7.2) is defined could be the whole brain parenchyma. However, the aim of this chapter is to present the basic workflow of the FEM and PINN approaches to estimating parameters from MRI in an easily reproducible fashion. Hence, we want to formulate the model such that the resulting numerical problems can be solved on personal computers, avoiding the need to run the computations on a supercomputer. To this end, our first criterion is to choose a subregion that is geometrically simple. For the FEM approach, a complicated domain will require a finer mesh resolution, increasing the memory and computing time requirements of the programs. Similarly, while PINNs are currently under rapid development (Cuomo et al., 2022), it appears that PINNs on complicated geometries currently require extensive tuning (Zapf et al., 2022) or domain decomposition strategies (e.g., D. Jagtap and Em Karniadakis, 2020), increasing the computational effort required to solve the problem (7.1)–(7.2).

Second, the choice of model limits the available choices of a region of interest (ROI). The parameters describing the dynamics of molecules in the brain vary among tissues and brain locations, as can be seen, for example, in diffusion tensor imaging for water molecules. Hence our modeling assumption of using scalar parameters D, r can be expected to be less accurate the larger the chosen ROI. An example of an ROI Ω fulfilling these two criteria is shown in Figure 7.1D–F.

Third, the ability to focus on small subregions of the brain can have relevance from a medical point of view, for example, to discern the underlying properties of damaged tissue. Choosing a small region, preferably containing a single type of tissue, increases the interpretability of the results. In Section 7.3 we describe how to define such a region for the MRI data accompanying this book, using Freeview and a Python script.

Finally, not all brain regions are enriched with tracer. This can—at least in part—be explained by the large variation in the dynamics of the contrast agent in the CSF spaces between subjects (Eide et al., 2021). The tracer enters the brain from CSF spaces, and, hence, if no tracer is available at the brain–CSF boundary, the region will not be enriched with tracer. Data from these regions hence contain no information about molecular transport in the brain. In exploratory studies we have used different ROIs to solve the parameter estimation problem described above. It was observed that choosing an ROI such that most of the region is enriched with

tracer during the observation period is crucial for the parameter estimation to be as robust as the results reported in this chapter. Choosing an ROI with minimal influx yields more unstable optimization problems and hence physiologically less valuable results. Hence, another criterion for an ROI for solving (7.1)–(7.2) is significant influx of tracer over the time span of interest.

7.3 Defining and Meshing an ROI

In this chapter, we assume that the data and scripts are arranged as described by Zapf et al. (2024b). We start by opening the concentration estimates and the baseline T1-weighted image in Freeview:

```
$  freeview ./data/freesurfer/REGISTERED/20230213_073508.mgz \
       --colormap Jet -v \
       ./data/freesurfer/CONCENTRATIONS/6.56.mgz \
       ./data/freesurfer/CONCENTRATIONS/26.05.mgz \
       ./data/freesurfer/CONCENTRATIONS/50.39.mgz
```

Setting the minimum and maximum of the color map to 0 and 0.2, respectively, is advised. This is beneficial for visualization of the data, since there are only a few outlier voxels with concentration values above 0.2 mmol/L. Setting the color map maximum to 0.2 hence yields a visualization that is not dominated by outliers.

By scrolling through the image along different axes, we can inspect the concentration data in different regions. By activating only one volume in the list of loaded volumes in Freeview at a time, we can switch the view between the three different timepoints. Moving the cursor to voxel indices (146, 100, 133) and displaying only the concentration at 26 hours, we obtain the views shown in Figure 7.1A–C. It can be seen that the brain is primarily enriched with tracer from the subarachnoid space, and hence our first criterion for a suitable ROI is fulfilled. However, these regions are characterized by the complex folding patterns of the cerebral cortex, and hence our second criterion for a good ROI is not fulfilled.

A closer look at the slices in Figure 7.1D–F reveals that the brain is also enriched with tracer from the lateral ventricles in some regions. In particular, the subcortical white matter around the position of our cursor at voxels (146, 100, 133) is clearly enriched with tracer at all timepoints 7, 26 and 50 hours after injection. Since the boundary between the tissue and the lateral ventricles is considerably simpler than the folds of the cerebral cortex, this region fulfills both our criteria for a suitable ROI.

We choose to define the ROI as the set of voxels indices $\{(a, b, c)\}$ that are contained in a sphere of radius $R = 10\,\text{mm}$ around the cursor at voxel indices (146, 100, 133) and are labeled as brain tissue in the FreeSurfer segmentation file `aseg.mgz`. Mathematically, these constraints can be formulated as

$$(a - 146)^2 + (b - 100)^2 + (c - 133)^2 \leq R^2 \quad \text{and} \quad (a, b, c) \in \Omega_{\text{brain}}. \tag{7.3}$$

We implement this condition in the script `./scripts/inverse-model/roi.py` found in the accompanying code repository (Zapf et al., 2024a). Executed with the instructions provided by Zapf et al. (2024a) and the binary mask for the brain parenchyma created by Zapf et al. (2024b, Chapter 5), this script creates an empty MRI volume and marks all voxels that fulfill (7.3) with the value 1 and the others with the value 0. From this binary mask, the script creates a surface and a volume mesh for this region with the resolution specified by the `--resolution` parameter as described by Mardal et al. (2022). The surface of this mesh is shown by white contours in Figure 7.1D–F. Running this script creates all these files and stores them in an output folder `./data/roi12/`. Therein, we find a finite element mesh for the ROI under `./data/roi12/parenchyma_mask_roi12.xml`, which is used for the FEM approach presented in this chapter. For convenience, the mesh is also stored in `.xdmf` format for visualization in ParaView. The PINN approach does not require a mesh, and our implementation uses the binary mask `./data/roi12/parenchyma_mask_roi.mgz`. This mask can be visualized in Freeview.

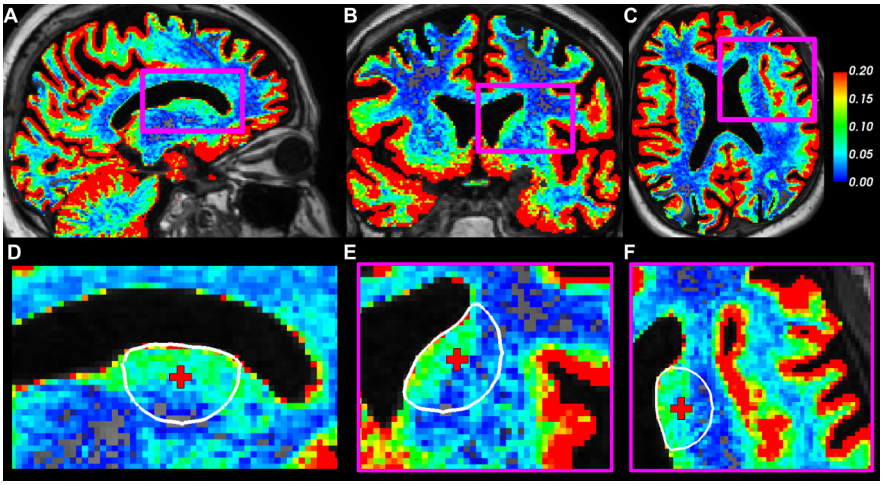


Fig. 7.1: This image shows slices through the baseline MR image (grayscale) and the tracer concentration in the brain after 26 hours (color scale). A–C: Whole brain. D–F: Zoomed-in region around the left lateral ventricle. The red cross indicates the location of the cursor at voxel indices (146, 100, 133), and the white contour lines depict the surface of the ROI on which we solve the inverse parameter estimation problem.

7.4 Finite Element Approach

We solve the parameter estimation problem (7.1)–(7.2) using a *reduced* approach with the FEM. That is, given Dirichlet boundary conditions[1] $g(t,x)$ and an initial condition $c_0(x)$, we can use the code from Zapf et al. (2024b) to solve the well-posed PDE problem

$$\frac{\partial}{\partial t}c(t,x) = D\Delta c(t,x) - rc(t,x) \quad \text{in } \Omega \times (0, 50\,\text{h}), \tag{7.4a}$$

$$c(t,x) = g(t,x) \quad \text{on} \quad \partial\Omega \times (0, 50\,\text{h}), \tag{7.4b}$$

$$c(0,x) = c_0(x) \tag{7.4c}$$

numerically to obtain a unique solution $c_{D,r}$. As in Section 5.5, the boundary condition g is taken as a linear interpolation in time between available MR images, and the initial condition is $c_0 = 0$, since the tracer is injected at a later timepoint $t > 0$.

The mismatch (7.2) between model and observation then becomes a function of D and r only and is called the *reduced functional*

$$\mathcal{J}_r(D,r) = \mathcal{J}(c_{D,r}, c^d). \tag{7.5}$$

Now we have reformulated the constrained optimization problem (7.1)–(7.2) into the unconstrained optimization of (7.5) with respect to D and r. This can be solved using gradient-based optimization algorithms. For the FEM approach, we use the quasi-second-order method L–BFGS–B (Fletcher, 2013).

Hence, an efficient way of computing the gradients

$$\frac{\partial \mathcal{J}_r(D,r)}{\partial D} \quad \text{and} \quad \frac{\partial \mathcal{J}_r(D,r)}{\partial r} \tag{7.6}$$

is needed. We use the adjoint method—known as reverse mode algorithmic differentiation in machine learning (Baydin et al., 2018)—as implemented for FEniCS(Alnæs et al., 2015) in dolfin-adjoint[2] (Mitusch et al., 2019). Under the hood, the dolfin-adjoint package achieves this task by overwriting the functions and objects defined in the Python interface to FEniCS. Our script to solve the optimization problem (7.5) can be found under `scripts/inverse-model/fem-inverse-diffusion.py`. Here, we outline the essential parts of this script, though the actual implementation might

[1] We work with Dirichlet boundary conditions here because they can be estimated from MRI data as described in Zapf et al. (2024b). In some applications, Neumann or Robin boundary conditions might be more appropriate. The methodology presented in this chapter can be equally applied when using these conditions instead.

[2] The interested reader is also referred to the dolfin-adjoint documentation at `http://www.dolfin-adjoint.org/en/latest/`. Various example implementations of PDE-constrained optimization problems and a more substantial introduction to the concepts behind PDE-constrained optimization can be found there.

differ slightly for technical reasons. The differences are explained in the comments accompanying the scripts.

After loading the mesh created in Section 7.3, we can define a finite element space `V` on this mesh and represent the concentration estimates as functions in this space. This process is wrapped into a class `FEniCS_Data`, and loading the MRI and storing the functions for visualizing can be done as follows:

```
mris = FEniCS_Data(function_space=V, datapath=datapath)
mris.dump_pvd(vtkpath=str(outfolder / "data.pvd"))
```

Now, given initial guesses for D and r, we can evaluate the forward simulation, and dolfin-adjoint keeps track of the intermediate results (needed to compute the gradients) as before; that is,

```
model = Model(V=V, mris=mris, outfolder=outfolder)
model.forward(D=D_w, r=r)
L2_mismatch = model.return_value()
model.save_predictions(name="initial")
```

The second-to-last line returns the data discrepancy (7.2) computed during the forward pass. In the last line, we store the simulated tracer fields at the timepoints where measurements are available for visualization. Note that the `forward` method is initialized with the diffusion coefficient D_w of water and, as such, the method is expected to iterate to determine the best estimate for the diffusion coefficient of gadobutrol. This implementation choice allows the flexibility to initialize the optimization with values from diffusion tensor imaging, which measures the diffusion of water.

Next, we define the set of optimization variables and the reduced functional (7.5) in dolfin-adjoint:

```
ctrls = [Control(D_w), Control(r)]
Jhat = ReducedFunctional(L2_mismatch, ctrls)
```

Finally, we optimize (7.5) for D, r using L–BFGS–B as

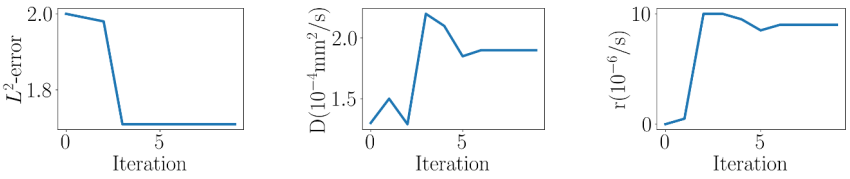
```
opt_ctrls = minimize(Jhat, method="L-BFGS-B", callback=iter_cb)
```

The so-called *callback* method `iter_cb` will be evaluated after every optimization step, and we use it to write the values of $\mathcal{J}, D$ and r during the optimization to a text file for later analysis.

The full optimization script can be run as described in the documentation README file under `scripts/inverse-model` of Zapf et al. (2024a). By default, the script uses a time step of one hour and the Crank–Nicolson time discretization scheme. With this setting, after nine iterations, which take about three minutes on a Lenovo ThinkPad P43s, the optimizer converges and the L^2 error is reduced by around

15% compared to our initial guess (see Fig. 7.2(A)). We find the optimal $D \approx 2 \times 10^{-4}$ mm^2/s and $r \approx 9 \times 10^{-6}$ 1/s (see Fig. 7.2(B)–(C)).

While these values are of the same magnitude as determined with PDE-constrained optimization using MRI from clinical studies (Valnes et al., 2020; Vinje et al., 2023), whether the numerical PDE solver has converged with respect to temporal and spatial discretization should be checked. We perform this important test in the next section.



(A) (B) (C)

Fig. 7.2: Behavior of the L^2 error, D, and r during optimization in the FEM approach. After nine iterations, one of the convergence criteria of the L-BFGS implementation used is reached and the optimization stops.

7.4.1 Convergence Analysis

In Section 7.3 we have created a mesh for Ω with the SVM-Tk mesh resolution parameter set to 12. Table 7.2 lists some characteristics of this mesh. We can see that this mesh consists of roughly 2×10^4 cells and has a maximum cell size (longest distance between any two cell vertices) of 1.73 mm. Given that the MRI resolution is 1 mm^3, we might ask if this resolution is fine enough. We therefore create two finer meshes with mesh resolution parameters set to 20 and 28, respectively, using the script presented in Section 7.3. Some characteristics of these meshes are tabulated in Table 7.1.

Table 7.1: Mesh characteristics for different resolution parameters

Mesh resolution parameter	12	20	28
Cells (10^4)	1.9	8.8	23.9
Vertices (10^4)	0.4	1.6	4.1
h_{min} (mm)	0.38	0.36	0.26
h_{max} (mm)	1.72	1.00	0.73

Next, we perform a systematic study and solve the inverse problem with all combinations of these three meshes and time steps of 60, 30, 15, and 7.5 min to assess if the

estimation of D, r has converged with respect to these numerical parameters. The shell script `scripts/inverse-model/run_fem_study.sh` automatically runs the simulations for all these parameters. Note that executing this script can take up to a day of computing time, since the computations are run sequentially. After the simulations have been run for all parameters, we can print the estimated coefficients to the terminal using the provided script `scripts/inverse-model/print_results.py`. The results are given in Table 7.2.

Table 7.2: Influence of mesh and time stepping size on the estimated parameters D, r and the relative error defined by (7.7).

Mesh resolution parameter	12	20	28	12	20	28	12	20	28
Time step dt (s)	D (10^{-4} mm^2/s)			r (10^{-6} 1/s)			Rel. L^2 error (10^{-2})		
3600	1.96	2.05	2.01	8.71	9.15	9.05	7.01	7.18	7.28
1800	1.97	2.07	2.02	8.79	9.25	9.14	7.89	8.06	8.16
900	1.73	1.81	1.77	7.40	7.76	7.67	7.51	7.71	7.80
450	1.63	1.70	1.66	6.83	7.14	7.07	7.30	7.51	7.59

Table 7.2 shows that the obtained diffusion coefficients vary by up to 24% from $D \sim 1.6 \times 10^{-4}\,\mathrm{mm}^2/\mathrm{s}$ to $D \sim 2.1 \times 10^{-4}\,\mathrm{mm}^2/\mathrm{s}$ between the different meshes and time step sizes. The variation in the reaction rate obtained is slightly larger, up to 27% between numerical parameters. We further compute the relative L^2 error defined as

$$\left(\frac{\sum_{i=1,2,3} \int_\Omega \left(c^d(t_i, x) - c_{D,r}(t_i, x) \right)^2}{\sum_{i=1,2,3} \int_\Omega \left(c^d(t_i, x) \right)^2} \right)^{1/2} \tag{7.7}$$

and present the numerical values in Table 7.2. While the values vary little between different numerical parameters, we observe in Section 7.5 that the FEM and PINN approaches yield different predictions. In Figure 7.4 we compare the FEM and PINN (described in the next section) reconstructions of the CSF tracer concentration at 7 hours and 50 hours after injection to the data.

7.5 PINN Approach

In this section we describe the PINN approach (Raissi et al., 2019) to determine the unknown diffusion and reaction term. Recall that we formulated the problem of recovering the diffusivity $D \in \mathbb{R}$ and the reaction coefficient $r \in \mathbb{R}$ from the MRI data as the minimization problem

$$\min_{c,D,r} J(c,D,r) = \sum_{i=1}^{3} \int_{\Omega} (c^d(t_i,x) - c(t_i,x))^2 \mathrm{d}x \tag{7.8}$$

subject to

$$\partial_t c(t,x) = D\Delta c(t,x) - rc(t,x) \quad \text{in } \Omega \times (0, 50\text{h}). \tag{7.9}$$

We keep the ROI and the data $c^d(t_i), t_i = 7, 26, 50$ h the same as in the FEM example.

7.5.1 Fully Connected Neural Networks

A fully connected feed-forward neural network $\mathcal{NN}$ of input dimension d with scalar-valued output and activation function $\sigma : \mathbb{R} \to \mathbb{R}$ is given by

$$\mathcal{NN} : \mathbb{R}^d \to \mathbb{R}, \quad \mathcal{NN}(x) = (T_N \circ \sigma \circ \cdots \circ T_2 \circ \sigma \circ T_1)(x), \tag{7.10}$$

where T_i is affine linear and hence given through a matrix W_i and a vector b_i as

$$T_i : \mathbb{R}^{n_{i-1}} \to \mathbb{R}^{n_i}, \quad T_i(x) = W_i x + b_i.$$

The activation function is applied component-wise. A network of the form (7.10) is said to be of depth N, and we refer to the affine maps T_i as the layers of the network. For notational convenience, we abbreviate the matrix–vector pairs as

$$\theta = ((W_1, b_1), \ldots, (W_N, b_N)).$$

The vector θ is referred to as the (trainable) parameters of the neural network. For a set of parameters θ, we denote the function represented via (7.10) by c_θ; that is , we have

$$c_\theta(x) = (T_N \circ \sigma \circ \cdots \circ T_2 \circ \sigma \circ T_1)(x) \quad \text{with } T_i z = W_i z + b_i.$$

An abundance of variations of neural network architectures are used in the literature, and the interested reader is referred to Zhang et al. (2021). For the applications in this section—and to some extent in the practice of PINNs—we keep the network sizes small, with only a few layers of moderate size, that is, with fewer than 100 neurons per layer. Such small networks are usually expressive enough to represent the data and benefit from reduced training costs compared to deeper neural networks. We choose the hyperbolic tangent tanh as the activation function in our application.

7.5.2 The PINN Formulation

The essential idea of PINNs is to approximate the solution c of the problem (7.8) by a neural network c_θ. To find parameters θ that lead to an accurate approximative solution of (7.8), one aggregates both the objective and the constraint into a loss

function L that one minimizes:

$$\begin{aligned}\min_{\theta,D,r} L(\theta) = \sum_{i=0}^{3} \int_{\Omega} (c^d(t_i,x) - c_\theta(t_i,x))^2 \mathrm{d}x \\ + w_{\mathrm{PDE}} \int_I \int_{\Omega} (\partial_t c_\theta - D\Delta c_\theta + r c_\theta)^2 \mathrm{d}x\mathrm{d}t,\end{aligned} \tag{7.11}$$

where w_{PDE} is a fixed, user-specified weighting term. The initial condition $c(0,x) = 0$ is weakly enforced via penalization of the first data loss term with index $i = 0$. Note that the first term in (7.11) vanishes if and only if the initial condition and data $(c^d(t_i))_{i=1,2,3}$ are matched, and the second term vanishes if and only if c_θ satisfies the PDE (7.9). Note the difference in the formulation of the minimization problem to the FEM approach:

(i) In the PINN approach, we rely only on the observational data and the interior PDE residual, whereas in the FEM we need to include boundary conditions. This may be seen as a benefit of the method, since specifying boundary conditions for the FEM approach requires one to make additional modeling assumptions (in our case, we interpolate the boundary condition linearly between observations). However, in cases in which one may have good boundary data or a plausible model for the boundary condition, including a boundary penalty into (7.11) adds several additional hyperparameters (magnitude of the weighting term, number of boundary points to use during minibatch training) to the PINN loss function. This issue has recently been addressed extensively in the PINN literature (Cuomo et al., 2022), but no common guidelines on how to choose them are available (Faroughi et al., 2024).

(ii) The FEM formulation makes it possible to solve the discretized PDE problem up to machine precision during the optimization process, whereas the PINN formulation includes the PDE as a soft penalty and thus does not guarantee an exact PDE solve. This motivates the weighting term w_{PDE} in (7.11).

