Neuron modeling, Bloch-Torrey equation, and their application to brain microstructure estimation using diffusion MRI

Chengran Fang

université
PARIS-SACLAY

Neuron modeling, Bloch-Torrey equation, and their application to brain microstructure estimation using diffusion MRI

Modélisation des neurones, équation de Bloch-Torrey et leur application à l'estimation de la microstructure du cerveau par IRM de diffusion

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 580, Sciences et Technologies de l'Information et de la Communication (STIC)
Spécialité de doctorat : Informatique mathématique
Graduate School : Informatique et sciences du numérique
Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche **Inria Saclay - Île-de-France** (Université Paris-Saclay, Inria), sous la direction de **Jing-Rebecca LI**, Directrice de recherche, et le co-encadrement de **Demian WASSERMANN**, Directeur de recherche.

Thèse soutenue à Paris-Saclay, le 2 février 2023, par

Chengran FANG

Composition du jury

Membres du jury avec voix délibérative

Clair POIGNARD Directeur de Recherche, Inria Bordeaux - Sud Ouest	Président & Rapporteur
Denis GREBENKOV Directeur de Recherche, École Polytechnique	Rapporteur & Examinateur
Marco PALOMBO Professeur Assistant, Cardiff University	Examinateur
Ileana JELESCU Professeur Assistant, Lausanne University Hospital and University of Lausanne	Examinatrice

THESE DE DOCTORAT
NNT : 2023UPASG010

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Modélisation des neurones, équation de Bloch-Torrey et leur application à l'estimation de la microstructure du cerveau par IRM de diffusion

Mots clés: IRM de diffusion, modélisation des neurones, équation de Bloch-Torrey, simulation numérique, imagerie de la microstructure du cerveau, apprentissage automatique

Résumé: L'estimation non invasive de la microstructure du cerveau, composée d'un très grand nombre de neurites, de somas et de cellules gliales, est essentielle pour la neuro-imagerie. L'IRM de diffusion (IRMd) est une technique prometteuse pour sonder les propriétés microstructurelles du cerveau en dessous de la résolution spatiale des scanners IRM. En raison de la complexité structurelle du tissu cérébral et du mécanisme complexe de l'IRM de diffusion, l'estimation de la microstructure in vivo est un défi. Les méthodes existantes utilisent généralement des géométries simplifiées, en particulier des sphères et des bâtons, pour modéliser les structures neuronales et obtenir des expressions analytiques des signaux intracellulaires. La validité des hypothèses faites par ces méthodes reste indéterminée. Cette thèse vise à faciliter l'estimation de la microstructure cérébrale par simulation en remplaçant les géométries simplifiées par des modèles réalistes de géométrie des neurones et les expressions analytiques des signaux intracellulaires par des simulations d'IRM de diffusion. Combinées à des modèles géométriques de neurones précis, les simulations numériques d'IRMd peuvent donner des signaux intracellulaires précis et incorporer de manière transparente les effets résultant, par exemple, de l'ondulation des neurites ou de l'échange d'eau entre le soma et les neurites.

Malgré ces avantages, les simulations d'IRMd n'ont pas été largement adoptées en raison des difficultés à construire des modèles géométriques réalistes, du coût de calcul élevé des simulations d'IRMd et de la difficulté à approximer les mappings implicites entre les signaux d'IRMd et les propriétés de la microstructure. Cette thèse aborde les problèmes mentionnés ci-dessus en apportant quatre contributions.

Premièrement, nous développons un générateur de maillage de neurones open-source de haute performance et mettons à la disposition du public plus d'un millier de maillages cellulaires réalistes. Le générateur de maillage de neurones, swc2mesh, peut convertir automatiquement et de manière robuste des données précieuses de traçage de neurones (neuron tracing) en maillages de neurones réalistes. Nous avons soigneusement conçu le générateur pour maintenir un bon équilibre entre la qualité et la taille du maillage. Une base de données de maillage de neurones, NeuronSet, qui contient 1213 maillages cellulaires prêts pour la simulation et leurs mesures neuroanatomiques, a été construite à l'aide du générateur de maillage.