In practice, we need to discretize the integrals appearing in (7.11). This is typically done employing Monte Carlo integration, that is, we sample random points $\{(t_j,x_j)\}_{i=1,\dots,N_{\mathrm{PDE}}} \subset I\times\Omega$ and subsample available data points $\{(t_j^d,x_j)\}_{i=1,\dots,N_{\mathrm{data}}}$, where $t_j^d \in \{1,2,3\}$ and x_j^d is a point in Ω where we have observational data. This results in the following discretized version of (7.11) that we still denote by L:

$$\begin{aligned}\min_{\theta,D,r} L(\theta) = \frac{1}{N_{\mathrm{data}}} \sum_{j=1}^{N_{\mathrm{data}}} (c^d(t_j^d,x_j^d) - c_\theta(t_j^d,x_j^d))^2 \\ + w_{\mathrm{PDE}} \frac{1}{N_{\mathrm{PDE}}} \sum_{j=1}^{N_{\mathrm{PDE}}} (\partial_t c_\theta(t_j,x_j) - D\Delta c_\theta(t_j,x_j) + r c_\theta(t_j,x_j))^2.\end{aligned} \tag{7.12}$$

For numerical stability, we restrict the ranges of D and r to certain intervals $[D_{\min}, D_{\max}]$, and $[r_{\min} r_{\max}]$ and parameterize

$$D = D(\delta) = D_{\min} + f(\delta)(D_{\max} - D_{\min}) \text{ and } r = r(\rho) = r_{\min} + f(\rho)(r_{\max} - r_{\min}),$$

where f is given by the sigmoid function $f(s) = (1 + e^{-s})^{-1}$. Hence, the optimization over D, r is implemented as an optimization over δ, ρ instead. In the application, we have to check if the optimization converges to the upper and lower bounds on the parameters. If so, the parameterization is too restrictive, and we increase the size of the interval between the minimal and maximal parameter values. We note that this parameterization approach also allows one to enforce the consistency of the parameters with the underlying physical model ($D, r > 0$ in our case).

7.5.3 Stochastic Gradient Descent

The prevailing strategy in the PINN community to solve (7.12) is to use stochastic first-order methods such as Adam (Kingma and Ba, 2014), although alternatives are currently actively being investigated (Müller and Zeinhofer, 2023; Siegel et al., 2023; Zeng et al., 2022). All methods require the computation of the gradients of L with respect to θ, D, and r. Modern deep learning frameworks provide efficient implementations of automatic differentiation and reduce the complexity of computer codes that compute the derivatives of L to a minimum. Here, we use the Python library JAX (Bradbury et al., 2018). In contrast to other deep learning frameworks, JAX implements `grad` as a function transformation. This means that it accepts a Python callable as a first argument and also returns a callable object, the gradient of the given function. The second argument, of integer type, denotes the variable with respect to which the gradient is taken.

Stochasticity is induced in the optimization process through variation of the quadrature points $\{(t_j^d, x_j^d)\}_{j=1,\dots,N_{\text{data}}}$ for the data and $\{(t_j, x_j)\}_{j=1,\dots,N_{\text{PDE}}}$ for the PDE residual. A common strategy is to use so-called minibatch training, where the total of the available data points is randomly partitioned in similarly sized minibatches and a gradient update is conducted for each minibatch before repeating the procedure. It is important to note that, while observational data $c^d(t, x)$ are typically only available at a finite number of space–time points (t, x), we can generate unlimited quadrature points for the PDE term, which allows more flexibility in handling this part of the loss function.

7.5.4 Hyperparameter Tuning

There are many design choices—commonly referred to as hyperparameters—to be made when employing PINNs in practice. Here, we present a non-exhaustive list,

discuss their relevance, and refer the reader to some of the literature. We present exemplary investigations concerning the sensitivity of the PINN estimates for D, r with respect to the weighting parameter w_{PDE}. Relevant hyperparameters include the following:

- The neural network's architecture, including both its fundamental structure, such as the choice between a Residual Network (ResNet) versus a fully connected architecture, the number and width of the layers, and the activation function. Hennigh et al. (2021) provides a summary of popular choices. For the results reported in this chapter, we use the hyperbolic tangent as an activation function.
- The weighting of the PDE and data loss terms. We demonstrate the effect of the weighting in Table 7.3 and refer the reader to Wang et al. (2021) for further details.
- Sampling strategies. This includes the size of the minibatch, the number of quadrature points employed for computing the PDE loss term, and possibly adaptive sampling strategies that redistribute quadrature points in regions of high residuals, as proposed by (Daw et al., 2022). In our implementation we use the evolutionary sampling proposed by (Daw et al., 2022).
- The choice of the L^p norm for the PDE loss in (7.11). Zapf et al. (2022) have documented that it can be advantageous to choose $p = 1$ instead of the natural choice $p = 2$.

It is important to note that hyperparameter tuning is a potentially difficult process. In practice, one should therefore carefully investigate the stability of the results with respect to the hyperparameters obtained with PINN (as with other methods).

Using the script `scripts/inverse-model/pinn-inverse-diffusion.py`, we train the PINN with three different choices of PDE weights w_{PDE}. By default, the script uses a PDE weight set to 10^3 and the Adam optimizer. The script can be called as described by (Zapf et al., 2024a) in the code documentation under `scripts/inverse-model`.

In Figure 7.3 we plot the loss, D, and r during the training of the PINN. First, we observe that significantly more iterations, compared to the FEM approach, are required for the determined parameters D, r to converge. This is explained by the fact that in the FEM approach we optimize using a quasi-Newton method, whereas in the PINN approach we use Adam, a first-order method. Further, the PINN optimization problem (7.12) is non-convex in terms of network parameters, making the PINN training numerically challenging. Second, from Figure 7.3 we observe that the optimization converges after 100,000 iterations in the present example. On a laptop and utilizing GPU acceleration, such training can take a few hours.

In Figure 7.4 we compare the PINN reconstruction of the data at 7 hours and 50 hours after injection to the data and the FEM reconstruction. We observe that the PINN and FEM approaches yield visually different reconstructions. This is possibly related to the difference in the problem setup; in the FEM approach the boundary

conditions are strictly enforced, whereas the PINN approach uses a weighted least squares minimization of both the PDE and the data.

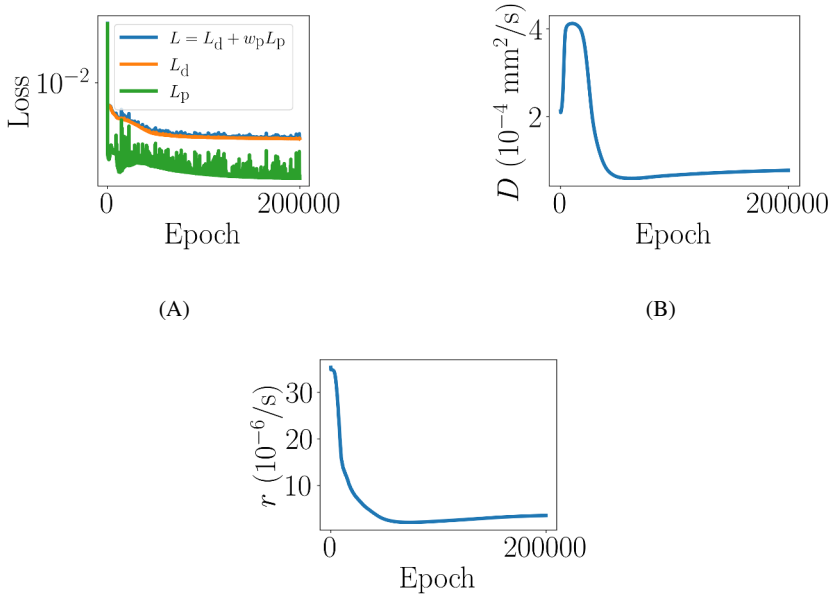


Fig. 7.3: Visualization of the different loss terms, D, and r during the training process in the PINN approach. Here we set the PDE weight to $w_p = 10^3$. Visual inspection suggests that the optimization has converged to a local minimum.

Next, we train the PINN with different PDE weights 10^2 and 10^4. The resulting parameter estimates are shown in Table 7.3. The diffusion coefficient and the reaction parameter vary by up to factors of 5.5 and 5.8, respectively, between the different PDE weights, indicating that the method has not yet converged with respect to the hyperparameters. We observe that for higher PDE weights, the parameter estimates are in closer correspondence to the FEM estimates. High PDE weights imply that the discretized PDE must be satisfied to higher accuracy, which is also the case in the FEM approach. Hence, high PDE weights can be expected to result in PINN predictions $c_\theta(t, x)$ closer to FEM solutions.

Table 7.3: The influence of the hyperparameter w_{PDE} on the outcome of the estimated transport parameters (D, r) and the relative L^2 error as defined by (7.7).

w_{PDE}	D $(10^{-4}\,\text{mm}^2/\text{s})$	r $(10^{-6}\,1/\text{s})$	Rel. L^2 error (10^{-2})
10^2	0.25	0.69	37.95
10^3	0.77	3.53	17.46
10^4	1.37	4.06	20.92

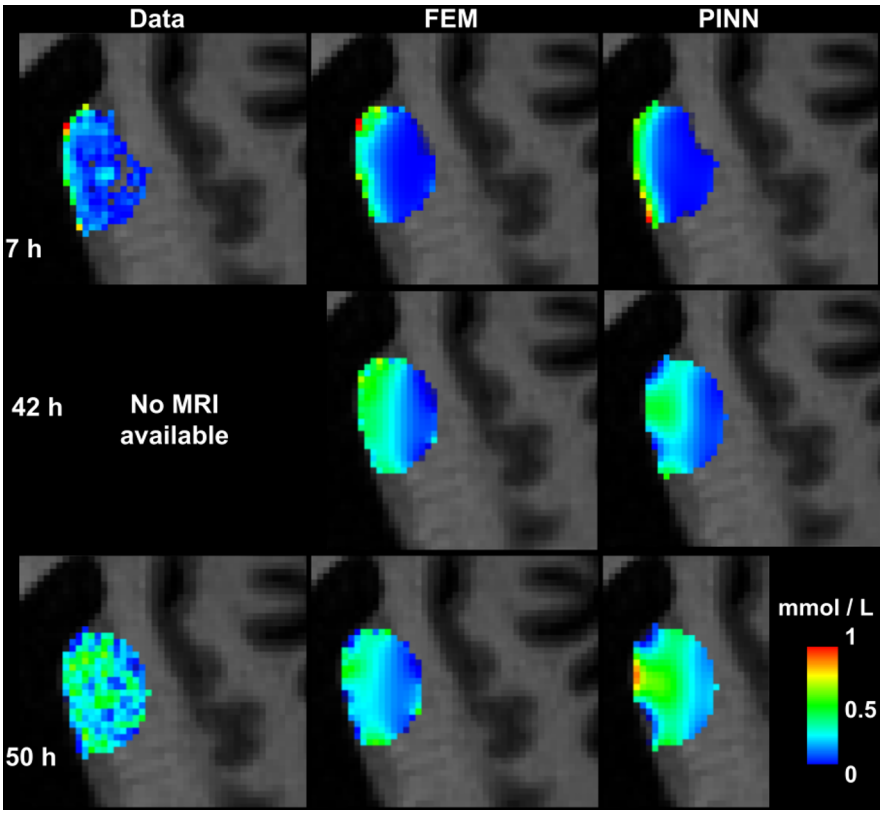


Fig. 7.4: This image shows slices through the baseline MR image (grayscale) and the tracer concentration in the brain after 7 hours and 50 hours (color scale) as estimated from MRI (left) compared to FEM reconstruction (center) and PINN reconstruction (right). The image also shows the prediction at 42 hours where no data are available.

7.6 Concluding Remarks

In this chapter, we have described two different computational methodologies to estimate unknown molecular transport parameters from MRI based on a simple diffusion–clearance model. We observe a certain mismatch between the values obtained for the diffusion coefficient when comparing the PINN estimates to the FEM estimates, which is not readily explainable. However, there are several differences between the methods. The PINNs approach is least squares based and, as such, does not require either the PDE or boundary conditions to be satisfied in the strict fashion enforced by the FEM method. In addition, the basis functions, optimization parameters, and so forth, vary among the methods. Furthermore, since the underlying model is a simple diffusion–clearance model, it might be that neither of the methods are able to provide good approximations and, rather, compensate in different manners to account for the additional/unknown physical processes or noise. We do note that the PINN errors are one magnitude greater than the FEM errors (see Tab. 7.2 and Tab. 7.3). We do not, however, believe that this is a general property of PINNs versus FEM but rather, that it is a consequence of the current setup.

Apart from the differences in problem formulation, another possible contribution to the differences already lies in the solution algorithm of the forward problem. While the discretized PDE problem in the FEM approach is a linear system that can be solved numerically with high accuracy, both forward and inverse PINN problems are nonlinear and difficult to solve with high precision in practice (Müller and Zeinhofer, 2023). Currently, however, algorithms to solve forward PINN problems with high precision have been developed (Müller and Zeinhofer, 2023; Siegel et al., 2023; Wang and Lai, 2024; Zeng et al., 2022). In future work, these algorithms could be adapted to inverse PINN problems.

References

Alnæs MS, et al. (2015) The fenics project version 1.5. Archive of Numerical Software 3, doi:10.11588/ans.2015.100.20553

Baydin AG, Pearlmutter BA, Radul AA, Siskind JM (2018) Automatic Differentiation in Machine Learning: a Survey. Journal of Marchine Learning Research 18:1–43, does not have doi

Bohr T, et al. (2022) The glymphatic system: Current understanding and modeling. Iscience 25:104987, doi:10.1016/j.isci.2022.104987

Bradbury J, et al. (2018) JAX: composable transformations of Python+NumPy programs. URL `http://github.com/google/jax`

Croci M, Vinje V, Rognes ME (2019) Uncertainty quantification of parenchymal tracer distribution using random diffusion and convective velocity fields. Fluids and Barriers of the CNS 16(1):1–21, doi:10.1186/s12987-019-0152-7

Cuomo S, Di Cola VS, Giampaolo F, Rozza G, Raissi M, Piccialli F (2022) Scientific machine learning through physics–informed neural networks: Where we are and what's next. Journal of Scientific Computing 92(3):88, doi:10.1007/s10915-022-01939-z

D Jagtap A, Em Karniadakis G (2020) Extended Physics-Informed Neural Networks (XPINNs): A Generalized Space-Time Domain Decomposition Based Deep Learning Framework for Nonlin-

ear Partial Differential Equations. Communications in Computational Physics 28(5):2002–2041, doi:10.4208/cicp.OA-2020-0164

Daversin-Catty C, Vinje V, Mardal KA, Rognes ME (2020) The mechanisms behind perivascular fluid flow. Plos one 15(12):e0244442, doi:10.1371/journal.pone.0244442

Daw A, Bu J, Wang S, Perdikaris P, Karpatne A (2022) Rethinking the importance of sampling in physics-informed neural networks. arXiv preprint arXiv:220702338 doi:10.48550/arXiv.2207.02338

Eide PK, Valnes LM, Lindstrøm EK, Mardal KA, Ringstad G (2021) Direction and magnitude of cerebrospinal fluid flow vary substantially across central nervous system diseases. Fluids and Barriers of the CNS 18(1):16, doi:10.1186/s12987-021-00251-6

Faroughi SA, Pawar NM, Fernandes C, Raissi M, Das S, Kalantari NK, Kourosh Mahjour S (2024) Physics-Guided, Physics-Informed, and Physics-Encoded Neural Networks and Operators in Scientific Computing: Fluid and Solid Mechanics. Journal of Computing and Information Science in Engineering 24(4):040802, doi:10.1115/1.4064449

Fletcher R (2013) Practical Methods of Optimization, 2nd edn. John Wiley & Sons

Hennigh O, Narasimhan S, Nabian MA, Subramaniam A, Tangsali K, Fang Z, Rietmann M, Byeon W, Choudhry S (2021) NVIDIA SimNet™: An AI-Accelerated Multi-Physics Simulation Framework. In: Paszynski M, Kranzlmüller D, Krzhizhanovskaya VV, Dongarra JJ, Sloot PM (eds) Computational Science – ICCS 2021, Springer, Cham, pp 447–461, doi:10.1007/978-3-030-77977-1_36

Hinze M, Pinnau R, Ulbrich M, Ulbrich S (2008) Optimization with PDE constraints, vol 23. Springer Science & Business Media, doi:10.1007/978-1-4020-8839-1

Holter KE, Kehlet B, Devor A, Sejnowski TJ, Dale AM, Omholt SW, Ottersen OP, Nagelhus EA, Mardal KA, Pettersen KH (2017) Interstitial solute transport in 3D reconstructed neuropil occurs by diffusion rather than bulk flow. Proceedings of the National Academy of Sciences 114(37):9894–9899, doi:10.1073/pnas.1706942114

Iliff JJ, et al. (2012) A Paravascular Pathway Facilitates CSF Flow Through the Brain Parenchyma and the Clearance of Interstitial Solutes, Including Amyloid β. Science Translational Medicine 4(147):147ra111–147ra111, doi:10.1126/scitranslmed.3003748

Iliff JJ, et al. (2013) Brain-wide pathway for waste clearance captured by contrast-enhanced MRI. The Journal of clinical investigation 123(3):1299–1309, doi:10.1172/JCI67677

Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. Proceedings of the 3rd International Conference on Learning Representations doi:10.48550/arXiv.1412.6980

Mardal KA, Rognes M, Thompson T, Valnes L (2022) Mathematical Modeling of the Human Brain: From Magnetic Resonance Images to Finite Element Simulation. Springer, Cham, doi:10.1007/978-3-030-95136-8

Mitusch SK, Funke SW, Dokken JS (2019) dolfin-adjoint 2018.1: automated adjoints for FEniCS and Firedrake. Journal of Open Source Software 4(38):1292, doi:10.21105/joss.01292

Müller J, Zeinhofer M (2023) Achieving high accuracy with PINNs via energy natural gradient descent. In: Krause A, Brunskill E, Cho K, Engelhardt B, Sabato S, Scarlett J (eds) Proceedings of the 40th International Conference on Machine Learning, PMLR, Proceedings of Machine Learning Research, vol 202, pp 25471–25485, URL `https://proceedings.mlr.press/v202/muller23b.html`

Nicholson C (2001) Diffusion and related transport mechanisms in brain tissue. Reports on progress in Physics 64(7):815, doi:10.1088/0034-4885/64/7/202

Raissi M, Perdikaris P, Karniadakis G (2019) Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics 378:686–707, doi:10.1016/j.jcp.2018.10.045

Rasmussen MK, Mestre H, Nedergaard M (2018) The glymphatic pathway in neurological disorders. The Lancet Neurology 17(11):1016–1024, doi:10.1016/S1474-4422(18)30318-1

Ray LA, et al. (2021) Quantitative analysis of macroscopic solute transport in the murine brain. Fluids and Barriers of the CNS 18(1):55, doi:10.1186/s12987-021-00290-z

Ringstad G, Vatnehol SAS, Eide PK (2017) Glymphatic MRI in idiopathic normal pressure hydrocephalus. Brain 140(10):2691–2705, doi:10.1093/brain/awx191

Ringstad G, Valnes LM, Dale AM, Pripp AH, Vatnehol SAS, Emblem KE, Mardal KA, Eide PK (2018) Brain-wide glymphatic enhancement and clearance in humans assessed with MRI. JCI Insight 3(13), doi:10.1172/jci.insight.121537

Riseth JN, Koch T, Mardal KA (2023) Two-compartment modeling of tracer transport in the brain. In: Dokken JS, et al. (eds) Mathematical modeling of the human brain (vol II): from glymphatics to deep learning, Springer

Rognes ME (2024) Estimating fluid flow fields from contrast imaging. In: Dokken JS, et al. (eds) Mathematical modeling of the human brain (vol II): from glymphatics to deep learning, Springer

Sharp MK (2023) Pulsatile cerebral paraarterial flow by peristalsis, pressure and directional resistance. Fluids and Barriers of the CNS 20(1):41, doi:10.1186/s12987-023-00445-0

Siegel JW, Hong Q, Jin X, Hao W, Xu J (2023) Greedy training algorithms for neural networks and applications to PDEs. Journal of Computational Physics 484:112084, doi:10.1016/j.jcp.2023.112084

Valnes LM, Mitusch SK, Ringstad G, Eide PK, Funke SW, Mardal KA (2020) Apparent diffusion coefficient estimates based on 24 hours tracer movement support glymphatic transport in human cerebral cortex. Scientific Reports 10(1):1–12, doi:10.1038/s41598-020-66042-5

Vinje V, Zapf B, Ringstad G, Eide PK, Rognes ME, Mardal KA (2023) Human brain solute transport quantified by glymphatic MRI-informed biophysics during sleep and sleep deprivation. Fluids and Barriers of the CNS 20(1):62, doi:10.1186/s12987-023-00459-8

Wang S, Teng Y, Perdikaris P (2021) Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM Journal on Scientific Computing 43(5):A3055–A3081, doi:10.1137/20M1318043

Wang Y, Lai CY (2024) Multi-stage neural networks: Function approximator of machine precision. Journal of Computational Physics 504:112865, doi:10.1016/j.jcp.2024.112865

Xie L, et al. (2013) Sleep Drives Metabolite Clearance from the Adult Brain. Science 342(6156):373–377, doi:10.1126/science.1241224

Zapf B, Haubner J, Kuchta M, Ringstad G, Eide PK, Mardal KA (2022) Investigating molecular transport in the human brain from MRI with physics-informed neural networks. Scientific Reports 12(1):15475, doi:10.1038/s41598-022-19157-w

Zapf B, Dokken JS, Riseth J (2024a) csftracermodeling/csftracermodeling: Submitted code. doi:10.5281/zenodo.10807300

Zapf B, Valnes LM, Mardal KA, Zikatanov L (2024b) Quantifying cerebrospinal fluid tracer concentration in the brain. In: Dokken JS, et al. (eds) Mathematical modeling of the human brain (vol II): from glymphatics to deep learning, Springer

Zeng Q, Bryngelson SH, Schäfer F (2022) Competitive physics informed networks. arXiv preprint arXiv:220411144 doi:10.48550/arXiv.2204.11144

Zhang A, Lipton ZC, Li M, Smola AJ (2021) Dive into deep learning. arXiv preprint arXiv:210611342 doi:10.48550/arXiv.2106.11342

Chapter 8

Two-compartment modeling of tracer transport in the brain

Jørgen N. Riseth, Timo Koch, and Kent-Andre Mardal

Abstract Understanding the mechanisms behind solute transport and waste clearance in the brain is crucial for understanding the pathophysiology of neurodegenerative disorders. Previous studies of brain-scale tracer transport based on glymphatic magnetic resonance imaging conclude that tracer influx and efflux rates from brain tissue are faster than predicted from a single-continuum diffusion process. This chapter describes a two-compartment diffusion–dispersion model for the transport of gadobutrol in the whole brain. The brain tissue is modeled as two interacting continua representing perivascular spaces (PVSs) and extracellular spaces (ECSs) in brain tissue, where transport in PVS is faster than in ECS. The model equations are solved using the finite element method to simulate tracer transport in the brain tissue following the intrathecal injection of gadobutrol. The simulation results show that, for the relevant parameter regime, an equilibrium is quickly attained between the ECSs and PVSs. Moreover, tracer dynamics are significantly influenced by perivascular diffusion/dispersion and solute transfer rates between PVSs and blood.