Deuxièmement, nous avons augmenté l'efficacité de calcul de la méthode de formalisme matriciel numérique en accélérant l'algorithme d'eigendecomposition et en exploitant le calcul GPU. La vitesse a été multipliée par dix. Avec une précision similaire, le formalisme matriciel numérique optimisé est 20 fois plus rapide que la méthode FEM et 65 fois plus rapide qu'une méthode Monte-Carlo basée sur le GPU. En effectuant des simulations sur des maillages de neurones réalistes, nous avons étudié l'effet de l'échange d'eau entre les somas et les neurites, ainsi que la relation entre la taille du soma et les signaux.

Nous avons ensuite mis en œuvre une nouvelle méthode de simulation qui fournit une représentation de type Fourier des signaux IRMd. Cette méthode a été dérivée théoriquement et mise en œuvre numériquement. Nous avons validé la convergence de la méthode et montré que le comportement de l'erreur est conforme à notre analyse de l'erreur.

Enfin, nous proposons un cadre fondé sur la simulation pour une modélisation géométrique précise de l'imagerie de la microstructure du cerveau. En exploitant les puissantes capacités de modélisation et de calcul, nous avons construit une base de données synthétique contenant les signaux IRMd et les paramètres de microstructure de 1,4 million de voxels cérébraux artificiels. Nous avons montré que cette base de données permet d'approximer les mappings sous-jacents des signaux d'IRMd aux fractions de volume et de surface en utilisant des réseaux de neurones artificiels.

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Title: Neuron modeling, Bloch-Torrey equation, and their application to brain microstructure estimation using diffusion MRI

Keywords: diffusion MRI, neuron modeling, Bloch-Torrey equation, numerical simulation, brain microstructure imaging, machine learning

Abstract: Non-invasively estimating brain microstructure that consists of a very large number of neurites, somas, and glial cells is essential for future neuroimaging. Diffusion MRI (dMRI) is a promising technique to probe brain microstructural properties below the spatial resolution of MRI scanners. Due to the structural complexity of brain tissue and the intricate diffusion MRI mechanism, in vivo microstructure estimation is challenging. Existing methods typically use simplified geometries, particularly spheres, and sticks, to model neuronal structures and to obtain analytical expressions of intracellular signals. The validity of the assumptions made by these methods remains undetermined. This thesis aims to facilitate simulation-driven brain microstructure estimation by replacing simplified geometries with realistic neuron geometry models and the analytical intracellular signal expressions with diffusion MRI simulations. Combined with accurate neuron geometry models, numerical dMRI simulations can give accurate intracellular signals and seamlessly incorporate effects arising from, for instance, neurite undulation or water exchange between soma and neurites.

Despite these advantages, dMRI simulations have not been widely adopted due to the difficulties in constructing realistic numerical phantoms, the high computational cost of dMRI simulations, and the difficulty in approximating the implicit mappings between dMRI signals and microstructure properties. This thesis addresses the above problems by making four contributions.

First, we develop a high-performance open-source neuron mesh generator and make publicly available over a thousand realistic cellular meshes. The neuron mesh generator, swc2mesh, can automatically and robustly convert valuable neuron tracing data into realistic neuron meshes. We have carefully designed the generator to maintain a good balance between mesh quality and size. A neuron mesh database, NeuronSet, which contains 1213 simulation-ready cell meshes and their neuroanatomical measurements, was built using the mesh generator. These meshes served as the basis for further research.

Second, we increased the computational efficiency of the numerical matrix formalism method by accelerating the eigendecomposition algorithm and exploiting GPU computing. The speed was increased tenfold. With similar accuracy, the optimized numerical matrix formalism is 20 times faster than the FEM method and 65 times faster than a GPU-based Monte Carlo method. By performing simulations on realistic neuron meshes, we investigated the effect of water exchange between somas and neurites, and the relationship between soma size and signals.

We then implemented a new simulation method that provides a Fourier-like representation of the dMRI signals. This method was derived theoretically and implemented numerically. We validated the convergence of the method and showed that the error behavior is consistent with our error analysis.

Finally, we propose a simulation-driven supervised learning framework to estimate brain microstructure using diffusion MRI. By exploiting the powerful modeling and computational capabilities that are mentioned above, we have built a synthetic database containing the dMRI signals and microstructure parameters of 1.4 million artificial brain voxels. We have shown that this database can help approximate the underlying mappings of the dMRI signals to volume and surface fractions using artificial neural networks.

Abstract

Non-invasively estimating brain microstructure that consists of a very large number of neurites, somas, and glial cells is essential for future neuroimaging. Diffusion MRI (dMRI) is a promising technique to probe brain microstructural properties below the spatial resolution of MRI scanners. Due to the structural complexity of brain tissue and the intricate diffusion MRI mechanism, in vivo microstructure estimation is challenging. Existing methods typically use simplified geometries, particularly spheres, and sticks, to model neuronal structures and to obtain analytical expressions of intracellular signals. The validity of the assumptions made by these methods remains undetermined. This thesis aims to facilitate simulation-driven brain microstructure estimation by replacing simplified geometries with realistic neuron geometry models and the analytical intracellular signal expressions with diffusion MRI simulations. Combined with accurate neuron geometry models, numerical dMRI simulations can give accurate intracellular signals and seamlessly incorporate effects arising from, for instance, neurite undulation or water exchange between soma and neurites.

Despite these advantages, dMRI simulations have not been widely adopted due to the difficulties in constructing realistic numerical phantoms, the high computational cost of dMRI simulations, and the difficulty in approximating the implicit mappings between dMRI signals and microstructure properties. This thesis addresses the above problems by making four contributions.

First, we develop a high-performance open-source neuron mesh generator and make publicly available over a thousand realistic cellular meshes. The neuron mesh generator, swc2mesh, can automatically and robustly convert valuable neuron tracing data into realistic neuron meshes. We have carefully designed the generator to maintain a good balance between mesh quality and size. A neuron mesh database, NeuronSet, which contains 1213 simulation-ready cell meshes and their neuroanatomical measurements, was built using the mesh generator. These meshes served as the basis for further research.

Second, we increased the computational efficiency of the numerical matrix formalism method by accelerating the eigendecomposition algorithm and exploiting GPU computing. The speed was increased tenfold. With similar ac-

curacy, the optimized numerical matrix formalism is 20 times faster than the FEM method and 65 times faster than a GPU-based Monte Carlo method. By performing simulations on realistic neuron meshes, we investigated the effect of water exchange between somas and neurites, and the relationship between soma size and signals.

We then implemented a new simulation method that provides a Fourier-like representation of the dMRI signals. This method was derived theoretically and implemented numerically. We validated the convergence of the method and showed that the error behavior is consistent with our error analysis.

Finally, we propose a simulation-driven supervised learning framework to estimate brain microstructure using diffusion MRI. By exploiting the powerful modeling and computational capabilities that are mentioned above, we have built a synthetic database containing the dMRI signals and microstructure parameters of 1.4 million artificial brain voxels. We have shown that this database can help approximate the underlying mappings of the dMRI signals to volume and surface fractions using artificial neural networks.

In summary, this thesis proposes a novel approach to estimating brain microstructure using realistic neuron geometric models and dMRI simulations. We developed a high-performance neuron mesh generator, optimized the numerical matrix formalism method to speed up the simulations, and proposed a method to approximate the implicit mapping between dMRI signals and microstructural properties using artificial neural networks. We hope that these contributions will bring the significance of geometric modeling and simulation to the attention of the community and lead to the design of more advanced microstructure imaging methods to explore the mysteries of the human brain.