8.1 Introduction

The introduction of the theory of the glymphatic system (Iliff et al., 2012; Xie et al., 2013) has caused an upheaval in neuroscience and sleep research in the last decade. According to this theory (see also Rognes et al. (2024)), extravascular fluid dynamics and solute transport are fundamental mechanisms for brain waste clearance, and neurodegenerative disorders such as Alzheimer's and Parkinson's disease can be explained in terms of long-term waste buildup in brain tissue.

From a fluid dynamics perspective, several interesting observations have been made in the last decade. The extracellular volume fraction of the tissue available for fluid and solute transport increases during sleep by up to 60% (Xie et al., 2013, in mice). Perivascular spaces (PVSs) have been described as possible highways for

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_8

extravascular transport (Iliff et al., 2012; Jessen et al., 2015). Waste clearance could be enhanced by cardiac vessel pulsations (Mestre et al., 2018) and pulsations at lower frequencies due to respiration (Vinje et al., 2019) and sleep waves (Bojarskaite et al., 2023). These findings suggest that fluid–structure interaction between the vasculature and PVSs play an important role in the transport or mixing of solutes in the fluid-filled spaces of the brain (Bohr et al., 2022).

Data from glymphatic magnetic resonance imaging (MRI) (Ringstad et al., 2017, 2018) allow us to investigate the transport of tracers borne by cerebrospinal fluid (CSF) on the organ scale (millimeters to decimeters) in humans. In an attempt to explain the transport phenomena observed by MRI, several mathematical models have been proposed on the basis of a single-continuum tissue description and advection–diffusion–reaction models (Ray et al., 2021; Valnes et al., 2020; Vinje et al., 2023).

The main conclusion from these works is that solutes seem to enter and clear from brain tissue faster than extracellular diffusion alone can explain. However, with a single-continuum description of brain tissue, one cannot distinguish between potentially faster and slower pathways to identify the underlying mechanisms and driving forces enhancing transport.

In this chapter, we investigate a multicompartment diffusion–dispersion model for which a given brain tissue volume is formally split into several interacting continua.[1] The PVS constitutes a pathway allowing for fast transport. The extracellular space (ECS) predominantly features diffusive transport (Holter et al., 2017). Following the procedures outlined by Zapf et al. (2024) we set out to simulate gadobutrol transport in brain tissue and investigate whether the different compartments and their interaction can account for the previously reported necessity of non-diffusive mechanisms. Source code and instructions for running simulations and creating figures are openly available (Riseth, 2024).

8.2 Models and Methods

We consider a multicompartment description of brain tissue, as shown in Figure 8.1, comprising cells and fibers (s), blood vessels (b), PVS (p), and ECS (e) and model for each of the compartments $K = \{s, b, p, e\}$ the spatial and temporal evolution of the tracer concentrations $c_\alpha = c_\alpha(x, t),\ \alpha \in K$. The volume fraction occupied by each compartment is denoted by $n_\alpha,\ \alpha \in K$. We make the following simplifying assumptions:

[1] Multicompartment models can be mathematically derived via homogenization techniques (Arbogast and Lehr, 2006; Shipley and Chapman, 2010) under consideration of the tissue microstructure (Fig. 8.1) and have been applied to the brain to investigate, for example, oxygen transport (Payne and Mai, 2023), Alzheimer's disease progression (Vardakis et al., 2021), transport in tumors (Baxter and Jain, 1989; Ehlers and Wagner, 2013; Shipley and Chapman, 2010), and waste clearance (Poulain et al., 2023).

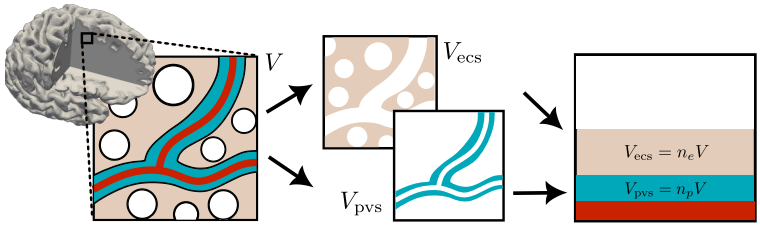


Fig. 8.1: Multicompartment description of brain tissue. The microstructure comprises cells and fibers (white), blood vessels (red), PVS (blue), and ECS. The different compartments are formally separated from each other—here only the main compartments of interest are shown, the ECS and PVS—and then described as overlapping continua occupying a certain volume fraction of the given tissue sample. The tissue microstructure only enters the model in the form of the model parameters in an average sense. Transport is described within and between the different compartments.

A.1 Tracer does not enter the cellular compartment ($c_s = 0$).

A.2 The clearance of tracer that entered the blood is faster than the time scale of interest ($c_b = 0$).

A.3 The tissue is rigid at the relevant time scale, while fast pulsatile vessel deformations may lead to enhanced mixing in the PVS compartment, modeled as dispersion.

A.4 Advective transport is negligible in the ECS and PVS.

A.5 There is no direct pathway from ECS to blood (tracer must pass through the PVS compartment).

The exchange between the PVS and the ECS compartment is modeled as diffusive transport through gaps between astrocyte endfeet, which form a continuous sheath around the PVS (see Koch et al. (2023) and references therein). PVS and ECS are assumed to be filled by water-like interstitial fluid with tracer concentrations c_p and c_e, respectively.

As a result of **A.1** and **A.2**, we only need to balance the amount of tracer in the ECS and PVS compartments, while transfer to the blood appears as a sink term for the PVS compartment balance. Therefore, the resulting simplified model is called a *two*-compartment model.

8.2.1 Two-compartment ECS–PVS Diffusion–dispersion Model

The spatial evolution of the fluid concentrations $c_\alpha(x,t)$ in the model domain Ω (human brain) over the time interval $[0,T]$ follows the governing equations (referred to as the two-compartment model):

$$\frac{\partial(n_p c_p)}{\partial t} - \operatorname{div}(n_p D_p^{\text{eff}} \nabla c_p) = t_{ep}(c_e - c_p) - t_{pb} c_p, \quad \text{in } \Omega \times (0,T], \tag{8.1a}$$

$$\frac{\partial(n_e c_e)}{\partial t} - \operatorname{div}(n_e D_e^{\text{eff}} \nabla c_e) = -t_{ep}(c_e - c_p), \quad \text{in } \Omega \times (0,T], \tag{8.1b}$$

subject to suitable boundary and initial conditions,

$$-n_p D_p^{\text{eff}} \nabla c_p \cdot \boldsymbol{n} = k_p(c_p - \hat{c}), \quad \text{on } \partial\Omega \times (0,T], \tag{8.1c}$$

$$-n_e D_e^{\text{eff}} \nabla c_e \cdot \boldsymbol{n} = k_e(c_e - \hat{c}), \quad \text{on } \partial\Omega \times (0,T], \tag{8.1d}$$

$$c_p(x,0) = c_{p,0}, \quad c_e(x,0) = c_{e,0}, \quad \text{in } \Omega \times \{0\}, \tag{8.1e}$$

where D_α^{eff} is the effective diffusion coefficient (in compartment α) modeling diffusion and dispersion, n_α is the corresponding volume fraction, and t_{ep} and t_{pb} are the diffusive transfer coefficients of the astrocyte endfeet sheath and the vessel wall, respectively.

The terms on the left-hand side of (8.1a) and (8.1b) describe a volume fraction–adjusted diffusion process for each of the compartments, whereas the right-hand sides represents solute transferred to and from the other compartments. We emphasize that the quantities in the equations should be interpreted as homogenized, volume-averaged quantities that—although related to a specific compartment—are defined everywhere in the domain. In other words, each compartment can be thought of as occupying a fraction of every point in the domain.

For the boundary conditions, $\boldsymbol{n}$ is the outward-oriented unit normal vector on the domain boundary $\partial\Omega$, and k_α is the conductivity of a surface membrane. We note that the limit $k_\alpha \to 0$ corresponds to a non-permeable membrane, and the limit $k_\alpha \to \infty$ models the absence of a membrane. In all cases, $\hat{c}$ is the tracer concentration in the CSF surrounding the brain tissue in the subarachnoid space (SAS), the ventricles, and the spinal canal.

8.2.2 Relation to the Single-compartment Diffusion Model.

For the interpretation of our results from the two-compartment model, we establish a relation to single-compartment models of previous works (Ray and Heys, 2019; Valnes et al., 2020; Vinje et al., 2023). We make one of two simplifying assumptions:

SC1 The ECS–PVS diffusive transfer coefficient, t_{ep}, is sufficiently large, leading to local chemical equilibrium, $c_e \approx c_p$,

SC2 The PVS volume fraction is negligible, $n_p \approx 0$, or, equivalently, $D_p^{\text{eff}} \approx D_e^{\text{eff}}$.

After summing the balance equations for each compartment, both **SC1** and **SC2** lead to a single-compartment model of the form

$$\frac{\partial(\phi c_{\text{F}})}{\partial t} - \operatorname{div}\left(\phi D^{\text{eff}} \nabla c_{\text{F}}\right) + t_{pb} c_{\text{F}} = 0 \qquad \text{in } \Omega \times (0,T], \tag{8.2a}$$

$$-\phi D^{\text{eff}} \nabla c_{\text{F}} \cdot \boldsymbol{n} = k(c_{\text{F}} - \hat{c}) \qquad \text{on } \partial\Omega \times (0,T], \tag{8.2b}$$

$$c_{\text{F}}(x,0) = c_{\text{F},0} \qquad \text{in } \Omega \times \{0\}, \tag{8.2c}$$

where $\phi := n_p + n_e$ is the combined volume fraction of ECS and PVS, $c_{\text{F}} = c_e = c_p$ denotes the fluid concentration, $\phi D^{\text{eff}} := \sum_{\alpha \in K} n_\alpha D_\alpha^{\text{eff}} = n_p D_p^{\text{eff}} + n_e D_e^{\text{eff}}$, and $k = k_p + k_e$. In the case of spatially constant volume fractions, the single-compartment model can also be written in terms of the total concentration $c_T := \sum_{\alpha \in K} n_\alpha c_\alpha = \phi c_{\text{F}}$ (Vinje et al., 2023). We note that **SC1** leads to $\phi D^{\text{eff}} = n_p D_p^{\text{eff}} + n_e D_e^{\text{eff}}$, whereas **SC2** leads to $D^{\text{eff}} = D_e^{\text{eff}}$, that is, **SC2** precludes an effect of PVS-enhanced transport.

8.2.3 Model Parameters

Table 8.1 summarizes estimates for two-compartment model parameter values together with a nominal value defining a reference scenario as the basis for the exploration of the input parameter space. In the following paragraphs, the chosen parameter ranges are motivated by modeling assumptions and data available in the literature. A schematic representation of the micro-scale configuration for the motivation of the intercompartmental transfer coefficients and the surface conductivity coefficients is shown in Figure 8.2.

8.2.4 Extracellular and Perivascular Volume Fractions

The extracellular volume fraction, n_e, is estimated to be in the range of 15–30% in normal-appearing adult brain tissue, with a typical value of 20% (Syková and Nicholson, 2008) that varies throughout the day and in disease. For instance, differences of 23% during sleep and 14% when awake have been measured in adult mouse cortical brain tissue (Xie et al., 2013).

The cerebral PVS volume fraction, n_p, is difficult to quantify in vivo, and it is unclear today whether fluid-filled spaces of significant size exist around vessels other than the largest vessels. The PVS volume fraction was estimated with MRI in white matter

Table 8.1: Model parameter values for the two-compartment diffusion model. The nominal values (NV) define a reference scenario. The range column indicates the defined, investigated ranges (see Sec. 8.3), with GM denoting gray matter, WM denoting white matter, DTI denoting diffusion tensor imaging, and BBB denoting the blood–brain barrier.

Symbol	Description	Unit	Varied range	NV	Comments/literature range
n_e	ECS volume fraction	-	fixed	0.20	0.14–0.30 (Syková and Nicholson, 2008; Xie et al., 2013)
n_p	PVS volume fraction	-	0.01–0.04	0.02	0.01–0.04 (see Sec. 8.2)
D_e^{eff}	Effective diffusion coefficient of ECS	$10^{-4}\text{mm}^2/\text{s}$	fixed	1.3	1.4 ± 0.4 (GM), 1.1 ± 0.3 (WM) (Valnes et al., 2020, DTI), 1.3 to 1.9 (Syková and Nicholson, 2008)).
D_p^{eff}	Effective diffusion coefficient in PVS	$10^{-4}\text{mm}^2/\text{s}$	1.3–130	3.9	Lower bound corresponds to the single-compartment assumption (**SC2**)
t_{ep}	Diffusive transfer coeff. ECS–PVS	10^{-2}s^{-1}	0.030–3.1	2.9	0.17–3.1 (Koch et al., 2023).
t_{pb}	Diffusive transfer coeff. PVS–blood	10^{-5}s^{-1}	0–2.3	0.2	2.3 (Vinje et al., 2023) (upper bound); 0 means the BBB is impermeable for gadobutrol
k_e	Pial surface conductivity	$10^{-5}\text{mm}\,\text{s}^{-1}$	0.026–2600	1.0	1.9 (Koch et al., 2023) (arteriole endfoot sheath, upper bound)
k_p	PVS Robin coeff.	$10^{-4}\text{mm}\,\text{s}^{-1}$	0.9–7.4	3.7	0.9–7.4 using 8.7, L = 20–40 µm (Brøchner et al., 2015), n_p (0.01-0.04).

at 1.14 ± 0.43%, with a range of 0.3–3.1% (Barisano et al., 2021), including PVS large enough to be visible in MRI.

Alternatively, n_p may be estimated from the cerebral blood volume fraction, n_b, assuming that the cross-sectional area of the lumen is proportional to the cross-sectional area of the PVS surrounding the vessel. Denoting by ψ the ratio of PVS area to vessel area, we have $n_p \approx \psi n_b$. Based on literature values for n_b[2] and the

[2] The *cerebral blood volume fraction* is consistently determined to be around 3–6% in gray matter and 1–3% in white matter using positron emission tomography (PET) or MRI (Ito et al., 2001; Leenders et al., 1990; Muizelaar et al., 1997). The arteries contribute about 20–30% (Hua et al., 2019; Ito et al., 2001; Muizelaar et al., 1997) of the blood volume, whereas the rest is contained in capillaries and veins. Methods based on PET and MRI cannot resolve the vessel microstructure and require additional assumptions for hematocrit and correction for flow artifacts (Muizelaar et al., 1997). Lauwers et al. (2008) determined the blood volume fraction in gray matter from the 3D vessel reconstruction of stained brain slices (Duvernoy et al., 1981) to be 2.7%, a higher value than the 1.5% reported for mice cortical microstructural data (Blinder et al., 2013). For a comparison between species, we refer the reader to Schmid et al. (2019).

available data on pial vessel PVS spaces,[3] we estimate $n_p = 1\%$ ($\psi = 0.3$ for all vessels with $n_b = 0.03$) as a lower bound and $n_p = 4\%$ ($\psi = 0.3$ for veins and capillaries vessels and $\psi = 1.7$ for arteries contributing 30% of $n_b = 0.06$) as an upper bound.

8.2.5 Effective Diffusion and Dispersion Coefficients

Transport in the ECS is assumed to occur predominantly through diffusion (Holter et al., 2017) due to low Péclet numbers. Diffusion in porous media is hindered by the tortuosity of the pore space and we model the effective diffusion coefficient following Syková and Nicholson (2008):

$$D_e^{\text{eff}} = \lambda_e^{-2} D^*, \tag{8.3}$$

where $\lambda_e = \sqrt{D^*/D_e^{\text{eff}}}$ denotes the tortuosity and D^* is the binary diffusion coefficient of gadobutrol in interstitial fluid. Using the Stokes–Einstein equation, which holds for the diffusion of spherical particles in a liquid with a low Reynolds number, we can estimate the binary diffusion coefficient of gadobutrol in water at 37 °C:

$$D^\star = \frac{k_B T}{6\pi\mu R} = 3.7 \times 10^{-4}\,\text{mm}^2/\text{s} \tag{8.4}$$

where $k_B = 1.38 \times 10^{-23}\,\text{m}^2\text{kg/s}^2/\text{K}$, $T = 310.15\,\text{K}$ (= 37 °C), and $R = 0.9\,\text{nm}$ for gadobutrol (Guthausen et al., 2015), and μ is the dynamic viscosity of the fluid (for water at 37 °C, $\mu = 0.69\,\text{mPa s}$). The tortuosity of brain tissue has been estimated to be in the range of 1.4–1.7 (Syková and Nicholson, 2008), using real-time iontophoresis. This corresponds to a range for D_e^{eff} of 1.3–1.9 mm^2/s. These values match well with estimates with diffusion-weighted MRI sequences. For instance, Valnes et al. (2020) estimated D_e^{eff} to be 1.4 ± 0.4 in gray matter and 1.1 ± 0.3 in white matter, using diffusion tensor imaging.

Transport in the PVS is assumed to occur through both diffusion and dispersion caused by oscillatory velocity fields due to vasomotion (Asgari et al., 2016; Bojarskaite et al., 2023; Keith Sharp et al., 2019):

$$D_p^{\text{eff}} = \lambda_p^{-2} D^* + D^{\text{disp}}, \tag{8.5}$$

[3] The *ratio of the PVS area to the vessel lumen area* in pial vessels has been estimated in mice to be 1.3–1.4 (Mestre et al., 2018; Schain et al., 2017) in pial arteries, 0.3 in veins (Schain et al., 2017), and 0.07–0.3 in capillaries (Tithof et al., 2022), based on a small number of samples. Broad distributions of the ratio were reported in mouse pial arteries, with values ranging from 0.3 to 6, and it was concluded that the vessel lumen area is mildly correlated with the PVS area (with a ratio of 1.66 and $R^2 = 0.2$) (Raicevic et al., 2023). However, since pial vessels are situated within the SAS, they are not part of our simulation domain. Moreover, functional tissue is tightly packed, with cells leaving little to no room for fluid-filled PVSs in comparison with the SAS.

where D_p^{eff} has been estimated as up to $2D^\star$ in simplified geometries.

In the absence of data, we estimate $\lambda_e \geq \lambda_p \geq 1$.

When estimating the parameters λ_p and D^{disp} in comparison with clinical data, they are structurally unidentifiable, since both appear only in the expression for D_p^{eff}. Therefore, for our parameter variation study, we investigate instead the effect of the ratio $\gamma := D_p^{\text{eff}}/D_e^{\text{eff}}$. In case **SC1** (or **SC2**) holds, we obtain an effective diffusion coefficient in terms of γ of $D^{\text{eff}} = \frac{(n_p\gamma+n_e)}{n_p+n_e}\lambda_e^{-2}D^*$ for the single-compartment model, Equation (8.2). For our parameter study, we choose a lower bound for γ of 1 (corresponding to **SC2** and $D^{\text{eff}} = D_e^{\text{eff}}$) and an upper bound of 100 (corresponding to a single-compartment coefficient of $D^{\text{eff}} \approx 5D^*$ if **SC1** holds).

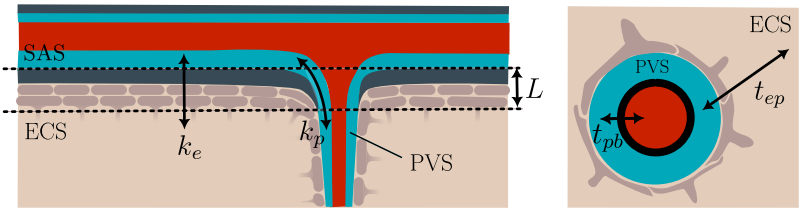


Fig. 8.2: **Boundary conditions and intercompartmental transfer.** Left: At the interface of the brain tissue to the SAS, diffusive transport between the SAS and the ECS and between the SAS and the PVS is described as diffusive transport across a membrane layer with conductivities k_e and k_p, respectively. Right: Intercompartmental transfer between PVS and blood and between PVS and ECS is described by the transfer coefficients t_{pb} and t_{ep}, respectively.

Intercompartmental transfer coefficients. We model the transfer coefficients between two compartments α and β by

$$t_{\alpha\beta} = \sigma_{\alpha\beta} C_{\alpha\beta} D^*, \tag{8.6}$$

where $\sigma_{\alpha\beta}$ is the volume-specific surface area of the intercompartmental interface (units m^{-1}) and $C_{\alpha\beta}$ (units m^{-1}) is a coefficient proportional to the inverse average path length a molecule has to travel from compartment α to compartment β.

The volume-specific surface area of blood vessels, σ_{bp}, has been reported to be between $10\,\text{mm}^{-1}$ and $13\,\text{mm}^{-1}$ in humans (Lauwers et al., 2008). The BBB is commonly assumed to be virtually impermeable to gadobutrol in healthy tissue (Jost et al., 2017; Weinmann et al., 1984). However, typical perfusion MRI sequences assess leakage on a time scale of minutes (Sourbron et al., 2009). When analyzing tracer transport with glymphatic MRI, the relevant time scale is days, and even very small transfer coefficients may have a significant effect on transport. Koch et al. (2020) estimated with a micro-scale model that a transfer coefficient equivalent to a value of $t_{pb} \lessapprox 1 \times 10^{-5}\,\text{s}^{-1}$ is unlikely to cause a significant leakage effect

on the signal during a typical 2 min duration of a dynamic susceptibility contrast MR perfusion measurement.[4] Chagnot et al. (2021) reviewed dynamic contrast-enhanced (DCE) MRI studies of contrast agent leakage across the BBB in patients with dementia as well as those with normal-appearing aging, with long acquisition times 5–25 min. They reported values for t_{pb} in the range $0.01–40 \times 10^{-6}\,\mathrm{s}^{-1}$, where the larger values are attributed to BBB breakdown due to various forms of dementia.[5]

In comparison with MRI data, the highest clearance rate estimated with computer simulations by Vinje et al. (2023, suppl.) corresponds to $t_{pb} \lessapprox 2.1 \times 10^{-5}\,\mathrm{s}^{-1}$. Here, we take t_{pb} in the range $0–2.1 \times 10^{-5}\,\mathrm{s}^{-1}$. Based on geometric reconstructions of two cortical vascular networks (mouse) and a theoretical model for the astrocyte endfoot sheath geometry, Koch et al. (2023) estimated a value for $\sigma_{ep}C_{ep}$ of $72–85\,\mathrm{mm}^{-2}$ corresponding to transfer coefficients of $t_{ep} = 2.7–3.1 \times 10^{-2}\,\mathrm{s}^{-1}$, using $D^\star = 3.7 \times 10^{-4}\,\mathrm{mm}^2/\mathrm{s}$. They also estimated that, along a pathway through the ECS and across the endfoot sheath, the effective permeability of the (longer) ECS pathway is approximately equal to the (shorter) pathway through the endfoot sheath and would add a factor of 0.5 to the estimate of t_{ep}. Furthermore, they estimated C_{ep} of the astrocyte endfoot sheath for arterioles, capillaries, and veins, with values in the range $0.5–6\,\mathrm{mm}^{-1}$, the smallest (corresponding to arterioles) being an order of magnitude smaller than the network average. Assuming this value for all relevant PVS, we obtain $t_{ep} \approx 1.7 \times 10^{-3}\,\mathrm{s}^{-1}$ ($C_{ep} = 0.25\,\mathrm{mm}^{-1}$, $\sigma_{ep} = 18\,\mathrm{mm}^{-1}$ (Koch et al., 2023, Tab. 1), $D^\star = 3.7 \times 10^{-4}\,\mathrm{mm}^2/\mathrm{s}$).

Boundary conditions and surface membrane. Assuming that there is no membrane obstructing the pathway from the PVS to the SAS, we can estimate the boundary conductivity for the PVS as

$$k_p = n_p L^{-1} D^*, \tag{8.7}$$

where L is the thickness of the boundary layer (see Fig. 8.2). The glia limitans of an adult human brain has been estimated at 20–40 μm (Brøchner et al., 2015). Assuming $L = 20\,\mu\mathrm{m}$ and $n_p = 0.04$ yields $k_p = 7.4 \times 10^{-4}\,\mathrm{mm\,s}^{-1}$.

The conductivity k_e of the membrane layer between the ECS and the SAS, forming the outer boundary of the ECS, is modeled as

$$k_e = C_{\mathrm{pial}} D^*, \tag{8.8}$$

where C_{pial} is a membrane coefficient proportional to the available pore space for transport across the layer and inversely proportional to its thickness. To estimate an upper bound for C_{pial}, we rescale the membrane coefficients for the astrocyte endfoot sheath around the largest penetrating arterioles $C_M = 0.5\,\mathrm{mm}$ (with a corresponding

[4] Combining a diffusive wall permeability of $1 \times 10^{-10}\,\mathrm{m\,s}^{-1}$ (Koch et al., 2020) with a volume-specific area of $10\,\mathrm{mm}^{-1}$ (Lauwers et al., 2008).

[5] Assuming t_{pb} is equal to K_{trans} of the Patlak model (Patlak et al., 1983; Sourbron et al., 2009) used in DCE MRI signal analysis. Note that t_{pb} is the micro-scale parameter and is likely larger than the corresponding voxel scale t_{pb}, which includes the resistance of the ECS and the astrocyte endfoot sheath, and appears in combination with the average intercompartmental concentration gradient instead of the micro-scale gradient across the endothelium.

sheath thickness[6] of $h_{\mathrm{ES}} \approx 2.8\,\mu\mathrm{m}$) as $C_{pial} = C_M h_{\mathrm{ES}} L^{-1}$, where L is again the thickness of the boundary layer. For $L = 20\mu m$, this corresponds to an upper bound of $k_e = 2.6 \times 10^{-5}\,\mathrm{mm\,s^{-1}}$. As a lower bound estimate, we consider the membrane layer to be virtually impermeable, $k_e \to 0$. As boundary conditions, we consider both Dirichlet boundary conditions (equivalent to the limit $k_\alpha \to \infty$) based on tissue concentration estimates from MRI data or Robin boundary conditions assuming a given tracer concentration in the SAS and ventricles.

In the case of Dirichlet boundary conditions, we use the interpolant u_d of the MRI data introduced in detail by Zapf et al. (2024). Between time steps with data (4 h, 24 h, 48 h, 70 h) the values are interpolated linearly in time on the domain boundary $\partial\Omega$. Since u_d represents the total concentration, the value must be divided by ϕ to obtain the fluid concentration in the compartments. To uniquely determine the fraction of the MRI signal from each of the compartments, we assume that the fluid concentrations of the compartments are in equilibrium, and thus $\hat{c} = u_d\,\phi = u_d/(n_e + n_p)$.

In the case of Robin boundary conditions, we consider different surface domains $\partial\Omega = \partial\Omega_{\mathrm{sas}} \cup \partial\Omega_{\mathrm{vent}}$ and model the CSF concentration in the SAS on the pial surface $\partial\Omega_{\mathrm{sas}}$ and the ventricular surface $\partial\Omega_{\mathrm{vent}}$ with the ansatz

$$\hat{c}(t, \partial\Omega_\zeta) = a_\zeta \phi^{-1} \left(-e^{-t/\tau_1} + e^{-t/\tau_2}\right), \quad \zeta = \{\mathrm{sas}, \mathrm{vent}\}, \tag{8.9}$$

and $\tau_1 = 4.43 \times 10^4$ s, $\tau_2 = 8.5 \times 10^4$ s, $a_{\mathrm{sas}} = 0.52\,\mathrm{mm^2/s}$, and $a_{\mathrm{vent}} = 0.2\,\mathrm{mm^2/s}$. A comparison between the CSF concentration calculated with Equation (8.9) and the surface-averaged concentrations of the interpolated MRI-data u_d is shown in Figure 8.3.

Initial concentration. Since the tracer is administered by intrathecal injection, there is initially no tracer in the brain tissue, $c_{p,0} = c_{e,0} = 0$.

8.3 Results

Parameter variations. A reference scenario is defined by choosing the parameters according to the nominal values given in Table 8.1. This implies Robin boundary conditions as described in the previous section. The Robin boundary conditions are used for all cases unless explicitly stated otherwise.

To investigate the parameter sensitivity of the two-compartment model, we varied individual parameters while keeping the other parameters fixed (reference scenario). The evolution of the total solute content in the brain over time is shown in Figure 8.4. The two-compartment model is most sensitive to variations of the effective perivascular diffusion coefficient, D_p^{eff}, and the transfer coefficient from PVS to blood, t_{pb}. For D_p^{eff}, there is an increase in the peak tracer content of 11% ($D_p^{\mathrm{eff}} = 10 D_e^{\mathrm{eff}}$) and 44%

[6] Using the formula $h_{\mathrm{ES}} = 1 + 0.15(r_v - 3)$ of Koch et al. (2023) with a vessel radius $r_v = 15\,\mu\mathrm{m}$.

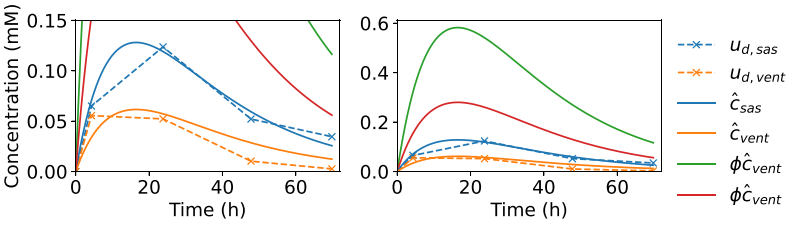


Fig. 8.3: Estimated mean SAS tracer concentration and mean tracer concentration in the ventricles (solid) in comparison with the boundary data (u_d) mean as estimated from MRI data (dashed). First, total concentrations in the tissue are obtained by fitting Equation (8.9) to the mean of u_d (left panel). Second, fluid concentrations in the SAS and the ventricles at the brain surface are estimated by division by the volume fraction available for fluid flow in the brain, $\phi = 0.2$ (right panel).

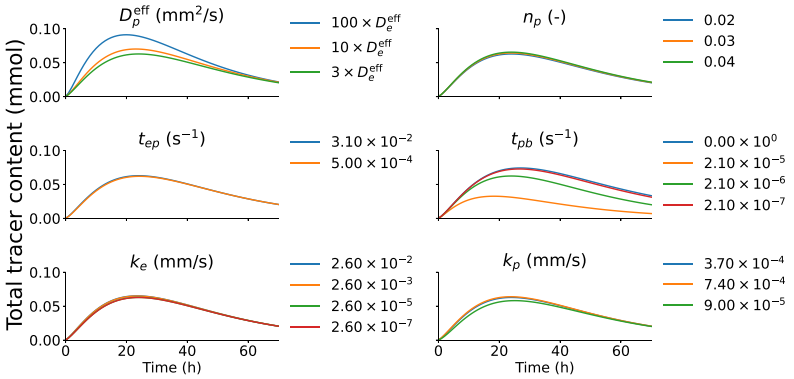


Fig. 8.4: The total tracer content in the brain over time, for various parameters in the two-compartment model. The parameters are described in Table 8.1. One parameter is varied while all other parameters are fixed at their nominal values listed in Table 8.1.

($D_p^{\text{eff}} = 100 D_e^{\text{eff}}$) with respect to the reference scenario ($D_p^{\text{eff}} = 3 D_e^{\text{eff}}$). For the transfer coefficient from PVS to blood, we observe a 48% decrease ($t_{pb} = 2.1 \times 10^{-5}\,\text{s}^{-1}$) and a 19% increase ($t_{pb} = 0\,\text{s}^{-1}$), respectively. For the remaining parameters, the relative difference between the minimum and maximum values for the peak tracer content is $\leq$ 5% for the perivascular volume fraction, $\leq$ 1.5% for the transfer between ECS and PVS, $\leq$ 4% for the ECS boundary conductivity, and $\leq$ 8% for the perivascular boundary conductivity.

Next, we explored scenarios in which the difference between the two-compartment model (featuring local concentration differences between the PVS and the ECS) and the single-compartment models (local equilibrium) becomes apparent. To this end,

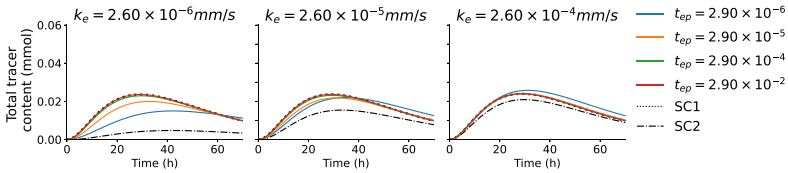


Fig. 8.5: The total tracer content in the brain for different effective transfer coefficients between ECS and PVS, t_{ep}, and extracellular surface-layer conductivity k_e for the different models. Solid lines represent the two-compartment model, whereas the dashed lines labelled **SC1** and **SC2** represent single-compartment models as stated in Equation (8.2).

we varied both the ECS surface conductivity k_e and the ECS–PVS transfer coefficient t_{ep}, keeping all other parameters as in the reference scenario. The resulting tracer content evolution in the brain tissue is presented in Figure 8.5. Decreasing k_e limits the tracer influx and efflux rates at the brain tissue surface, increasing the relative importance of the PVS pathway. Decreasing t_{ep} limits the exchange rate between the ECS and the PVS compartments.

Decreasing the ECS surface conductivity and ECS–PVS transfer simultaneously, we see lower tracer influx and efflux rates with respect to the reference scenario. A large difference between the equilibrium model **SC1** (or two-compartment model with a large t_{ep}) and the two-compartment model is evident for a low t_{ep} and, in particular, with decreasing k_e. As expected, the **SC1** model matches the results of the two-compartment model for a high t_{ep}, which is the limit under which it was derived.

Figure 8.6 shows the spatial concentration distribution ($t_{ep} = 2.0 \times 10^{-6}\,\mathrm{s}^{-1}$, $k_e = 2.6 \times 10^{-6}\,\mathrm{mm\,s}^{-1}$) in a coronal slice with tracer clearance to blood ($t_{pb} = 0.2 \times 10^{-5}\,\mathrm{s}^{-1}$) and without it ($t_{pb} = 0$). Without clearance to blood, in the influx phase ($\leq 24\,\mathrm{h}$) the tracer concentration increases monotonically from the boundary to the center of the brain, while the concentration is lower in the ECS than in the PVS ($c_e - c_p < 0$). However, clearance from the PVS to blood leads to a spatial concentration distribution, where, deep within the brain, $c_e - c_p \geq 0$, and $c_e - c_p < 0$ is only observed in a small region close to the brain surface, (see Fig. 8.6, second row). For the efflux phase ($> 24\,\mathrm{h}$), a monotone profile is established. The tracer is cleared both into the blood and from the surface into CSF. In the efflux phase, $c_e - c_p \leq 0$ everywhere, since the tracer is cleared faster from the PVS compartment than from the ECS compartment.

We note that similar qualitative observations hold for the reference scenario, although with much smaller concentration differences between the PVS and ECS compartments (less than 0.1% with respect to the maximum concentration after 24 h). Thus, in the reference scenario, the fluid concentrations in both compartments are in equilibrium, and assumption **SC1** is fulfilled.

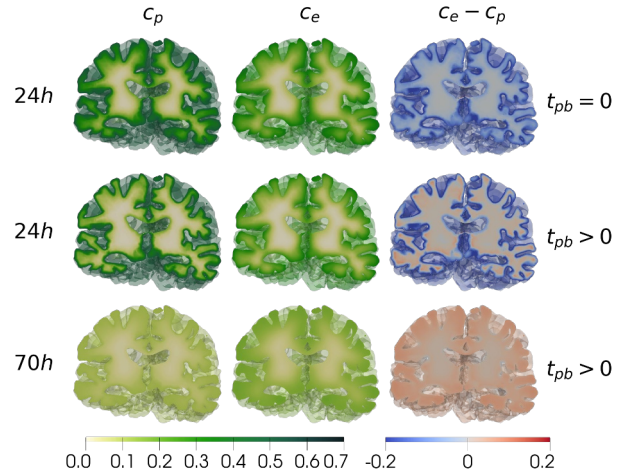


Fig. 8.6: Spatial distribution of the fluid concentrations in the PVS compartment, c_p, and the ECS compartment, c_e, at two different timepoints and for two different blood clearance rates, $t_{pb} = 0.2 \times 10^{-5}\,\mathrm{s}^{-1}$ and $t_{pb} = 0$, on a coronal slice.

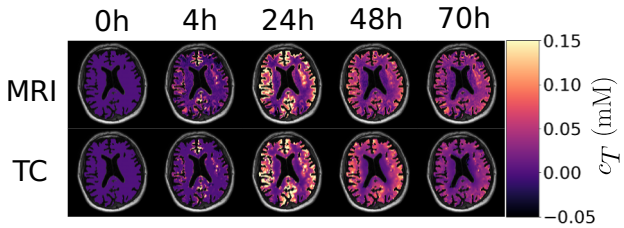


Fig. 8.7: Spatial distributions of the total concentration, c_T, for the two-compartment (TC) model (bottom) and the concentration estimates derived from MRI data (top) over time. The true value range (−0.2, 1.2) is clipped to the range (−0.05, 0.15) to remove outliers impairing visual interpretability. Negative concentrations result from noise in the MRI data, since the raw MRI data have a low signal-to-noise ratio (Valnes et al., 2020).

Comparison with MRI data. We analyzed the brain tracer content evolution simulated with Dirichlet boundary conditions ($k_\alpha \to \infty$) set by MRI data on the boundary and compared it with the corresponding MRI-based tracer content estimates within the brain tissue. The parameters were set to the reference scenario (except for k_α). Figure 8.7 shows the total concentration c_T estimated from MRI data and simulated with the two-compartment model on a transverse slice over 72 h.

Figure 8.8 shows the total tracer content in the whole brain and separately in gray and white matter. The peak tracer content for the **SC2** model is reduced with respect to the two-compartment model (6%, 3%, and 11% decreases in the whole brain, gray

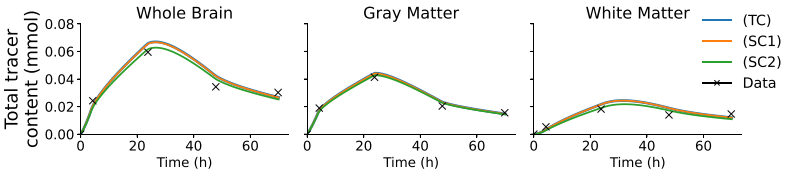


Fig. 8.8: Total tracer content in the whole brain, gray matter, and white matter for data derived from MRI and simulations with the two- and single-compartment models. The label TC represents the two-compartment model, whereas the labels SC1 and SC2 represent corresponding single-compartment models as stated in Equation (8.2).

matter, and white matter, respectively), while **SC1** and the two-compartment model coincide, signifying local equilibrium between the ECS and PVS concentrations. The simulation results and data match well without systematically optimizing the parameters. With the reference parameter configuration, the model overestimates the tracer content compared to data in both white and gray matter in the approximate interval of 10–55 h. Moreover, it can be observed that the maximum solute content is attained later than 24 h (where the boundary data show the maximum value). Due to the blood clearance (as shown in Fig. 8.6), the tracer concentration close to the surface is likely lower than the concentration prescribed at the boundary and remains lower for a few hours after the 24 h timepoint, resulting in continued influx and delaying the beginning of the efflux phase.

8.4 Discussion

Fast equilibrium between the PVS and ECS. The single-compartment model (**SC1**) is derived from the two-compartment model by assuming a sufficiently high transfer coefficient t_{ep} for the compartments to be in equilibrium, an assumption that is supported by the comparison with the concentration evolution estimated from MRI data (Fig. 8.7). For a local, non-equilibrium configuration to be established between the PVS and ECS compartments, the transfer coefficient t_{ep} (which is proportional to the astrocyte endfoot sheath permeability) would need to be an order of magnitude smaller than the estimated lower bound supported by data in the literature (see Tab. 8.1), even for low ECS conductivity at the brain surface (Fig. 8.5). We can support the observations with the two-compartment model by simple theoretical considerations. The ratio of the ECS–PVS transfer rate across the astrocyte endfoot sheath and the transversal diffusion rate in the ECS is given by the Damkhler number,[7] $\mathrm{Da} = t_{ep} L_{ep} (n_e D_e^{\mathrm{eff}} \sigma_{ep})^{-1}$, where L_{ep} is a characteristic transversal length scale (the mean inter-vessel distance). For $L_{ep} = 50\,\mu\mathrm{m}$ (as the mean inter-

[7] Balancing fluxes over the astrocyte endfoot sheath interface Γ yields $\int_\Gamma -n_e D_e^{\mathrm{eff}} (\nabla c_e \cdot \boldsymbol{n})\, \mathrm{d}s = \int_\Omega t_{ep}(c_p - c_e)\, \mathrm{d}x$. Assuming uniform parameters and normal fluxes, we can integrate and non-

vessel distance), we find Da ≈ 0.3–3 and conclude that both rates are approximately equal. A corresponding characteristic time scale is given by $T = t_{ep}^{-1} \approx 30$–$600\,\mathrm{s}$ (using the ranges from Tab. 8.1). The time scale for longitudinal transport in the PVS is given by $T_d = L_\ell^2 (D_p^{\mathrm{eff}})^{-1}$, where L_ℓ is a characteristic longitudinal length scale. For $L_\ell = 0.5\,\mathrm{cm}$ and $D_e^{\mathrm{eff}} \leq D_p^{\mathrm{eff}} < 100 D_e^{\mathrm{eff}}$, we obtain $T_d \approx 2$–$200 \times 10^3\,\mathrm{s}$. Hence, even for the upper estimate for D_p^{eff}, the transversal transport rate dominates the longitudinal transport rate, leading to a fast equilibrium of the concentrations in the PVS and ECS compartments.

Brain surface membranes have little impact on influx and efflux rates. Increasing the PVS surface conductivities beyond the nominal values has very little impact on the brain tracer content (Fig. 8.4), indicating that, for $k_p \gtrapprox 1 \times 10^{-4}\,\mathrm{mm/s}$, the surface membrane poses no significant barrier for tracer influx along the PVS. Similarly, given fast ECS–PVS exchange, varying the ECS surface membrane conductivity k_{ep} within the range supported by literature values has little impact on the tracer content in the brain. The tracer can circumvent the ECS surface layer by entering the brain along the more permeable PVS, spreading to the ECS through the astrocyte endfoot sheath.

Effective diffusivity and clearance to blood. The two parameters with the most significant impact on the two-compartment model (in the reference scenario) are the perivascular diffusion coefficient D_p^{eff} and the PVS–blood transfer coefficient t_{pb} (see Fig. 8.5).

The effective diffusion coefficient, D^{eff}, of the single-compartment model (**SC1**) is related to the individual effective diffusion coefficients of the compartments and $D^{\mathrm{eff}}/D_e^{\mathrm{eff}} = (n_p\gamma + n_e)/(n_p + n_e)$ (where $\gamma = D_e^{\mathrm{eff}}/D_p^{\mathrm{eff}}$). The ratio $D^{\mathrm{eff}}/D_e^{\mathrm{eff}}$ was estimated by Ray et al. (2021) based on MRI data in mice at 10–25, corresponding to $\gamma = 100$–265 (with $n_e/n_p = 10$). Vinje et al. (2023) estimated $D^{\mathrm{eff}}/D_e^{\mathrm{eff}}$ between 1.1 and 7.0, corresponding to $\gamma = 2$–67. For the single subject investigated here, we obtained a good fit between the data and simulation with $D^{\mathrm{eff}}/D_e^{\mathrm{eff}} = 1$.