Résumé

L'estimation non invasive de la microstructure du cerveau, composée d'un très grand nombre de neurites, de somas et de cellules gliales, est essentielle pour la neuro-imagerie. L'IRM de diffusion (IRMd) est une technique prometteuse pour sonder les propriétés microstructurelles du cerveau en dessous de la résolution spatiale des scanners IRM. En raison de la complexité structurelle du tissu cérébral et du mécanisme complexe de l'IRM de diffusion, l'estimation de la microstructure in vivo est un défi. Les méthodes existantes utilisent généralement des géométries simplifiées, en particulier des sphères et des bâtons, pour modéliser les structures neuronales et obtenir des expressions analytiques des signaux intracellulaires. La validité des hypothèses faites par ces méthodes reste indéterminée. Cette thèse vise à faciliter l'estimation de la microstructure cérébrale par simulation en remplaçant les géométries simplifiées par des modèles réalistes de géométrie des neurones et les expressions analytiques des signaux intracellulaires par des simulations d'IRM de diffusion. Combinées à des modèles géométriques de neurones précis, les simulations numériques d'IRMd peuvent donner des signaux intracellulaires précis et incorporer de manière transparente les effets résultant, par exemple, de l'ondulation des neurites ou de l'échange d'eau entre le soma et les neurites.

Malgré ces avantages, les simulations d'IRMd n'ont pas été largement adoptées en raison des difficultés à construire des modèles géométriques réalistes, du coût de calcul élevé des simulations d'IRMd et de la difficulté à approximer les mappings implicites entre les signaux d'IRMd et les propriétés de la microstructure. Cette thèse aborde les problèmes mentionnés ci-dessus en apportant quatre contributions.

Premièrement, nous développons un générateur de maillage de neurones open-source de haute performance et mettons à la disposition du public plus d'un millier de maillages cellulaires réalistes. Le générateur de maillage de neurones, swc2mesh, peut convertir automatiquement et de manière robuste des données précieuses de traçage de neurones (neuron tracing) en maillages de neurones réalistes. Nous avons soigneusement conçu le générateur pour maintenir un bon équilibre entre la qualité et la taille du maillage. Une base de don-

nées de maillage de neurones, NeuronSet, qui contient 1213 maillages cellulaires prêts pour la simulation et leurs mesures neuroanatomiques, a été construite à l'aide du générateur de maillage. Ces maillages ont servi de base aux recherches ultérieures.

Deuxièmement, nous avons augmenté l'efficacité de calcul de la méthode de formalisme matriciel numérique en accélérant l'algorithme d'eigendecomposition et en exploitant le calcul GPU. La vitesse a été multipliée par dix. Avec une précision similaire, le formalisme matriciel numérique optimisé est 20 fois plus rapide que la méthode FEM et 65 fois plus rapide qu'une méthode Monte-Carlo basée sur le GPU. En effectuant des simulations sur des maillages de neurones réalistes, nous avons étudié l'effet de l'échange d'eau entre les somas et les neurites, ainsi que la relation entre la taille du soma et les signaux.

Nous avons ensuite mis en œuvre une nouvelle méthode de simulation qui fournit une représentation de type Fourier des signaux IRMd. Cette méthode a été dérivée théoriquement et mise en œuvre numériquement. Nous avons validé la convergence de la méthode et montré que le comportement de l'erreur est conforme à notre analyse de l'erreur.

Enfin, nous proposons un cadre fondé sur la simulation pour une modélisation géométrique précise de l'imagerie de la microstructure du cerveau. En exploitant les puissantes capacités de modélisation et de calcul, nous avons construit une base de données synthétique contenant les signaux IRMd et les paramètres de microstructure de 1,4 million de voxels cérébraux artificiels. Nous avons montré que cette base de données permet d'approximer les mappings sous-jacents des signaux d'IRMd aux fractions de volume et de surface en utilisant des réseaux de neurones artificiels.

En résumé, cette thèse propose une approche novatrice pour l'estimation de la microstructure cérébrale en utilisant des modèles géométriques de neurones réalistes et des simulations d'IRMd. Nous avons développé un générateur de maillage de neurones open-source de haute performance, optimisé la méthode du formalisme matriciel numérique pour accélérer les simulations, proposé une méthode pour approximer le mapping implicite entre les signaux IRMd et les propriétés microstructurelles en utilisant des réseaux de neurones artificiels.

Acknowledgement

First, I would like to thank my PhD supervisors, Jing-Rebecca Li and Demian Wassermann. I am thankful for their guidance and mentorship throughout my three years of doctoral study. I also want to thank the committee members, including Clair Poignard, Denis Grebenkov, Marco Palombo, Ileana Jelescu, and Dmitry Novikov, for their feedback and suggestions for my thesis. Their advice and criticism have helped me improve the quality of my work.

Furthermore, I would like to acknowledge the help from my friends in the IDE-FIX and MIND teams, especially Marcella, Maeliss, Thomas, Gaston, Louis, and Raphael. I will always cherish the memories of working alongside them during my doctoral studies and the support they provided during my defense.

I want to give special thanks to my friends who gave specific help to my thesis. In particular, I would like to thank Wei Li for his constructive advice in the second chapter. I am also grateful to Syver for refactorizing the SpinDoctor Toolbox, which facilitated my research work. Zheyi's active assistance in optimizing the MF solver deserves my special appreciation. Additionally, I would like to thank Chuhan Wang, Zhihao Ding, Yang Zhang, Jiaqi Li, and Xiaoli Liu for sharing their research experience in fluid mechanics, computer science, and mathematics. Their insights have been invaluable in shaping my research perspective.

In addition, I want to thank my friends, namely, Xiaoxiao Zhang, Xingyu Chen, Haotian Yang, Xiaoning Meng, Xiaolei Chen, Xu Liu, Jia Guo, and Yining Huang. With them, my stay in Palaiseau was memorable and pleasant.

Finally, I want to express my deepest gratitude to my parents and grandparents. Due to the COVID epidemic, I have not been able to reunite with them for the past three years. But they always give me understanding and support. Their support has helped me overcome many difficulties.

Contributions

List of journal articles

1. **Chengran Fang**, Van-Dang Nguyen, Demian Wassermann, Jing-Rebecca Li, *Diffusion MRI simulation of realistic neurons with SpinDoctor and the Neuron Module*, NeuroImage 222 (2020): 117198.
2. **Chengran Fang**, Demian Wassermann, Jing-Rebecca Li, *Fourier representation of the diffusion MRI signal using layer potentials*, SIAM Journal on Applied Mathematics 83, no. 1 (2023): 99-121.

List of conference abstracts

1. Van-Dang Nguyen, **Chengran Fang**, Demian Wassermann, Jing-Rebecca Li, *Diffusion MRI simulation of realistic neurons with SpinDoctor*, International Society of Magnetic Resonance in Medicine, 2020.
2. **Chengran Fang**, Demian Wassermann, Jing-Rebecca Li, *Fourier Potential method*, International Society of Magnetic Resonance in Medicine, 2022.
3. **Chengran Fang**, Demian Wassermann, Jing-Rebecca Li, *Simulation-driven machine learning framework to estimate brain microstructure using diffusion MRI*, ISMRM 2022 Workshop on Diffusion.

List of open-source projects

1. Author of *swc2mesh*: https://github.com/fachra/swc2mesh
2. Author of *NeuronSet*: https://github.com/fachra/NeuronSet
3. Author of *FourierPotential*: https://github.com/fachra/FourierPotential
4. Contributor of *SpinDoctor*: https://github.com/SpinDoctorMRI/SpinDoctor