The PVS–blood transfer coefficient t_{pb} models clearance of gadobutrol across the BBB. The reference value of $t_{pb} = 2.1 \times 10^{-6}\,\mathrm{s}^{-1}$ is within the range of the blood-to-brain transport coefficients of gadolinium-based contrast agents reported in the review article Chagnot et al. (2021) of 0.01–$34 \times 10^{-6}\,\mathrm{s}^{-1}$ for normal aging, 0.5–$13 \times 10^{-6}\,\mathrm{s}^{-1}$ for cerebral small vessel disease, and 0.01–$40 \times 10^{-6}\,\mathrm{s}^{-1}$ in patients with mild cognitive impairment or Alzheimer's disease. Disorders such as multiple sclerosis and glioma are known to cause a breakdown of the BBB, leading to an increase of up to several magnitudes in blood vessel wall permeability. The MRI data used in this study fall in the category of normal aging.

dimensionalize ($\nabla^\star := L_{ep}\nabla$, $c_\alpha^\star := C_0 c_\alpha$, $\sigma_{ep} = (\int_\Gamma \mathrm{d}s)(\int_\Omega \mathrm{d}x)^{-1}$) to arrive at $-\nabla^* c_e^* \cdot \boldsymbol{n} = \mathrm{Da}\,(c_p^* - c_e^*)$, where $\mathrm{Da} = t_{ep} L_{ep} (n_e D_e^{\mathrm{eff}} \sigma_{ep})^{-1}$.

8.5 Conclusion

A two-compartment model of tracer transport in brain tissue allowed us to investigate the effect of compartment-specific diffusion coefficients and membrane-specific transfer coefficients, and we related the model parameters of the two-compartment model to previously employed single-compartment models. We found that, for a wide range of parameters, a single-compartment model (**SC1**) serves as a good approximation to the two-compartment model due to the fast transfer between the ECS and PVS compartments through the astrocyte endfoot sheath. In comparison with concentration distributions from reference MRI data, we find that the combination of extracellular diffusion, fast exchange between the ECS and PVS compartments, and tracer clearance to blood provide a model that explains the tracer evolution in the patient-specific brain tissue geometry over 70 h after intrathecal injection of a gadolinium-based contrast agent.

References

Arbogast T, Lehr HL (2006) Homogenization of a Darcy–Stokes system modeling vuggy porous media. Computational Geosciences 10(3):291–302, doi:10.1007/s10596-006-9024-8

Asgari M, de Zlicourt D, Kurtcuoglu V (2016) Glymphatic solute transport does not require bulk flow. Sci Rep 6(1):38635, doi:10.1038/srep38635

Barisano G, Sheikh-Bahaei N, Law M, Toga AW, Sepehrband F (2021) Body mass index, time of day and genetics affect perivascular spaces in the white matter. Journal of Cerebral Blood Flow & Metabolism 41(7):1563–1578, doi:10.1177/0271678X20972856

Baxter LT, Jain RK (1989) Transport of fluid and macromolecules in tumors. I. Role of interstitial pressure and convection. Microvascular Research 37(1):77–104, doi:10.1016/0026-2862(89)90074-5

Blinder P, Tsai PS, Kaufhold JP, Knutsen PM, Suhl H, Kleinfeld D (2013) The cortical angiome: an interconnected vascular network with noncolumnar patterns of blood flow. Nature Neuroscience 16(7):889–897, doi:10.1038/nn.3426

Bohr T, et al. (2022) The glymphatic system: Current understanding and modeling. Iscience 25:104987, doi:10.1016/j.isci.2022.104987

Bojarskaite L, et al. (2023) Sleep cycle-dependent vascular dynamics in male mice and the predicted effects on perivascular cerebrospinal fluid flow and solute transport. Nature Communications 14(1):953, doi:10.1038/s41467-023-36643-5

Brøchner CB, Holst CB, Møllgrd K (2015) Outer brain barriers in rat and human development. Frontiers in Neuroscience 9:75, doi:10.3389/fnins.2015.00075

Chagnot A, Barnes SR, Montagne A (2021) Magnetic Resonance Imaging of Blood–Brain Barrier permeability in Dementia. Neuroscience 474:14–29, doi:10.1016/j.neuroscience.2021.08.003

Duvernoy H, Delon S, Vannson J (1981) Cortical blood vessels of the human brain. Brain Research Bulletin 7(5):519–579, doi:10.1016/0361-9230(81)90007-1

Ehlers W, Wagner A (2013) Multi-component modelling of human brain tissue: a contribution to the constitutive and computational description of deformation, flow and diffusion processes with application to the invasive drug-delivery problem. Computer Methods in Biomechanics and Biomedical Engineering 18(8):861–879, doi:10.1080/10255842.2013.853754

Guthausen G, et al. (2015) Characterisation and application of ultra-high spin clusters as magnetic resonance relaxation agents. Dalton Transactions 44(11):5032–5040, doi:10.1039/c4dt02916j

Holter KE, Kehlet B, Devor A, Sejnowski TJ, Dale AM, Omholt SW, Ottersen OP, Nagelhus EA, Mardal KA, Pettersen KH (2017) Interstitial solute transport in 3D reconstructed neuropil occurs by diffusion rather than bulk flow. Proceedings of the National Academy of Sciences 114(37):9894–9899, doi:10.1073/pnas.1706942114

Hua J, Liu P, Kim T, Donahue M, Rane S, Chen JJ, Qin Q, Kim SG (2019) MRI techniques to measure arterial and venous cerebral blood volume. NeuroImage 187:17–31, doi:10.1016/j.neuroimage.2018.02.027

Iliff JJ, et al. (2012) A Paravascular Pathway Facilitates CSF Flow Through the Brain Parenchyma and the Clearance of Interstitial Solutes, Including Amyloid β. Science Translational Medicine 4(147):147ra111–147ra111, doi:10.1126/scitranslmed.3003748

Ito H, Kanno I, Iida H, Hatazawa J, Shimosegawa E, Tamura H, Okudera T (2001) Arterial fraction of cerebral blood volume in humans measured by positron emission tomography. Annals of Nuclear Medicine 15(2):111–116, doi:10.1007/bf02988600

Jessen NA, Munk ASF, Lundgaard I, Nedergaard M (2015) The Glymphatic System: A Beginner's Guide. Neurochem Res 40(12):2583–2599, doi:10.1007/s11064-015-1581-6

Jost G, et al. (2017) Penetration and distribution of gadolinium-based contrast agents into the cerebrospinal fluid in healthy rats: a potential pathway of entry into the brain tissue. European Radiology 27(7):2877–2885, doi:10.1007/s00330-016-4654-2

Keith Sharp M, Carare RO, Martin BA (2019) Dispersion in porous media in oscillatory flow between flat plates: applications to intrathecal, periarterial and paraarterial solute transport in the central nervous system. Fluids and Barriers of the CNS 16(1):1–17, doi:10.1186/s12987-019-0132-y

Koch T, Flemisch B, Helmig R, Wiest R, Obrist D (2020) A multiscale subvoxel perfusion model to estimate diffusive capillary wall conductivity in multiple sclerosis lesions from perfusion MRI data. International Journal for Numerical Methods in Biomedical Engineering 36(2), doi:10.1002/cnm.3298

Koch T, Vinje V, Mardal KA (2023) Estimates of the permeability of extra-cellular pathways through the astrocyte endfoot sheath. Fluids and Barriers of the CNS 20(1):20, doi:10.1186/s12987-023-00421-8

Lauwers F, Cassot F, Lauwers-Cances V, Puwanarajah P, Duvernoy H (2008) Morphometry of the human cerebral cortex microcirculation: General characteristics and space-related profiles. NeuroImage 39(3):936–948, doi:10.1016/j.neuroimage.2007.09.024

Leenders K, et al. (1990) Cerebral blood flow, blood volume and oxygen utilization: normal values and effect of age. Brain 113(1):27–47, doi:10.1093/brain/113.1.27

Mestre H, et al. (2018) Flow of cerebrospinal fluid is driven by arterial pulsations and is reduced in hypertension. Nat Commun 9(1):4878, doi:10.1038/s41467-018-07318-3

Muizelaar JP, Fatouros PP, Schröder ML (1997) A New Method for Quantitative Regional Cerebral Blood Volume Measurements Using Computed Tomography. Stroke 28(10):1998–2005, doi:10.1161/01.str.28.10.1998

Patlak CS, Blasberg RG, Fenstermacher JD (1983) Graphical Evaluation of Blood-to-Brain Transfer Constants from Multiple-Time Uptake Data. Journal of Cerebral Blood Flow & Metabolism 3(1):1–7, doi:10.1038/jcbfm.1983.1

Payne S, Mai VP (2023) Human whole-brain models of cerebral blood flow and oxygen transport. Brain Multiphysics 5:100083, doi:10.1016/j.brain.2023.100083

Poulain A, Riseth J, Vinje V (2023) Multi-compartmental model of glymphatic clearance of solutes in brain tissue. Plos one 18(3):e0280501, doi:10.1371/journal.pone.0280501

Raicevic N, Forer JM, de Guevara AL, Du T, Nedergaard M, Kelley DH, Boster K (2023) Sizes and shapes of perivascular spaces surrounding murine pial arteries. Fluids and Barriers of the CNS 20(1), doi:10.1186/s12987-023-00454-z

Ray LA, Heys JJ (2019) Fluid Flow and Mass Transport in Brain Tissue. Fluids 4(4), doi:10.3390/fluids4040196

Ray LA, et al. (2021) Quantitative analysis of macroscopic solute transport in the murine brain. Fluids and Barriers of the CNS 18(1):55, doi:10.1186/s12987-021-00290-z

Ringstad G, Vatnehol SAS, Eide PK (2017) Glymphatic MRI in idiopathic normal pressure hydrocephalus. Brain 140(10):2691–2705, doi:10.1093/brain/awx191

Ringstad G, Valnes LM, Dale AM, Pripp AH, Vatnehol SAS, Emblem KE, Mardal KA, Eide PK (2018) Brain-wide glymphatic enhancement and clearance in humans assessed with MRI. JCI Insight 3(13), doi:10.1172/jci.insight.121537

Riseth J (2024) jorgenriseth/mri2fem-multicompartment: v1.0. doi:10.5281/zenodo.13941194
Rognes ME, Mardal KA, Valnes LM, Vinje V (2024) From brain physiology to brain physics. In: Dokken JS, et al. (eds) Mathematical modeling of the human brain (vol II): from glymphatics to deep learning, Springer
Schain AJ, Melo-Carrillo A, Strassman AM, Burstein R (2017) Cortical spreading depression closes paravascular space and impairs glymphatic flow: Implications for migraine headache. The Journal of Neuroscience 37(11):2904–2915, doi:10.1523/jneurosci.3390-16.2017
Schmid F, Barrett MJ, Jenny P, Weber B (2019) Vascular density and distribution in neocortex. NeuroImage 197:792–805, doi:10.1016/j.neuroimage.2017.06.046
Shipley RJ, Chapman SJ (2010) Multiscale Modelling of Fluid and Drug Transport in Vascular Tumours. Bull Math Biol 72(6):1464–1491, doi:10.1007/s11538-010-9504-9
Sourbron S, Ingrisch M, Siefert A, Reiser M, Herrmann K (2009) Quantification of cerebral blood flow, cerebral blood volume, and blood–brain-barrier leakage with DCE-MRI. Magnetic Resonance in Medicine 62(1):205–217, doi:10.1002/mrm.22005
Syková E, Nicholson C (2008) Diffusion in brain extracellular space. Physiological Reviews 88(4):1277–1340, doi:10.1152/physrev.00027.2007, pMID: 18923183
Tithof J, Boster KA, Bork PA, Nedergaard M, Thomas JH, Kelley DH (2022) A network model of glymphatic flow under different experimentally-motivated parametric scenarios. iScience 25(5):104258, doi:10.1016/j.isci.2022.104258
Valnes LM, Mitusch SK, Ringstad G, Eide PK, Funke SW, Mardal KA (2020) Apparent diffusion coefficient estimates based on 24 hours tracer movement support glymphatic transport in human cerebral cortex. Scientific Reports 10(1):1–12, doi:10.1038/s41598-020-66042-5
Vardakis JC, Guo L, Chou D, Ventikos Y (2021) Using Multicompartmental Poroelasticity to Explore Brain Biomechanics and Cerebral Diseases. In: Braza M, et al. (eds) Advances in Critical Flow Dynamics Involving Moving/Deformable Structures with Design Applications, Springer, pp 151–163, doi:10.1007/978-3-030-55594-8_15
Vinje V, Ringstad G, Lindstrøm EK, Valnes LM, Rognes ME, Eide PK, Mardal KA (2019) Respiratory influence on cerebrospinal fluid flow – a computational study based on long-term intracranial pressure measurements. Scientific Reports 9(1):9732, doi:10.1038/s41598-019-46055-5
Vinje V, Zapf B, Ringstad G, Eide PK, Rognes ME, Mardal KA (2023) Human brain solute transport quantified by glymphatic MRI-informed biophysics during sleep and sleep deprivation. Fluids and Barriers of the CNS 20(1):62, doi:10.1186/s12987-023-00459-8
Weinmann HJ, Brasch RC, Press WR, Wesbey GE (1984) Characteristics of gadolinium-DTPA complex: a potential NMR contrast agent. AJR American journal of roentgenology 142(3):619–624, doi:10.2214/ajr.142.3.619
Xie L, et al. (2013) Sleep Drives Metabolite Clearance from the Adult Brain. Science 342(6156):373–377, doi:10.1126/science.1241224
Zapf B, Valnes LM, Mardal KA, Zikatanov L (2024) Quantifying cerebrospinal fluid tracer concentration in the brain. In: Dokken JS, et al. (eds) Mathematical modeling of the human brain (vol II): from glymphatics to deep learning, Springer

Chapter 9

An introduction to identifying velocity fields from contrast imaging via PDE-constrained optimization

Marie E. Rognes

Abstract Repeated contrast-enhanced magnetic resonance imaging is developing as a technique for studying the molecular entryways and clearance routes of the human brain, for which many transport parameters are unknown. This approach provides a sequence of images representing the concentration c of the contrast agent at high resolution in space and low resolution in time. A natural question is whether and to what extent these imaging data can inform molecular transport parameters via inverse computational modeling. More specifically, we ask here, Given concentration fields c_1 and c_2 measured at t_1 and t_2, is there a velocity field ϕ that transports c_1 at t_1 to c_2 at t_2? This chapter gives a rapid and practical introduction to different methods for addressing this *optimal transport* problem, including implementations of each method in the FEniCS software.

9.1 Introduction

Given a pair of images representing concentration fields $c_1 = c_1(x)$ and $c_2 = c_2(x)$ defined over $x \in \Omega \subset \mathbb{R}^d$ at times t_1 and t_2, respectively, can we identify a vector field $\phi = \phi(x)$ that maps c_1 to c_2? This problem setting is readily encountered when leveraging contrast-enhanced magnetic resonance imaging to inform computational models of molecular transport within the brain (Ringstad et al., 2018; Vinje et al., 2023). It is also a classical problem setting in image processing, referred to as the optical flow problem (see, e.g., the highly readable paper by Wildes et al. (2000)). Several physics-based (as well as non–physics-based) approaches to determining optical flow exist; a key idea is to use a mass continuity equation to constrain a velocity vector field in combination with the minimization of, for example, kinetic energy (Benamou and Brenier, 2000).

In the context of brain solute transport and clearance, Tannenbaum and coauthors (Ratner et al., 2015, 2017) pioneered the use of optical flow–based methods to

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_9

extract glymphatic flow fields from series of two-photon fluorescence microscopy images. Their approach, as described by Mueller et al. (2013), estimates a flow field via an optimal mass transport–based (OMT) approach to optical flow. More recently, Vinje et al. (2023) quantified human brain interstitial fluid flow velocities during solute clearance from contrast-enhanced magnetic resonance images using an optimal convection–diffusion (OCD) approach. This chapter presents mathematical and numerical formulations for both methods and compares their accuracy when tasked with identifying a velocity field in a manufactured convection–diffusion example. The numerical implementation is based on the FEniCS and dolfin-adjoint finite element and adjoint-based optimization software (Alnæs et al., 2015; Farrell et al., 2013; Mitusch et al., 2019) and is freely available (Rognes, 2024).

9.2 Mathematical Models for Flow Field Estimation

Let Ω be an open, bounded domain in $\mathbb{R}^d$ ($d = 1, 2, 3$) with coordinates x and let $t \in (t_1, t_N)$. Generally, our goal is to identify a flow (vector) field $\phi = \phi(x, t)$ from a discrete sequence of solute concentration or signal intensity images $c_n = c_n(x)$ at times t_n for $n = 1, 2, \ldots, N$. For compactness in presentation, we let $N = 2$ and denote $\tau = t_2 - t_1$.

9.2.1 Optical Flow and Optimal Mass Transport

We begin by considering an *optimal mass transport* approach to *optical flow* (Mueller et al., 2013). Assume that the transport of the concentration c by the velocity field ϕ is governed by the continuity equation:

$$f(c, \phi) \equiv c_t + \operatorname{div}(c\phi) = 0 \qquad \text{in } \Omega, \quad t > 0. \tag{9.1}$$

We define the kinetic energy associated with c and ϕ as follows:

$$R(c, \phi) = \int_{t_1}^{t_2} \int_\Omega \phi \cdot \phi c \, \mathrm{d}x \, \mathrm{d}t,$$

which will play the role of a *regularization term* or *prior*. The OMT problem is now to find $\phi = \phi(x, t)$ that minimizes

$$\min_\phi \frac{1}{2} \int_{t_1}^{t_2} \int_\Omega f(c, \phi)^2 \, \mathrm{d}x \, \mathrm{d}t + \frac{1}{2} \alpha R(c, \phi) \tag{9.2}$$

with $c(t_1) = c_1$ and $c(t_2) = c_2$ and where $\alpha > 0$ is a parameter. Variations on this formulation easily result from different choices of the regularization functional R. For example, Mueller et al. (2013) also consider a version with L^2 regularization:

$$R(c,\phi) = \int_0^T \int_\Omega \phi \cdot \phi \,\mathrm{d}x\,\mathrm{d}t,$$

which they solve numerically using a discretize-then-minimize approach with finite differences in time and space. We consider here a revised formulation using a variational formulation and finite elements in space instead. To this end, we discretize (9.2) in time using the midpoint rule, that is, we approximate the time integral as

$$\int_{t_1}^{t_2} f(t)\,\mathrm{d}t \approx \tau f(t^{1/2}),$$

where $t^{1/2} = \frac{1}{2}(t_1 + t_2)$ is the midpoint of the time interval. We approximate the time derivative via the natural first-order approximation,

$$c_t \approx \frac{c_2 - c_1}{\tau} \equiv \delta c,$$

and then approximate ϕ as a constant in time by its midpoint value, $\phi = \phi(x) \approx \phi(x, t^{1/2})$, and introduce the average concentration

$$\bar{c} \equiv \frac{1}{2}(c_1 + c_2).$$

Since c_1 and c_2 are given, $\bar{c}$ and δc are known (computed) fields. The time-discrete OMT minimization problem then reads as follows:

$$\min_\phi M(\phi) = \frac{1}{2}\tau \left(\int_\Omega (\delta c + \operatorname{div}(\phi\bar{c}))^2 \,\mathrm{d}x + \alpha \int_\Omega (\phi \cdot \phi)\bar{c}\,\mathrm{d}x \right). \tag{9.3}$$

The optimal solution of the minimization problem (9.3) satisfies the Euler–Lagrange equations involving the Gateaux derivative of M in the direction $\psi = \psi(x)$:

$$\frac{\partial}{\partial \epsilon} M(\phi + \epsilon\psi)\Big|_{\epsilon=0} = 0.$$

Computing the Gateaux derivative, we find

$$\begin{aligned}
&\frac{\partial}{\partial \epsilon} M(\phi + \epsilon\psi)\Big|_{\epsilon=0} \\
&= \frac{\partial}{\partial \epsilon} \left(\frac{1}{2}\tau \left(\int_\Omega (\delta c + \operatorname{div}((\phi + \epsilon\psi)\bar{c}))^2 \,\mathrm{d}x + \alpha \int_\Omega ((\phi + \epsilon\psi) \cdot (\phi + \epsilon\psi))\bar{c}\,\mathrm{d}x \right) \right) \Big|_{\epsilon=0} \\
&= \frac{1}{2}\tau \left(\int_\Omega 2(\delta c + \operatorname{div}(\bar{c}\phi))(\operatorname{div}(\bar{c}\psi))\,\mathrm{d}x + \alpha \int_\Omega 2\phi \cdot \psi\bar{c}\,\mathrm{d}x \right) \\
&= \tau \left(\int_\Omega (\delta c + \operatorname{div}(\bar{c}\phi))(\operatorname{div}(\bar{c}\psi))\,\mathrm{d}x + \alpha \int_\Omega \phi \cdot \psi\bar{c}\,\mathrm{d}x \right).
\end{aligned}$$

The resulting OMT variational problem reads as follows: find $\phi \in V$ such that for all $\psi \in V$

$$a(\phi, \psi) \equiv \langle \operatorname{div}(\bar{c}\phi), \operatorname{div}(\bar{c}\psi) \rangle + \alpha \langle \bar{c}\phi, \psi \rangle = -\langle \delta c, \operatorname{div} \bar{c}\psi \rangle \equiv L(\psi), \tag{9.4}$$

where we have introduced the familiar notation for the $L^2(\Omega)$ inner product

$$\langle \phi, \psi \rangle = \int_\Omega \phi \cdot \psi \, \mathrm{d}x.$$

This is a linear variational problem that can be solved numerically using straightforward finite element techniques in, for example, FEniCS.

To also include transport by diffusion, we can replace (9.1) by a convection–diffusion equation (more details follow below). The same steps as for the OMT formulation above again yields a linear variational problem for ϕ, but now also involving second-order derivatives of $\bar{c}$. Evaluating such terms for low-regularity data, for example, images with noise, pose additional challenges.

9.2.2 An Optimal Convection-diffusion Formulation

In the spirit of Glowinski et al. (2022) and Andreev et al. (2015), an alternative approach is to define a bilinear optimal control problem where both the final concentration and the convective velocity are treated as variables. This formulation lends itself easily to also accounting for diffusion and/or reaction terms.

So, let us consider the convection–diffusion equation

$$F(c, \phi) \equiv c_t + \operatorname{div}(\phi c) - \operatorname{div} D\nabla c = 0 \qquad \text{in } \Omega, \quad t > 0 \tag{9.5}$$

and assume that observations of the concentration c_1, c_2 at t_1, t_2 are given. We are now interested in c and ϕ that both satisfy (9.5) and that best match observations—in the sense that c and ϕ minimize the data misfit objective functional J:

$$J(c, \phi) = \frac{1}{2} \| c(t_2) - c_2 \|^2_{L^2(\Omega)} + R(c, \phi), \tag{9.6}$$

recalling that c_2 is the given data field and $c(t_2)$ is the solution to (9.5) at $t = t_2$ and where R again represents a regularization term or prior. Specifically, the minimization problem of interest reads as follows:

$$\min_{c,\phi} J(c, \phi) \quad \text{such that} \quad F(c, \phi) = 0, \quad c(t_1) = c_1,$$

where we have also imposed the observations at t_1 as an initial condition for c. Observe that F is nonlinear when viewed as a function of both c and ϕ. In the optimization literature, the pattern with a multiplicative control variable (here ϕ) is referred

to as a bilinear control problem. Using a Lagrange multiplier technique (Hinze et al., 2008), let us now introduce the Lagrangian L, defined by

$$L(c, \phi, y) = J(c, \phi) - \langle F(c, \phi), y \rangle_{Y \times Y*}$$

for $c \in C$, $\phi \in Q$, and the adjoint variable $y \in Y$. For our example, the Lagrangian thus becomes

$$L(c, \phi, y) = \frac{1}{2} \|c(t_2) - c_2\|^2_{L^2(\Omega)} + R(c, \phi) + \langle c_t + \operatorname{div}(\phi c) - \operatorname{div} D \nabla c, y \rangle_{Y \times Y*}. \quad (9.7)$$

Discretizing the Lagrangian (9.7) in time with an implicit Euler scheme and using the same time resolution as that of the given data, it remains to compute the single (in time) approximations $c(x) = c(x, t_2)$, $\phi(x) = \phi(x, t)$, and $y(x) = y(x, t)$. In addition, we identify the $Y^* \times Y$ duality pairing with the $L^2(\Omega)$ inner product, denoted $\langle \cdot, \cdot \rangle$, to obtain the time-discrete Lagrangian:

$$L_\tau(c, \phi, y) = \frac{1}{2} \|c - c_2\|^2_{L^2(\Omega)} + R(c, \phi) + \langle \tau^{-1}(c - c_1) + \operatorname{div}(\phi c) - \operatorname{div}(D \nabla c), y \rangle_{L^2(\Omega)}.$$

Now, we again take the Gateaux derivatives of L_τ with respect to c, ϕ, and y to derive the optimality conditions. First, taking the derivatives with respect to $y \in Y$ gives the state equation:

$$\frac{\partial}{\partial y} L_\tau(c, \phi, y)[dy] = \langle \tau^{-1} c + \operatorname{div}(\phi c) - \operatorname{div}(D \nabla c) - \tau^{-1} c_1, dy \rangle.$$

Next, taking the derivative with respect to $c \in C \subseteq H^1(\Omega)$ in the direction of $dc \in C$ gives the adjoint equation:

$$\frac{\partial}{\partial c} L_\tau(c, \phi, y)[dc] = \langle c - c_2, dc \rangle_{L^2(\Omega)} + R_c(c, \phi)[dc] + \langle \tau^{-1} dc + \operatorname{div}(\phi dc) - \operatorname{div}(D \nabla dc), y \rangle.$$

Taking the derivative with respect to $\phi \in Q \subseteq H(\operatorname{div}, \Omega)$ in the direction of $d\phi \in Q$ gives

$$\frac{\partial}{\partial \phi} L_\tau(c, \phi, y)[d\phi] = R_\phi(c, \phi)[d\phi] + \langle \operatorname{div}(c d\phi), y \rangle.$$

Finally, we need to select a regularization term R. Andreev et al. (2015) considered an $H(\operatorname{div})$-regularization, which is a very natural Sobolev space setting for the velocity field. To enforce a bit more regularity, we use H^1 regularization here,

$$R(c, \phi) = R(\phi) = \frac{1}{2} \alpha \left(\|\phi\|^2 + \| \nabla \phi \|^2 \right),$$

for a regularization parameter $\alpha \geq 0$. Thus $R_c = 0$, while

$$R_\phi(c, \phi)[d\phi] = \alpha \left(\langle \phi, d\phi \rangle + \langle \nabla \phi, \nabla d\phi \rangle \right).$$

With these assumptions and some renaming of the test functions, we obtain the following time-independent but nonlinear system of governing equations, where we find $c \in C$, $\phi \in Q$, and $y \in Y$ such that

$$\begin{aligned} \langle c, d \rangle + \langle \tau^{-1} d + \operatorname{div}(\phi\, d) - \operatorname{div} D\nabla d, y \rangle &= \langle c_2, d \rangle, \\ \langle \operatorname{div}(c\, \psi), y \rangle + \alpha \left(\langle \phi, \psi \rangle + \langle \nabla \phi, \nabla \psi \rangle \right) &= 0, \\ \langle \tau^{-1} c + \operatorname{div}(\phi\, c) - \operatorname{div}(D\nabla c), z \rangle &= \langle \tau^{-1} c_1, z \rangle \end{aligned}$$

for all $d \in C$, $z \in Y$, and $\psi \in Q$. To reduce regularity requirements, we integrate second-order derivatives by parts. To address the resulting boundary terms, we prescribe Dirichlet boundary conditions for the concentration c and corresponding homogeneous Dirichlet conditions for the adjoint field y:

$$c = c_2 \quad y = 0 \qquad \text{on } \partial\Omega.$$

In summary, for the OCD method, we solve the following nonlinear variational formulation: for given image data $c_1, c_2 \in L^2(\Omega)$, time step $\tau > 0$, diffusion tensor $D \in L^2(\Omega)$, and regularization parameter $\alpha \geq 0$, find $c \in H^1_{c_1}(\Omega)$, $y \in H^1_0(\Omega)$, $\phi \in H^1(\Omega; R^d)$ such that

$$\langle c, d \rangle + \langle \tau^{-1} d + \operatorname{div}(\phi\, d), y \rangle + \langle D\nabla d, \nabla y \rangle = \langle c_2, d \rangle, \tag{9.9a}$$

$$\langle \operatorname{div}(c\, \psi), y \rangle + \alpha \left(\langle \phi, \psi \rangle + \langle \operatorname{div} \phi, \operatorname{div} \psi \rangle \right) = 0, \tag{9.9b}$$

$$\langle \tau^{-1} c + \operatorname{div}(\phi\, c), z \rangle + \langle D\nabla c, \nabla z \rangle = \langle \tau^{-1} c_1, z \rangle \tag{9.9c}$$

for all $d \in H^1(\Omega)$, $z \in H^1_0(\Omega)$, $\psi \in H^1(\Omega; \mathbb{R}^d)$. We discretize this variational formulation by a finite element method with continuous piecewise linear elements for all fields and solve the resulting system of nonlinear algebraic equations using Newton's method. The image data c_1 for c and $\phi = 0$ provide a natural initial guess.

9.2.3 Reduced Optimal Convection-diffusion (rOCD) Formulation

Instead of solving the OCD optimization problem (9.9) via the "all-at-once" Lagrange multiplier approach (9.9), we may consider an iterative so-called reduced approach (Farrell et al., 2013; Hinze et al., 2008). To this end, we introduce the reduced objective functional $\tilde{J}$, defined as J (see (9.6)), viewed in terms of the control parameter ϕ only, and target finding the ϕ that minimize $\tilde{J}$:

$$\min_\phi \tilde{J}(\phi) = J(c(\phi), \phi). \tag{9.10}$$

The convection–diffusion equation (9.5) defines the map $\phi \mapsto c(\phi)$.

To solve the rOCD problem numerically, we discretize the convection–diffusion equation in time from t_1 to t_2, again using the interval between images as the time step $\tau = t_2 - t_1$ for simplicity. With an implicit Euler scheme, the equation for $c \approx c(t_2)$ becomes

$$c - \tau\,(\operatorname{div}(\phi c) - \operatorname{div} D\,\nabla c)) = c_1 \tag{9.11}$$

with the initial condition $c(t_1) = c_1$. To discretize (9.11) in space, we can use a standard finite element variational formulation and impose the Dirichlet boundary condition $c = c_2$ on $\partial\Omega$ (see (9.2.2)).

We can then solve the optimization problem (9.10) using standard (unconstrained) optimization algorithms, for instance, Newton or quasi-Newton methods (Al-Baali et al., 2014) such as the popular L-BFGS algorithm (Liu and Nocedal, 1989). In practice, this is easily accomplished via the dolfin-adjoint library with FEniCS or Firedrake as the finite element backends (Farrell et al., 2013).

9.3 Numerical Comparison of the OMT, OCD, and rOCD Methods

To study the accuracy and predictability of the different algorithms (OMT, OCD, and rOCD), we consider here a numerical test case with a manufactured solution (Fig. 9.1, Tab. 9.1). Interestingly, we observe that the OMT velocity (Fig. 9.1D) differs substantially from the original convective contribution, both qualitatively and quantitatively, and that its norm(s) varies strongly with the choice of parameter α. On the other hand, the OCD and rOCD velocity fields are very similar, qualitatively agree well with the original velocity, and are robust with respect to the regularization parameters for small and moderately small α, but both underestimate the velocity magnitude and show bias toward regions with higher concentration values. A full-scale application of the OCD and rOCD methods to identify convective fluid flow fields from human glymphatic MRI images is presented by Vinje et al. (2023).

9.4 Concluding Remarks and Outlook

In conclusion, we highlight the following observations. First, the OMT formulation is easy to implement and relatively inexpensive, but it is found here to be sensitive to numerical parameters such as the choice of regularization parameters and mesh resolution and can yield spurious velocity field patterns. Moreover, this OMT formulation does not account for the diffusion directly. Regularized or unbalanced OMT formulations (Chen et al., 2023, 2024) may address these points. On the other hand, the OCD and rOCD approaches are both also easily implemented and more compu-

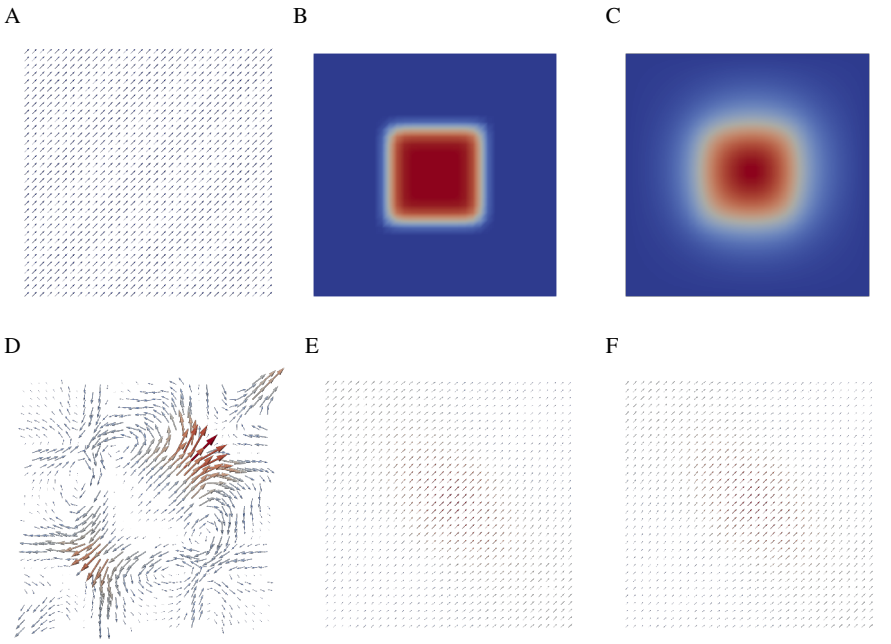


Fig. 9.1: Velocity field estimation with known data. Consider a (unitless) manufactured test case with a given velocity field $\phi = (0.02, 0.02)$ (A) and diffusion coefficient $D = 0.01$ over the domain $[0, 1]^2 \subset \mathbb{R}^2$. A smooth bump function (B) is used as initial data c_1 at $t_1 = 0$ and the continuous piecewise linear finite element/implicit Euler approximation to (9.5) with zero boundary condition at $t_2 = 1$ (C) are used as synthetically generated data. D–F: The velocity fields recovered by the OMT (D), OCD (E), and rOCD (F) approaches ($\alpha = 10^{-3}$, $h = \frac{1}{32}$, fields scaled 2×).

tationally expensive, but they yield more robust results with regard to the numerical parameters. Solvers for the nonlinear OCD problem (e.g., Newton's method) may easily fail to converge, and, similarly, many optimization iterations may be required for the rOCD approach. Finally, note that none of the approaches considered here enforces a volume preservation constraint

$$\operatorname{div} \phi = 0 \quad \text{in } \Omega, \quad t > 0, \tag{9.12}$$

in contrast to, for example, the study by Glowinski et al. (2022).

Acknowledgements The author graciously acknowledges valuable discussions and input from Dr. Vegard Vinje and Dr. Bastian Zapf.

A: OMT vs α

α	$\|\phi\|_{L^2}$	$\|\phi\|_{L^\infty}$	$\|\phi\|_{H_0^1}$
10^0	0.0129	0.0302	0.1686
10^{-1}	0.0171	0.0542	0.2459
10^{-2}	0.0236	0.0786	0.3332
10^{-3}	0.0282	0.0845	0.4181
10^{-4}	0.0448	0.0851	0.6477
10^{-5}	0.0709	0.1441	0.6857
Exact	0.0283	0.0283	0.0000

B: OCD vs α

α	$\|\phi\|_{L^2}$	$\|\phi\|_{L^\infty}$	$\|\phi\|_{H_0^1}$
10^0	0.0049	0.0070	0.0068
10^{-1}	0.0088	0.0124	0.0119
10^{-2}	0.0098	0.0135	0.0131
10^{-3}	0.0100	0.0137	0.0133
10^{-4}	0.0100	0.0137	0.0133
10^{-5}	0.0100	0.0137	0.0133
Exact	0.0283	0.0283	0.0000

C: rOCD vs α

α	$\|\phi\|_{L^2}$	$\|\phi\|_{L^\infty}$	$\|\phi\|_{H_0^1}$
10^0	0.0049	0.0070	0.0068
10^{-1}	0.0088	0.0124	0.0119
10^{-2}	0.0099	0.0136	0.0131
10^{-3}	0.0103	0.0139	0.0131
10^{-4}†	0.0116	0.0148	0.0122
10^{-5}†	0.0029	0.0069	0.0208
Exact	0.0283	0.0283	0.0000

D: OMT vs n

n	$\|\phi\|_{L^2}$	$\|\phi\|_{L^\infty}$	$\|\phi\|_{H_0^1}$
4	0.0129	0.0402	0.1574
8	0.0313	0.0714	0.4666
16	0.0282	0.0845	0.4181
32	0.0236	0.0763	0.3756
64	0.0217	0.0714	0.3568
128	0.0191	0.0617	0.3209
Exact	0.0283	0.0283	0.0000

E: OCD vs n

n	$\|\phi\|_{L^2}$	$\|\phi\|_{L^\infty}$	$\|\phi\|_{H_0^1}$
4	0.0027	0.0046	0.0078
8	0.0095	0.0132	0.0131
16	0.0100	0.0137	0.0133
32	0.0103	0.0140	0.0134
64	0.0104	0.0141	0.0134
128	0.0104	0.0141	0.0134
Exact	0.0283	0.0283	0.0000

F: rOCD vs n

n	$\|\phi\|_{L^2}$	$\|\phi\|_{L^\infty}$	$\|\phi\|_{H_0^1}$
4	0.0026	0.0044	0.0079
8	0.0093	0.0130	0.0132
16	0.0103	0.0139	0.0131
32	0.0101	0.0139	0.0136
64	0.0099	0.0137	0.0138
128	0.0100	0.0138	0.0138
Exact	0.0283	0.0283	0.0000

Table 9.1: For all methods, for varying α, $n = 16$, while for varying n, $\alpha = 10^{-3}$. † indicates that the maximal number of iterations (500) has been reached. For the rOCD method, we used the L-BFGS algorithm with gtol set to 10^{-12} and a maximal number of iterations of 500.

References

Al-Baali M, Spedicato E, Maggioni F (2014) Broyden's quasi-Newton methods for a nonlinear system of equations and unconstrained optimization: a review and open problems. Optimization Methods and Software 29(5):937–954, doi:10.1080/10556788.2013.856909

Alnæs MS, et al. (2015) The fenics project version 1.5. Archive of Numerical Software 3, doi:10.11588/ans.2015.100.20553

Andreev R, Scherzer O, Zulehner W (2015) Simultaneous optical flow and source estimation: Space-time discretization and preconditioning. Applied Numerical Mathematics 96:72–81, doi:10.1016/j.apnum.2015.04.007

Benamou JD, Brenier Y (2000) A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem. Numerische Mathematik 84(3):375–393, doi:10.1007/s002110050002

Chen X, Tran AP, Elkin R, Benveniste H, Tannenbaum AR (2023) Visualizing Fluid Flows via Regularized Optimal Mass Transport with Applications to Neuroscience. Journal of Scientific Computing 97(2):26, doi:10.1007/s10915-023-02337-9

Chen X, Benveniste H, Tannenbaum AR (2024) Unbalanced regularized optimal mass transport with applications to fluid flows in the brain. Scientific Reports 14(1):1111, doi:10.1038/s41598-023-50874-y

Farrell PE, Ham DA, Funke SW, Rognes ME (2013) Automated derivation of the adjoint of high-level transient finite element programs. SIAM Journal on Scientific Computing 35(4):C369–C393, doi:10.1137/120873558

Glowinski R, Song Y, Yuan X, Yue H (2022) Bilinear Optimal Control of an Advection-Reaction-Diffusion System. SIAM Review 64(2):392–421, doi:10.1137/21M1389778

Hinze M, Pinnau R, Ulbrich M, Ulbrich S (2008) Optimization with PDE constraints, vol 23. Springer Science & Business Media, doi:10.1007/978-1-4020-8839-1

Liu DC, Nocedal J (1989) On the limited memory BFGS method for large scale optimization. Mathematical Programming 45(1):503–528, doi:10.1007/BF01589116

Mitusch SK, Funke SW, Dokken JS (2019) dolfin-adjoint 2018.1: automated adjoints for FEniCS and Firedrake. Journal of Open Source Software 4(38):1292, doi:10.21105/joss.01292

Mueller M, Karasev P, Kolesov I, Tannenbaum A (2013) Optical Flow Estimation for Flame Detection in Videos. IEEE Transactions on Image Processing 22(7):2786–2797, doi:10.1109/TIP.2013.2258353

Ratner V, Zhu L, Kolesov I, Nedergaard M, Benveniste H, Tannenbaum A (2015) Optimal-mass-transfer-based estimation of glymphatic transport in living brain. In: Ourselin S, Styner MA (eds) Medical Imaging 2015: Image Processing, International Society for Optics and Photonics, SPIE, vol 9413, p 94131J, doi:10.1117/12.2076289

Ratner V, Gao Y, Lee H, Elkin R, Nedergaard M, Benveniste H, Tannenbaum A (2017) Cerebrospinal and interstitial fluid transport via the glymphatic pathway modeled by optimal mass transport. NeuroImage 152:530–537, doi:10.1016/j.neuroimage.2017.03.021

Ringstad G, Valnes LM, Dale AM, Pripp AH, Vatnehol SAS, Emblem KE, Mardal KA, Eide PK (2018) Brain-wide glymphatic enhancement and clearance in humans assessed with MRI. JCI Insight 3(13), doi:10.1172/jci.insight.121537

Rognes ME (2024) meg-simula/optimal-transport-fenics: v1.0. doi:10.5281/zenodo.10623495

Vinje V, Zapf B, Ringstad G, Eide PK, Rognes ME, Mardal KA (2023) Human brain solute transport quantified by glymphatic MRI-informed biophysics during sleep and sleep deprivation. Fluids and Barriers of the CNS 20(1):62, doi:10.1186/s12987-023-00459-8

Wildes RP, Amabile MJ, Lanzillotto AM, Leu TS (2000) Recovering Estimates of Fluid Flow from Image Sequence Data. Computer Vision and Image Understanding 80(2):246–266, doi:10.1006/cviu.2000.0874

Chapter 10

An introduction to network models of neurodegenerative diseases

Georgia S. Brennan and Alain Goriely

Abstract Neurodegenerative diseases are characterized by the buildup of key toxic proteins. For instance, Alzheimer's disease, impacting over 50 million people worldwide, is associated with amyloid beta and tau proteins. The spatial and temporal evolution of these proteins and their association with cognitive decline suggest the existence of underlying principles that might be modeled mathematically. Fundamentally, the spread of these diseases is characterized by the transport, expansion, and saturation of toxic proteins in the brain. We show that a simple model that includes all these aspects and reproduces the basic invasion patterns is a network-based model. This reductionist model effectively approximates the full dynamics. This chapter is conceived as a gentle step-by-step introduction to these ideas and models.