Abbreviations

aFI Anterior frontal insula

ACC Anterior cingulate cortex

ADC Apparent diffusion coefficient

AMICO Accelerated Microstructure Imaging via Convex Optimization

BT equation Bloch-Torrey partial differential equation

CPU Central processing unit

CSF Cerebrospinal fluid

d-PGSE Double pulsed-gradient spin echo

DMRI, dMRI Diffusion magnetic resonance imaging

DTI Diffusion tensor imaging

ECS Extracellular space

EPI Echo planar imaging

FA Fractional anisotropy

FAQ Frequently asked question

FE Finite element

FEM Finite element method

FOV Field of view

FPM Finite potential method

GELU Gaussian error linear unit

GM Gray matter

GPA Gaussian phase assumption

GPU Graphics processing unit

ICC Intracellular compartment

MF Matrix formalism

ML Machine learning

MLP Multilayer perceptron

MR Magnetic resonance

MRI Magnetic resonance imaging

MSE Mean squared error

NMR Nuclear magnetic resonance

ODE Ordinary differential equation

ODF Orientation distribution function

OGSE Oscillating gradient spin echo

PGSE Pulsed-gradient spin echo

PLY Polygon file format

RA Relative anisotropy

ReLU Rectified linear unit

RF Radiofrequency

SANDI Soma and neurite density imaging

SNR Signal-to-noise ratio

STL Standard triangle language (file format)

SWC Initials of the last names of E.W. Stockley, H.V. Wheal, and H.M. Cole (file format)

WM White matter

Contents

Chapter 1

Introduction

1.1 Introductin to human brain

The brain is the most important and complex human organ. It controls most human activities, including body movement, vision, memory, emotion, reasoning, and decision-making. The human brain consists of three major parts: the brainstem, cerebellum, and cerebrum, as shown in fig. 1.1(a) [1]. The brainstem connects the cerebrum with the spinal cord and controls vital functions like breathing, heart rate, and sleep. The cerebellum lies at the posterior of the brain, regulating body movement. The cerebrum is the largest part of the human brain, which supports most brain functions. It is divided into left and right hemispheres. The cerebrum's outer layer, also known as the cerebral cortex, is folded to increase the brain's surface area. The ridges are called gyri, and the grooves are called sulci. Large sulci are often called fissures. For example, the longitudinal fissure is the groove that splits the two cerebral hemispheres, as illustrated in fig. 1.1(b).

Each cerebral hemisphere can be further divided into four lobes, i.e., the frontal, parietal, occipital, and temporal lobes. Although most brain functions require the synergy of many different regions across the entire brain, it is still true that each lobe is primarily responsible for certain functions. For example, the frontal lobes control high-level cognitive functions such as attention, memory, and language [2]. It also contains the primary motor cortex located in the precentral gyrus (marked in fig. 1.1(b)), which is responsible for voluntary movements [3]. The central sulcus separates the frontal lobes and the parietal lobes. The postcentral gyrus in the parietal lobe is next to the central sulcus. It contains the primary somatosensory cortex responsible for processing sensory inputs, such as touch, temperature, body position, and pain [4]. The lists of brain functions are not exhaustive. The study of the functions of brain regions is still

a research priority.