10.1 Introduction

Alzheimer's disease affects more than 50 million people worldwide and is found in about one in nine people aged 65 and older. Unlike cancer and many viral and bacterial diseases, Alzheimer's disease, in its most common form, presents itself in stages that have been codified and are found in most patients. We also know that this cognitive staging is associated with a systematic invasion in the brain of two key toxic proteins that were identified by Alois Alzheimer in 1905. The first one is the well-known amyloid beta, an extracellular protein whose natural functions are not well understood. The second one is the group of tau proteins that normally stabilize microtubules in axons. However, they can misfold. It is believed that a misfolded version of tau acts as a template for healthy tau proteins and promotes the formation of oligomers of increasing sizes, eventually leading to large aggregates that can be observed in brain tissues after death (Jucker and Walker, 2013). While most drug trials focus on amyloid beta, it is the presence of toxic tau proteins that is most

J. S. Dokken et al. (eds.), *Mathematical Modelling of the Human Brain II*,
Simula SpringerBriefs on Computing 18, https://doi.org/10.1007/978-3-032-00679-0_10

strongly correlated to brain atrophy and to cognitive symptoms. The typical pattern of tau evolution obtained through histological staining is known as the Braak staging, after the neuroanatomists Eva and Heiko Braak who first proposed it in 1991 (Braak and Braak, 1991). It describes the evolution of the disease in six stages, starting in the entorhinal cortex and evolving through the hippocampal region (associated with memory), the temporal lobe, the occipital lobe, and all regions of the neocortex, with lesions in each region worsening in time.

The prion-like hypothesis broadly postulates that neurodegenerative diseases result from an accumulation of abnormally misfolded proteins that aggregate and contribute to tissue death, causing associated neurodegenerative pathology and cognitive decline. In this process, certain disease-specific misfolded proteins can influence healthy proteins, forming extensive chains that can be transported through the brain along axonal pathways. Given that aggregates of differing sizes exhibit unique transport characteristics and varying levels of toxicity, it is crucial to independently monitor their spatial and temporal evolution.

From a modeling point of view, the systematic pattern of invasion through the brain and the multiple cognitive effects suggest to the curious mind that some simple underlying features of the brain could be responsible for this spatiotemporal pattern. The challenge is therefore to obtain minimal mathematical models based on clear principles that can capture, at the brain level, the staging as well as other characteristics of the disease, such as brain atrophy and changes in overall brain dynamics.

From a phenomenological point of view, there are three processes to consider: *(transport)*, where toxic proteins are transported in the brain, mostly along axonal bundles (connecting different parts of the brain and acting as information highways); *(expansion)*, where the aggregation process is autocatalytic and leads to an initial exponential increase of small toxic populations; and *(saturation)*, where each region can only support a certain level of toxic proteins. A canonical model for such a process in a continuum medium is the celebrated Fisher–Kolmogorov–Petrovsky–Piskunov (Fisher-KPP) equation. If $p(x,t) \in [0,1]$ is the scaled concentration of the toxic protein, the equation reads as follows:

$$\frac{\partial p}{\partial t} = \nabla \cdot (\mathbf{D}\nabla p) + \alpha\, p\,(1-p), \tag{10.1}$$

where $\mathbf{D}$ is a transversely anisotropic diffusion tensor with a strong preferential direction along the axonal bundle and $\alpha > 0$ is the growth rate. Remarkably, this model can be obtained, under generic conditions, as a normal form of the full aggregation–fragmentation equations that track the evolution of oligomers of different sizes (Thompson et al., 2021). Hence the parameters can be directly related to the aggregation and clearance rates at the microscopic level.

Modern brain imaging techniques are now well developed, and it is relatively routine to use medical resonance imaging to obtain both the full brain geometry as well as the direction of axonal bundles. Then, starting with an initial seed of toxic proteins

in the entorhinal cortex, we simulated the evolution of the field in the brain as shown in Figure 10.1. Without considering added complexities of extracellular diffusion, aggregation kinetics, or clearance, the dynamics obtained from this minimal model recovered the basic dynamics of the disease in great detail (Weickenmeier et al., 2018). Further, the same model applied to other neurodegenerative diseases associated with other toxic proteins, such as α-synuclein for Parkinson's or TDP-43 for amyotrophic lateral sclerosis, also reproduces their basic spatiotemporal patterns as well as atrophy patterns obtained post-processing from finite element simulations, the only difference being the region and extent of seeding. Despite the complexity and diversity of these diseases, universal features of progression emerge from the combination of an autocatalytic process and transport in an anisotropic medium.

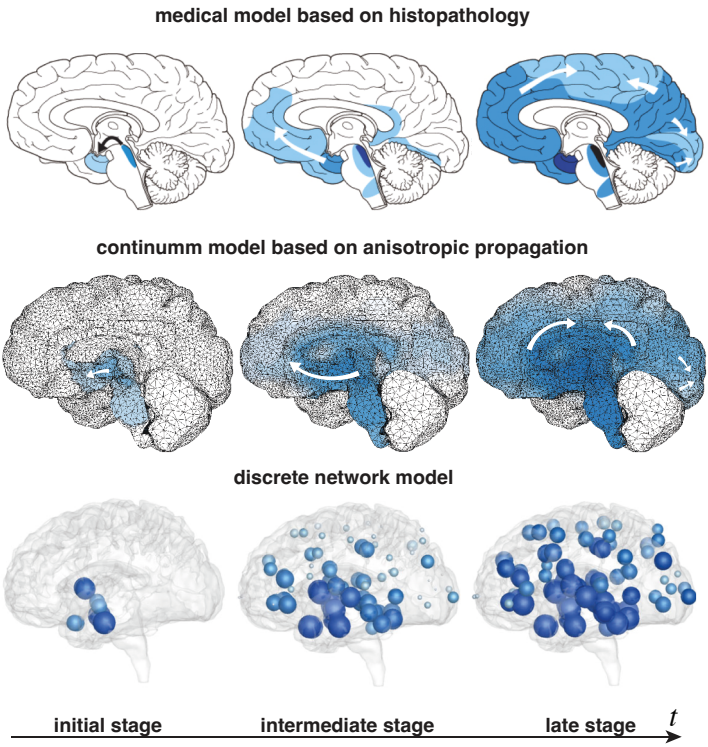


Fig. 10.1: Progression of toxic tau proteins in the brain. Top: medical staging based on histopathology (the analysis of postmortem brain slices). Middle: Simulation of the anisotropic Fisher-KPP model with an initial value in the entorhinal cortex. Bottom: Simulation of the network Fisher-KPP model with initial seeding at the entorhinal node (adapted from Fornari et al. (2019)).

When such strong universal patterns appear, a natural question emerges: what are the essential features responsible for the invasion pattern? Since the autocatalytic dynamics cannot be further simplified, we can look into the transport term and test the hypothesis that the observed patterns are a consequence of the strong transport anisotropy along the axonal bundles. These bundles are not only information highways; they could also carry efficiently toxic proteins across the brain. A simple way to test this idea is to coarse-grain the brain by considering multiple regions and the connections between them. The full brain is then replaced by a network, *the connectome*, where each node is a region and an edge represents the possible connections between regions. This approach is the basis for a large field of neuroscience, and various groups have successfully used the connectome to look at the effect of protein diffusion in the brain (Raj et al., 2012).

The natural discretization of the diffusion operator on a network is the weighted graph Laplacian $\mathbf{L}$ with weights proportional to the number of connections and inversely proportional to the squared distance between nodes. Then, defining $p_i(t)$ to be the concentration of toxic proteins at node i, the discrete Fisher-KPP equation reads as follows:

$$\dot{p}_i = -\rho \sum_{j=1}^{N} L_{ij} p_j + \alpha p_i (1 - p_i), \quad i = 1, \ldots, N. \tag{10.2}$$

This system of nonlinear ordinary differential equations turns out to be an excellent approximation of the dynamics generated by the full nonlinear partial differential equations that we started with, as shown in Figure 10.1.

The remainder of the chapter is an in-depth technical study and derivation of this network model. All files and the Mathematica workbook are provided by Goriely and Brennan (2024).

10.2 Network Models

10.2.1 Diffusion Models

Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be a network with node set $\mathcal{N}$ and edge set $\mathcal{E}$. Since we are interested in diffusion processes, we assume that the network is connected (otherwise we can consider diffusion on each component of the network). Consider two nodes in $\mathcal{N}$ with labels i and j and denote by a_{ij} the weight of the edge connecting node j to node i. We define the weighted adjacency matrix $\mathbf{A}$ as the $N \times N$ matrix with entries

$$A_{ij} = a_{ij}, \quad i, j = 1, \ldots, N. \tag{10.3}$$

We are interested in modeling diffusion-like transport processes on the network where each node has a spatial location $\mathbf{x}_i$ denoting the central point of a region of

typical volume V_i. In the first instance, we consider the case in which each region has the same constant volume $V = V_i$, $\forall i$, and we will consider the correction due to the volume later on. We define at each node a concentration p_i, representing the amount of a certain quantity defined in that region.

The natural coarse-graining of the diffusion operator on a regular lattice is the *graph Laplacian*. Therefore, our goal is to define a graph Laplacian $\mathbf{L}$ that models basic transport properties on the network so that, in the absence of other physical processes, the evolution of the concentration is given by

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\rho \mathbf{L} \cdot \mathbf{p}, \tag{10.4}$$

where $\mathbf{p} = (p_1, \ldots, p_N)$ is the (column) vector of concentrations and $\rho > 0$ is a diffusion-like constant (note that the minus sign in front of $\mathbf{L}$ is just a matter of convention and has no particular significance) and the dot represents the regular index contraction, $(\mathbf{L} \cdot \mathbf{p})_i = \sum_{j=1}^{N} L_{ij} p_j$.

There are two natural constraints associated with a transport process that depends on the relative concentration between two different regions. First, we assume that the concentration represents a physical quantity and that the transport process does not change the total amount of that quantity, in our case, the total mass of a given chemical. Therefore, mass conservation imposes the total mass in the system,

$$M = \sum_{i=1}^{N} V_i p_i, \tag{10.5}$$

to remain constant in time. Therefore, using the assumption that all volumes are equal, we have

$$\sum_{i=1}^{N} V_i \frac{\mathrm{d}p_i}{\mathrm{d}t} = V \mathbf{1} \cdot \frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = 0, \tag{10.6}$$

where $\mathbf{1} = (1, \ldots, 1)$ is the one vector. Using (10.4), this condition implies

$$\mathbf{1} \cdot \mathbf{L} \cdot \mathbf{p} = 0, \tag{10.7}$$

which must be true for all $\mathbf{p}$. Hence, we have

$$\mathbf{1} \cdot \mathbf{L} = \mathbf{0}, \tag{10.8}$$

where $\mathbf{0} = (0, \ldots, 0)$ is the null vector. In other words, the one vector is in the left null space of $\mathbf{L}$. We address heterogeneity in the regional volume later in this chapter.

The existence of at least one zero eigenvalue for $\mathbf{L}$ implies that the right null space is non-empty. Hence, there exists a vector $\widehat{\mathbf{p}}$ such that

$$\mathbf{L} \cdot \widehat{\mathbf{p}} = \mathbf{0}, \tag{10.9}$$

which defines a steady-state concentration.

10.2.2 Fick's Condition for Undirected Networks

Next, we assume that the network is undirected and that transport is driven by differences in concentration between different regions. In this case, we have $\mathbf{L} = \mathbf{L}^T$, which implies, for connected networks, that the steady concentration is uniform $\widehat{\mathbf{p}} = M/N\mathbf{1}$. This condition is related to Fick's condition (10.10) that, in the absence of a concentration gradient, there is no diffusion flux. The discrete form of this condition is that, if all concentrations are equal, there is no change in concentration. Hence, for undirected networks, we have a second condition:

$$\mathbf{L} \cdot \mathbf{1} = \mathbf{0}. \tag{10.10}$$

We note that this condition is not as stringent as the condition on mass conservation. Indeed, other assumptions are possible, for instance, when the transport process is viewed as a random walk. Masuda et al. (2017) studied *continuous-time random walks* where the walker waits until the next move for a time τ, with τ a random variable chosen to follow a Poisson process (τ is distributed according to the exponential distribution with parameter $\lambda = 1$). We can imagine two cases.

In the *edge-centric* case, each edge is opened independently with a given Poisson process with a rate proportional to the edge weight. Once that edge is open, a walker can move through it. In this case, the equation for the evolution of the probabilities, $\tilde{\mathbf{p}}$, is

$$\frac{\mathrm{d}\tilde{\mathbf{p}}}{\mathrm{d}t} = -\mathbf{L} \cdot \tilde{\mathbf{p}}, \tag{10.11}$$

where $\mathbf{D} = \mathrm{diag}(\mathbf{d})$ is the *degree matrix*, a diagonal matrix with diagonal elements $\mathbf{d} = \mathbf{A} \cdot \mathbf{1}$ and

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \tag{10.12}$$

where $\mathbf{L}$ is the *standard graph Laplacian*. In particular, in the unweighted case, the weight is taken to be $a_{ij} = 1$ whenever an edge is present between i and j, as shown in Figure 10.2. In the particular Boolean case, a walker travels at a unit rate and goes from a node to its neighbor with rate one. That is, a traveler leaves a node i with a rate d_i, whereas in the previous case the transition rate for leaving a node is the same for all nodes.

In the *node-centric* case, a walker moves from node i to node j with a probability proportional to a_{ij}. Then the evolution of the probability density $\tilde{\mathbf{p}}$ (i.e., $\tilde{p}_i$ gives the probability that a walker visits node i) is

$$\frac{\mathrm{d}\tilde{\mathbf{p}}}{\mathrm{d}t} = -\mathbf{L}\mathbf{D}^{-1}\tilde{\mathbf{p}}. \tag{10.13}$$

In the particular Boolean case, a walker travels at a unit rate and goes from a node to its neighbor with a rate proportional to its degree (a node with three numbers has a probability 1/3 of jumping to any one of them).

10.2.3 The Graph Laplacians of Connected and Undirected Networks

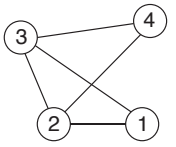
For a weighted, connected, and undirected network, the standard network diffusion equation is obtained by choosing the standard graph Laplacian

$$\frac{\mathrm{d}\mathbf{p}}{\mathrm{d}t} = -\rho \mathbf{L} \cdot \mathbf{p}. \tag{10.14}$$

The definition of $\mathbf{D}$ and the symmetry of $\mathbf{A}$ imply

$$\mathbf{D} \cdot \mathbf{1} = \mathbf{1} \cdot \mathbf{D} = \mathbf{A} \cdot \mathbf{1} = \mathbf{1} \cdot \mathbf{A} = \mathbf{d}. \tag{10.15}$$

Other possible Laplacian matrices can be defined through $\mathbf{A}$ and $\mathbf{D}$ and have been



$$A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad L = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ 0 & -1 & -1 & 2 \end{bmatrix}$$

Fig. 10.2: A simple example of an undirected unweighted graph with its adjacency and Laplacian matrix.

used in various studies such as spectral clustering. These include

$$\mathbf{L}_{1,0} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A} = \mathbf{D}^{-1}\mathbf{L}, \tag{10.16}$$

$$\mathbf{L}_{0,1} = \mathbf{I} - \mathbf{A}\mathbf{D}^{-1} = \mathbf{L}\mathbf{D}^{-1}, \tag{10.17}$$

$$\mathbf{L}_{1/2,1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}, \tag{10.18}$$

where $\mathbf{I}$ is the identity matrix. These choices motivate the introduction of the family

$$\mathbf{L}_{a,b} = \mathbf{D}^{1-a-b} - \mathbf{D}^{-a}\mathbf{A}\mathbf{D}^{-b} = \mathbf{D}^{-a}\mathbf{L}\mathbf{D}^{-b}. \tag{10.19}$$

The dynamics generated by these Laplacians only depend on the sum $a + b$. Indeed, consider the system (10.4) and introduce the change of variable $\mathbf{q} = \mathbf{D}^c\mathbf{p}$. We then have

$$\frac{\mathrm{d}\mathbf{q}}{\mathrm{d}t} = -\rho \mathbf{D}^c \mathbf{L}_{a,b} \mathbf{D}^{-c} \cdot \mathbf{q}, = -\rho \mathbf{L}_{a-c,b+c} \cdot \mathbf{q}. \tag{10.20}$$

Hence, up to a scaling of the variables, the three systems (10.16–10.18) have the same dynamics (same eigenvalues and scaled eigenvectors). However, the Laplacian $\mathbf{L}_{0,0}$ is not part of that family, and no straightforward connection between its spectrum and the spectrum of other Laplacians can be made.

We can now check, within the family $\mathbf{L}_{a,b}$, which Laplacian satisfies Fick's condition:

$$\mathbf{L}_{a,b} \cdot \mathbf{1} = \mathbf{D}^{1-a-b} \cdot \mathbf{1} - \mathbf{D}^{-a}\mathbf{A}\mathbf{D}^{-b} \cdot \mathbf{1}, \tag{10.21}$$

which implies, after multiplication by $\mathbf{D}^a$,

$$\mathbf{D}^{1-b} \cdot \mathbf{1} = \mathbf{A}\mathbf{D}^{-b} \cdot \mathbf{1}. \tag{10.22}$$

This last equality can only be true if $\mathbf{A}\mathbf{D}^{-b} = \mathbf{D}^{-b}\mathbf{A}$. Indeed, assuming $b \neq 0$, if there exist i and j such that $d_i \neq d_j$ (otherwise $\mathbf{D}$ is a multiple of the identity and all Laplacians are equal up to a multiple), we have $(\mathbf{A}\mathbf{D}^{-b})_{ij} = d_j^{-b}\mathbf{A}_{ij} \neq (\mathbf{D}^{-b}\mathbf{A})_{ij} = d_i\mathbf{A}_{ij}$. Hence we conclude that, unless $d_i = d_j$ for all i, j, we must have $b = 0$.

The same reasoning applied to the mass conservation condition implies that, unless $d_i = d_j$ for all i, j, we must have $a = 0$. Indeed, if $a \neq 0$, then the total mass is not conserved in the dynamics. Indeed, the stationary state is generally given by a multiple of $\widehat{\mathbf{p}} = \mathbf{D}^b \cdot \mathbf{1}$. An initial concentration $\mathbf{p_0}$ will evolve asymptotically (as $t \to \infty$) toward $\mathbf{p}_\infty = \lambda(\mathbf{p}_0)\widehat{\mathbf{p}}$, where $\lambda(\mathbf{p}_0)$ depends linearly on the initial conditions (e.g., for a symmetric $\mathbf{L}_{a,b}$, we have simply $\lambda = \mathbf{p}_0 \cdot \widehat{\mathbf{p}}/\widehat{\mathbf{p}} \cdot \widehat{\mathbf{p}}$). Therefore, an initial mass $m_0 = \mathbf{1} \cdot \mathbf{p}_0$ evolves asymptotically to a mass

$$m_\infty = \lambda\, \mathbf{1} \cdot \widehat{\mathbf{p}}. \tag{10.23}$$

Depending on the initial conditions, degrees, and choice of $a \neq 0$, m_∞ can be larger, smaller, or equal to m_0. From a modeling perspective, it should be clear that such a choice cannot be justified. A change in mass should be properly modeled and cannot depend on the topology of the graph or the initial conditions. Although a normalized graph Laplacian (with $a \neq 0$) is a convenient object for many mathematical studies, it cannot be used to model a transport process (and modeling a transport process that includes either the creation or removal of material is problematic, since such a process would depend on the topology of the graph and not a physical process).

Hence, for a generic graph we conclude that *the only Laplacian* $\mathbf{L}_{a,b}$ *that satisfies both the mass conservation and Fick's conditions is the standard graph Laplacian* $\mathbf{L} = \mathbf{L}_{0,0}$. This is an important conclusion, since other Laplacians have been used in the literature (Abdelnour et al., 2014).

Correction for Varying Volumes

If the nodes have different volumes $\mathbf{v} = (V_1, \ldots, V_N)$, then the condition for the conservation of mass must be modified. Indeed, the total mass is now $M = \mathbf{v} \cdot \mathbf{p}$.

Enforcing $\dot{M} = 0$ in (10.4) implies

$$\mathbf{v} \cdot \mathbf{L} = \mathbf{0}. \tag{10.24}$$

This condition is satisfied by choosing

$$\mathbf{L} = \mathbf{L}_V = \mathbf{V}^{-1}\mathbf{L}, \qquad \mathbf{V} = 1/V_0 \mathrm{diag}(\mathbf{v}), \tag{10.25}$$

where V_0 is a reference volume (average or total volume, e.g.). We note that this modified graph Laplacian is not symmetric but that Fick's condition is still satisfied ($\mathbf{L}_V \cdot \mathbf{1} = \mathbf{0}$), since diffusion takes place when a concentration gradient is established, independently of the node's volume.

In the context of modeling neurodegenerative diseases, we note that the multiplication of the standard Laplacian on the left by a diagonal matrix has also been used to define regions of vulnerability (Henderson et al., 2020, 2019), thus assuming that diffusion takes place differently in different nodes. Mathematically, it is the same operation, but its interpretation in terms of volumes or vulnerability is different and corresponds to a different modeling choice (e.g., if we insist on mass conservation, then mass should be conserved and interpreting $\mathbf{V}^{-1}$ as vulnerability precludes mass conservation).

10.3 The Choice of Weights

We now turn our attention to the problem of appropriately choosing the weights of the network. We assume that these weights take positive real values and model the fact that some connections may favor transport over others. Indeed, in a physical network, one could expect some of the edges to be larger than others or to have different properties that will enhance or inhibit diffusion. There is no particular added mathematical difficulty to treat this case, and all the definitions of Laplacians seen so far still apply by using a real-valued *weighted adjacency matrix* $\mathbf{A}$ and adding the word *weighted* in front of all other matrices. For instance, the *weighted degree matrix* is the diagonal matrix with the diagonal elements given by the *weighted degrees* $\mathbf{d} = \mathbf{A} \cdot \mathbf{1}$. We note however, that these degrees lose their interpretation as the number of edges attached to a node.

The problem is now how to choose the weights of the adjacency matrix. Based on the available data for the brain connectome, we assume that an edge between distinct nodes i and j is represented by an idealized uniform cylinder built from $n_{ij} \in \mathbb{R}$ fibers and node distance measured along tracts $g_{ij} \in \mathbb{R}_0$. The number of fibers is obtained as an average over multiple data sets. We assume that all material properties are uniform, and we discuss different modeling assumptions for the weights a_{ij}.

Length-free transport: If we assume that the transport process does not depend on the length of the fibers but is limited by other transport mechanisms (e.g., cell-to-cell transport), then a suitable choice for the weighted adjacency matrix

is simply $a_{ij} = n_{ij}$. Then the *standard weighted graph Laplacian* satisfying both mass conservation and Fick's condition is

$$L_{ij} = (d_i \delta_{ij} - n_{ij}), \qquad d_i = \sum_{j=1}^{N} n_{ij}, \tag{10.26}$$

where δ_{ij} is Kronecker's delta and the constant $\rho = 1/\tau$ appearing in (10.4) has the dimension of inverse time. Hence, in this diffusion process, there is no notion of distance and the particular location of different nodes is not taken into account. This process assumes a much slower time scale in the overall diffusive process, which justifies the fact that transport along edges is essentially instantaneous and only modulated by the number of fibers in a given edge and not its length.

Ballistic transport: A second model involves transport penalized by length, so that we have $a_{ij} = n_{ij}/g_{ij}$ and $\mathbf{L} = \mathbf{D} - \mathbf{A}$. The Laplacian

$$L_{ij} = d_i \delta_{ij} - \frac{n_{ij}}{g_{ij}}, \qquad d_i = \sum_{j=1}^{N} \frac{n_{ij}}{g_{ij}}, \tag{10.27}$$

where ρ has now the dimension of a velocity (length over time). Hence, this scaling assumes that the transport motion along the edges scales is a ballistic process. This assumption is suitable to describe nonlinear processes within edges, such as front propagation produced by a nonlinear reaction–diffusion process. In the case of intercellular transport within an axon, if we assume that there is a pool of proteins that can interact autocatalytically with the population of toxic proteins, we expect the development of a front of toxic proteins traveling along the axon, and this choice may be justified.

Diffusive transport: A third model is that transport depends on the inverse of length squared: $a_{ij} = n_{ij}/g_{ij}^2$ and

$$L_{ij} = d_i \delta_{ij} - \frac{n_{ij}}{g_{ij}^2}, \qquad d_i = \sum_{j=1}^{N} \frac{n_{ij}}{g_{ij}^2}, \tag{10.28}$$

which now implies that ρ has the dimension of diffusivity (length squared over time). This is the scaling that we would obtain by directly discretizing the continuous Laplacian over a regular grid. Hence, in the absence of other assumptions about transport, this model seems to be the most natural one (in the sense that the other ones require further modeling assumptions about the transport process taking place within edges or the existence of different time scales in the problem).

The three choices above are part of the general family of weights

$$a_{ij} = \frac{n_{ij}}{g_{ij}^{w}}, \tag{10.29}$$

where $w = 0, 1, 2$.

Distance-based transport: A completely different model solely based on distance has been proposed by Pandya et al. (2019). In this case, the adjacency matrix is not based on connectivity but directly on the Euclidean distance between nodes i and j given by ℓ_{ij};

$$f_{ij} = e^{-\ell_{ij}/\ell} H(e^{-\ell_{ij}/\ell} - T) - \delta_{ij}, \tag{10.30}$$

where g is a typical length scale (1/2 of the mean distance) and H is the Heaviside function. A further step of removing long connections with small values with an (arbitrary) threshold T is used to obtain a sparse matrix. The value chosen by Pandya et al. (2019) is $T = 0.15$, which we adopt for the 83-node connectome to produce a graph with 1552 edges. This model assumes a connection between nodes only if they are sufficiently close. Note that we have subtracted the identity so that $a_{ii} = 0, \ \forall i$.

Normalization: The adjacency matrix is defined up to an arbitrary multiplicative constant (balanced in Eq. (10.2) by the diffusion constant). Therefore, after we obtain the adjacency matrices, we normalize them so that their largest entries are one. The matrices given in the GitHub repository all have this property.

There is no obvious reason to choose one type of weight over another. A conservative modeling approach is to use a ballistic or diffusion assumption (exponent 1 or 2), which can be justified when the diffusion process takes place on a regular lattice.

10.3.1 The Connectome

A brain connectome is obtained by defining nodes of the network to be regions of interest, typically associated with well-known areas from a brain atlas. The edges of this network are defined as the connections between regions. The brain connectome is then modeled as an undirected weighted graph $\mathcal{G}$ with n nodes and m edges. The weights for the adjacency matrix for the simulation are derived from the tractography of diffusion tensor magnetic resonance images corresponding to 418 healthy subjects of the Human Connectome Project (McNab et al., 2013) using the Budapest Reference Connectome v3.0 (Szalkai et al., 2017).

Multiple resolutions with N ranging from 83 to 1015 are available. The values of n_{ij} and g_{ij} are obtained as averages over the 418 brains before the weight is computed. Here, for the purpose of a general introduction, we will use the 83-node network. This network is also easier to match with other types of data (such as volume and clearance) that are typically only available at low resolution. The graph

contains N =83 nodes, 1654 edges, and the weighted adjacency matrix (with ballistic weights) shown in Figure 10.3. Hence, we have

$$a_{ij} = 1/a \frac{\langle n_{ij} \rangle}{\langle g_{ij} \rangle^w}, \quad \langle n_{ij} \rangle = \frac{1}{418} \sum_{k=1}^{418} n_{ij}^{(k)}, \quad \langle g_{ij} \rangle = \frac{1}{418} \sum_{k=1}^{418} g_{ij}^{(k)}, \tag{10.31}$$

where $n_{ij}^{(k)}$ is the number of fibers k in the brain between nodes i and j. The parameter w denotes the choice of transport ($w = 0$ is length- free, $w = 1$ is ballistic, $w = 2$ is diffusion). The parameter a is chosen for normalization so that the maximal entry of $\mathbf{A}$ is one. As defined above, we use the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ where $\mathbf{D} = \mathbf{1} \cdot \mathbf{A}$.

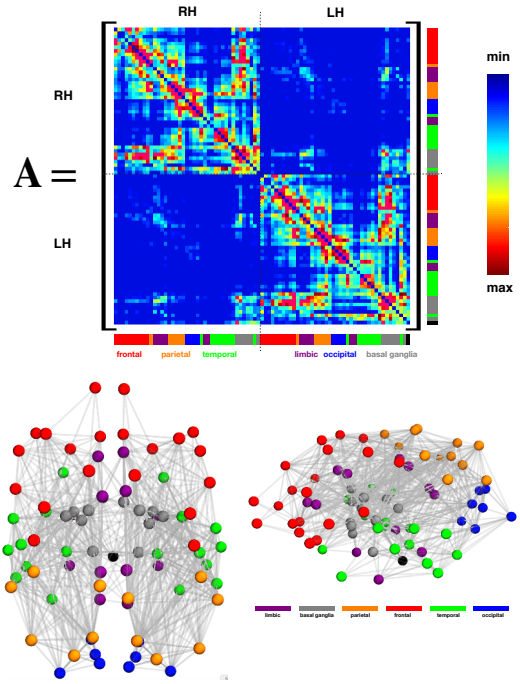


Fig. 10.3: The average weighted-adjacency of 418 brain connectomes with $n = 83$ nodes and ballistic weights, together with the node position in the brain and the lobes to which they belong.

10.4 Implementation

Once a given weight is chosen, we compute the corresponding graph Laplacian $\mathbf{L}$ and the problem then reduces to integrating the discrete Fisher-KPP equation:

$$\dot{p}_i = -\rho \sum_{j=1}^{N} L_{ij} p_j + \alpha p_i (1 - p_i), \quad i = 1, \ldots, N \tag{10.32}$$

$$p_i(0) = \epsilon, \quad i \in \mathcal{S}, \qquad p_i(0) = 0, \quad \text{otherwise}, \tag{10.33}$$

where $\mathcal{S}$ is the set of nodes that are seeded. For instance, it is known that, for Alzheimer's disease, the disease starts in the entorhinal cortex (which corresponds to nodes 27 and 68 in our atlas, and we therefore have $p_{27}(0) = p_{68}(0) = \epsilon$ and $p_i(0) = 0$ everywhere else).

Mathematically, there are two regimes of interest depending on the ratio ρ/α. In the diffusion-dominated regime $\rho/\alpha \gg 1$, the system behaves mostly homogeneously, with the concentration of toxic proteins increasing in all regions uniformly, as found in the propagation of amyloid beta. However, a systematic study of data available from an open database[1] using hierarchical Bayesian parameter inference taught us that the evolution of tau proteins is in the growth-dominated regime ($\rho/\alpha \ll 1$), where each region is invaded in turn from the primary seed. This is the regime of interest for our discussion.

While integrating the system of equations (10.32–10.33) for given parameter values is straightforward, their numerical solutions are given by 83 different curves, as shown in Figure 10.4, which is not particularly useful or directly interpretable.

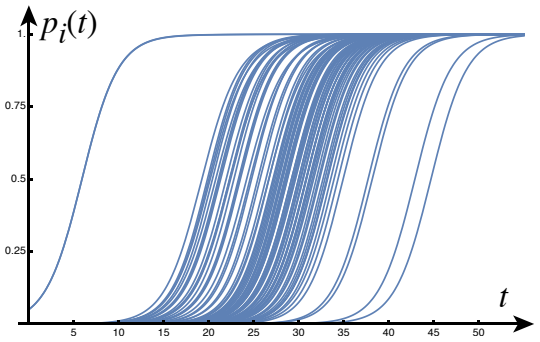


Fig. 10.4: Integration of the discrete Fisher-KPP equations with diffusive weights, $N = 83$, $\alpha = 0.5$/year, $\rho = 0.01$/year, and $\epsilon = 1/20$.

[1] Alzheimer's Disease Neuroimaging Initiative, at https://adni.loni.usc.edu/.

A more informative way to present the results is to average the concentration by Braak regions corresponding to a subset of nodes. Using the basic decomposition in six regions corresponding to different stages of the disease, the Braak region i is defined by a subset $\mathcal{B}_i$ of N_i nodes. For instance, Braak I is defined by the two nodes defining the entorhinal cortex, $\mathcal{B}_{\mathrm{I}} = \{27, 68\}$. We define the averaged concentrations

$$P_i(t) = \frac{1}{N_i} \sum_{j \in \mathcal{B}_i} p_j(t), \qquad i = \mathrm{I}, \ldots, \mathrm{VI}. \tag{10.34}$$

An example of such a simulation is shown in Figure 10.5 together with the different regions. In addition, an asymptotic approximation of the numerical solution is also shown that follows Putra et al. (2023), but it will not be discussed further here.

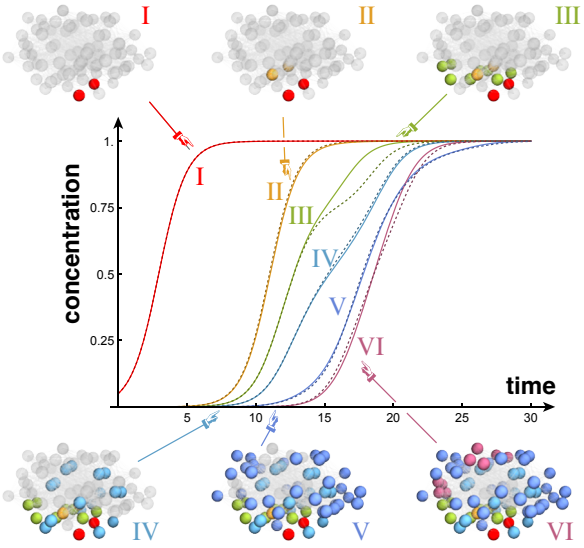


Fig. 10.5: Average concentration of toxic proteins in each Braak region. The solid curves are the numerical solutions, and the dashed curves are their approximations obtained from a nonlinear perturbation expansion (Putra et al., 2023). Initial conditions and parameters are as in Figure 10.4.

We can now compare the different choices of weights for the Laplacian. In Figure 10.6, we run the same model with the same parameters but with different weights. We see that the dynamics are essentially the same, irrespective of the choice of weights.

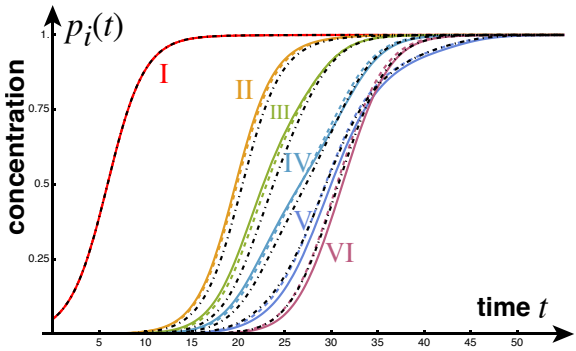


Fig. 10.6: A comparison of the three weight choices (diffusive, solid line; ballistic, dashed line; length-free, dot-dashed and black) shows little effect of the weight. Initial conditions and parameters are as in Figure 10.4.

10.4.1 Limitations and Extensions

Despite its simplicity, the network model (10.32 and 10.33) can be used as a basic starting point to study many different aspects of the disease, quantify new hallmarks of the disease, and test possible mechanisms responsible for the onset and progression of neurodegeneration. In more recent work, we have considered local variations in parameters associated with brain inhomogeneity; derived analytical estimates for the arrival time of the propagating front of toxic proteins through the connectome (Putra et al., 2023); studied the coupling between amyloid beta and tau proteins (Thompson et al., 2020), the role of clearance in the initiation and dynamics of the disease (Brennan et al., 2023), and the interactions between the vasculature and toxic proteins (Ahern et al., 2025); predicted downstream effects such as brain atrophy from tau pathology (Schäfer et al., 2021); and identified topological signatures of the disease in graph space (Goodbrake et al., 2024). We have developed generalized Smoluchowski equations for the microscale aggregation kinetics and used this model to extract bifurcation points from experimental data, showing the critical levels of clearance above which protein aggregation does not occur Brennan and Goriely (2025); Thompson et al. (2021), and further approximated how much toxic concentration a region of the brain can withstand before the natural defenses of brain clearance are overwhelmed and the region becomes saturated in toxic proteins (Brennan et al., 2023). We have also studied the perplexing dynamics in brain activity observed in patients who typically show periods of hyperactivity followed by hypoactivity and a shift in brain wave frequencies. The same model coupled to so-called neuronal mass models for brain activity allowed us to test multiple hypotheses and to conclude that local damage of particular groups of neuronal cells is the most likely mechanism responsible for these observations (Alexandersen et al., 2022).

Limitations of the model come from the low availability of human clinical data to apply to our models, due to ethical experimental constraints, imaging measurement noise, and inter-subject variability. There is always an intrinsic measure of uncertainty in our parameter values, but thankfully this is quantifiable by methods such as Bayesian inference. The availability of accurate mathematical models, however, will lessen the need for human experiments to further our understanding of the disease and simulate therapeutic intervention strategies. Further, exploring the parameter spaces heightens our intuition of disease dynamics from a purely theoretical view. The brain network itself and model parameters could also be personalized to predict patient-specific neurodegeneration.

10.5 Conclusion

At present, the driving factors of Alzheimer's disease are poorly understood, and there are many open questions. Only in the last decade or so has mathematical modeling joined the fight against Alzheimer's disease. Relatively simple reaction–diffusion network models of neurodegeneration reproduce the spatiotemporal spread of toxic proteins observed in imaging studies, thus capturing the governing dynamics of the disease. Network models thus provide a powerful platform for realistic and fast computational experiments that heighten our understanding of key mechanisms of neurodegeneration and enable clinical hypotheses to be explored in silico on human brain graphs. Once network models are developed, the application of computational experiments is not only to understand the mathematical dependencies, but also to address the big questions at the forefront of Alzheimer's research. What interplay exists between the underlying mechanisms driving the disease? Why do we see such distinct spreading patterns in Alzheimer's disease? What sets the time scales for disease progression (approximately 40 years to full invasion)? And, most pressing, is there any way to intervene in the devastating cascade of toxic proteins through the connectome?

Acknowledgements This work was supported by the Engineering and Physical Sciences Research Council grant EP/R020205/1 to Alain Goriely. For the purpose of Open Access, the author has applied a Creative Commons public copyright license to any author accepted manuscript version arising from this submission. Georgia S. Brennan acknowledges the support provided by the EPSRC Centre for Doctoral Training in Industrially Focused Mathematical Modelling (EP/L015803/1).

References

Abdelnour F, Voss HU, Raj A (2014) Network diffusion accurately models the relationship between structural and functional brain connectivity networks. Neuroimage 90:335–347, doi:10.1016/j.neuroimage.2013.12.039

Ahern A, Thompson TB, Oliveri H, Lorthois S, Goriely A (2025) Modelling cerebrovascular pathology and the spread of amyloid beta in alzheimer's disease. Proceedings of the Royal Society A 481(2311):20240548, doi:10.1098/rspa.2024.0548

Alexandersen CG, de Haan W, Bick C, Goriely A (2022) A mechanistic model explains oscillatory slowing and neuronal hyperactivity in Alzheimer's disease. bioRxiv doi:10.1101/2022.06.20.496731

Braak H, Braak E (1991) Neuropathological stageing of Alzheimer-related changes. Acta Neuropathologica 82(4):239–259, doi:10.1007/BF00308809

Brennan GS, Goriely A (2025) A network aggregation model for amyloid-β dynamics and treatment of alzheimer's diseases at the brain scale. Journal of Mathematical Biology 90(2):22, doi:10.1007/s00285-024-02179-5

Brennan GS, Thompson TB, Oliveri H, Rognes ME, Goriely A (2023) The Role of Clearance in Neurodegenerative Diseases. SIAM Journal on Applied Mathematics 0(0):S172–S198, doi:10.1137/22M1487801

Fornari S, Schfer A, Jucker M, Goriely A, Kuhl E (2019) Prion-like spreading of Alzheimer's disease within the brain's connectome. J R Soc Interface 16, doi:10.1098/rsif.2019.0356

Goodbrake C, Beers D, Thompson TB, Harrington HA, Goriely A (2024) Brain chains as topological signatures for Alzheimer's disease. Journal of Applied and Computational Topology 8(5):1257–1298, doi:10.1007/s41468-024-00168-7

Goriely A, Brennan G (2024) gsbrennan/network83: Network83. doi:10.5281/zenodo.13940343

Henderson MX, Cornblath EJ, Darwich A, Zhang B, Brown H, Gathagan RJ, Sandler RM, Bassett DS, Trojanowski JQ, Lee VMY (2019) Spread of α-synuclein pathology through the brain connectome is modulated by selective vulnerability and predicted by network analysis. Nature Neuroscience 22(8):1248–1257, doi:10.1038/s41593-019-0457-5

Henderson MX, Sedor S, McGeary I, Cornblath EJ, Peng C, Riddle DM, Li HL, Zhang B, Brown HJ, Olufemi MF, Bassett DS, Trojanowski JQ, Lee VMY (2020) Glucocerebrosidase Activity Modulates Neuronal Susceptibility to Pathological α-Synuclein Insult. Neuron 105(5):822–836.e7, doi:10.1016/j.neuron.2019.12.004

Jucker M, Walker LC (2013) Self-propagation of pathogenic protein aggregates in neurodegenerative diseases. Nature 501(7465):45–51, doi:10.1038/nature12481

Masuda N, Porter MA, Lambiotte R (2017) Random walks and diffusion on networks. Physics Reports 716-717:1–58, doi:10.1016/j.physrep.2017.07.007

McNab JA, Edlow BL, Witzel T, Huang SY, Bhat H, Heberlein K, Feiweier T, Liu K, Keil B, Cohen-Adad J, Tisdall MD, Folkerth RD, Kinney HC, Wald LL (2013) The Human Connectome Project and beyond: Initial applications of 300mT/m gradients. NeuroImage 80:234–245, doi:10.1016/j.neuroimage.2013.05.074, mapping the Connectome

Pandya S, Zeighami Y, Freeze B, Dadar M, Collins D, Dagher A, Raj A (2019) Predictive model of spread of Parkinson's pathology using network diffusion. NeuroImage 192:178–194, doi:10.1016/j.neuroimage.2019.03.001

Putra P, Oliveri H, Thompson T, Goriely A (2023) Front Propagation and Arrival Times in Networks with Application to Neurodegenerative Diseases. SIAM Journal on Applied Mathematics 83(1):194–224, doi:10.1137/21M1467547

Raj A, Kuceyeski A, Weiner M (2012) A Network Diffusion Model of Disease Progression in Dementia. Neuron 73(6):1204–1215, doi:10.1016/j.neuron.2011.12.040

Schäfer A, Chaggar P, Thompson TB, Goriely A, Kuhl E (2021) Predicting brain atrophy from tau pathology: a summary of clinical findings and their translation into personalized models. Brain Multiphysics 2:100039, doi:10.1016/j.brain.2021.100039

Szalkai B, Kerepesi C, Varga B, Grolmusz V (2017) Parameterizable consensus connectomes from the Human Connectome Project: the Budapest Reference Connectome Server v3.0. Cognitive Neurodynamics 11(1):113–116, doi:10.1007/s11571-016-9407-z

Thompson TB, Chaggar P, Kuhl E, Goriely A, for the Alzheimers Disease Neuroimaging Initiative (2020) Protein-protein interactions in neurodegenerative diseases: A conspiracy theory. PLOS Computational Biology 16(10):1–41, doi:10.1371/journal.pcbi.1008267

Thompson TB, Meisl G, Knowles TPJ, Goriely A (2021) The role of clearance mechanisms in the kinetics of pathological protein aggregation involved in neurodegenerative diseases. The Journal of Chemical Physics 154(12):125101, doi:10.1063/5.0031650

Weickenmeier J, Kuhl E, Goriely A (2018) Multiphysics of Prionlike Diseases: Progression and Atrophy. Phys Rev Lett 121:158101, doi:10.1103/PhysRevLett.121.158101