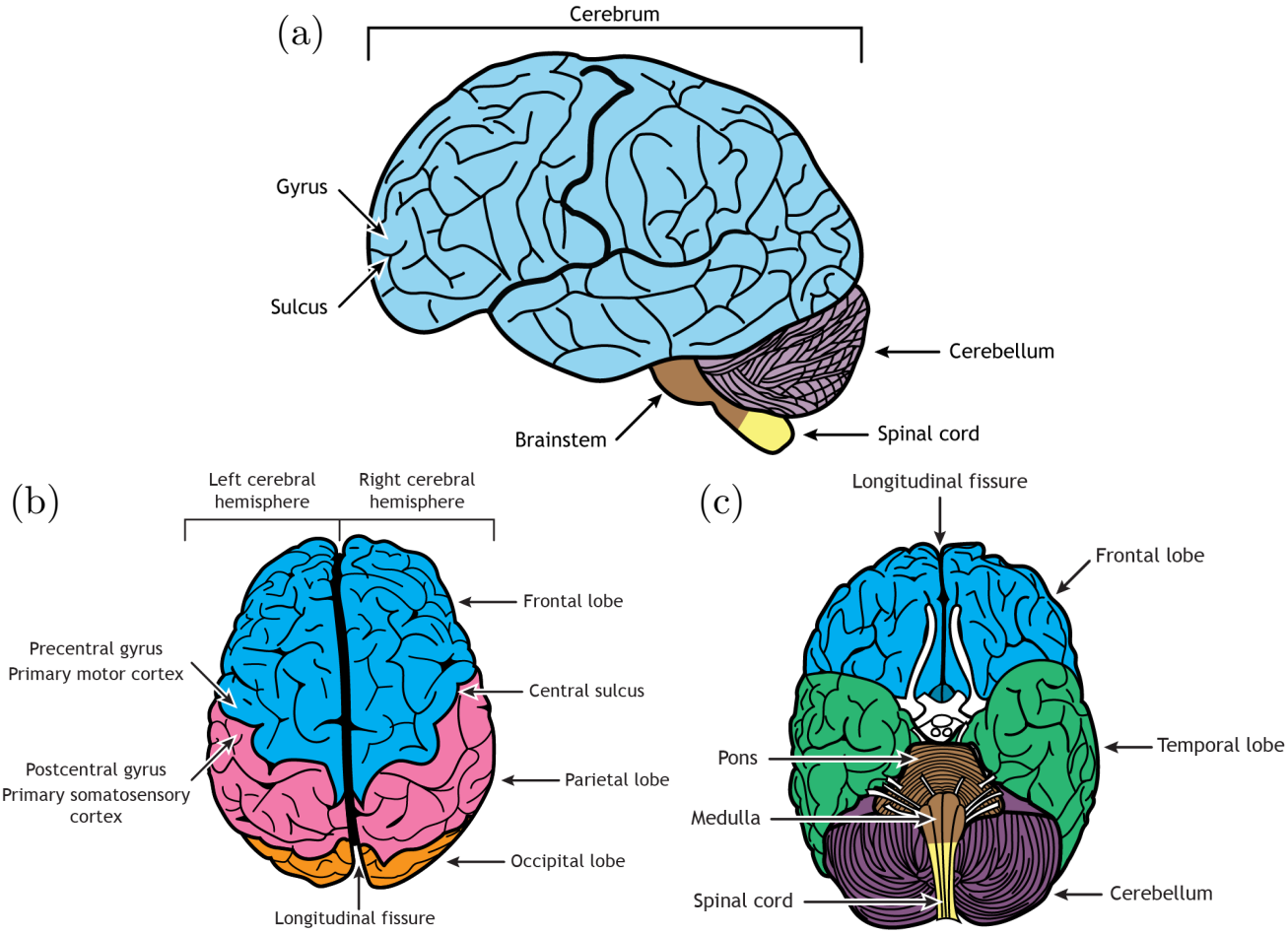


Figure 1.1: Different views of the human brain. **(a)** The lateral view of the brain. The cerebrum, the cerebellum, and the brainstem are indicated by the cyan, violet, and brown regions, respectively. **(b)** The dorsal view of the brain. Three cerebral lobes can be seen in this view. The cerebrum is divided into two hemispheres by the longitudinal fissure. **(c)** The inferior view of the brain. The temporal lobe and the two hemispheres of the cerebellum are visible. Pons and medulla are two major parts of the brainstem. Images from the book of Casey Henley [1], distributed under the CC BY-NC-SA 4.0 International license.

The functional units of the brain are cells. The two main types of cells in the brain are neurons and glial cells (glia) [5]. A neuron consists of two compartments: a cell body, also known as a soma, and neuronal processes called neurites, as shown in fig. 1.2. The brain functions rely on the exchange of information between neurons through neurites. There are two types of neurites:

dendrites and axons. Dendrites project from the cell body and receive information from other neurons. An axon, or nerve fiber, is a thin projection extending from a soma. The function of the axon is to transmit information to other cells. A single neuron may contact 30,000-60,000 cells, and the axon length ranges from $< 1\ mm$ to $1\ m$ [6]. Neurons are supported by glial cells. The oligodendrocytes, a class of glial cells, can wrap around the axon to form the insulating myelin sheath that facilitates the propagation of electrical signals [7].

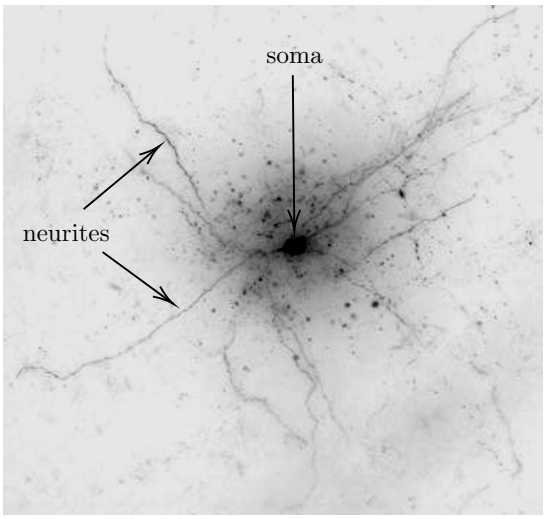


Figure 1.2: A microscopic image of a human neuron from the Allen Institute Cell Types database [8]. The neuron is stained by biocytin. Most visible neurites are dendrites. Axons are less visible because of their small radii. In the database, the neuron ID is 643582610, and its donor is H17.06.013.

The human brain consists of 86 billion neurons and a similar amount of glial cells [9]. The extracellular fluid bathing the cells and spinal cord is called cerebrospinal fluid (CSF) [10]. Neurons, glial cells, and extracellular fluid are the main components of brain tissue. According to the composition, brain tissue can be divided into white and gray matter (WM and GM). Figure 1.3 illustrates the brain regions where the two types of tissues are located. White matter regions are brighter than gray matter because it has a high density of myelinated axons. The lipid-rich myelin sheath renders the nerve fibers white [11, p. 129]. Gray matter regions mainly comprise somas and dendrites. Different gray matter regions are connected by axons passing through white matter.

To visually demonstrate their differences, we present the microscopic images of gray and white matter tissue in fig. 1.4. The 3D geometrical models of some selected cellular components are plotted to show the tissue microstructure. The images and geometrical models are shared by a state-of-the-art brain

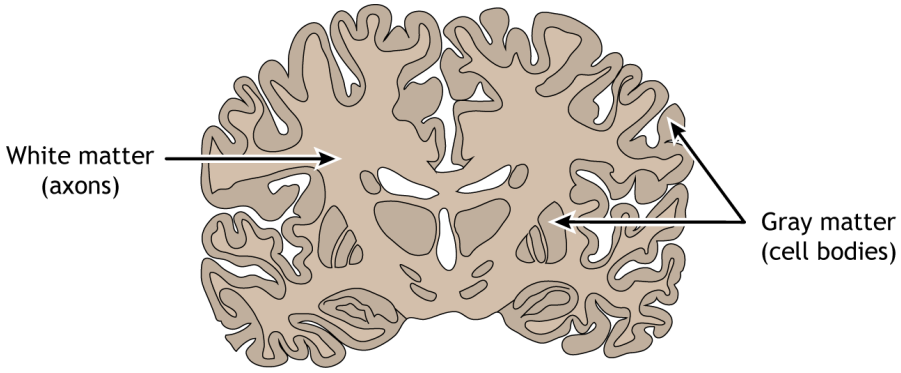


Figure 1.3: An illustration of gray matter and white matter regions. Image from the book of Casey Henley [1], distributed under the CC BY-NC-SA 4.0 International license.

histology study[1] [12], which cut a stained human brain sample with a volume of $1\ mm^3$ into more than 5000 slices. An electron microscope then digitalizes each slice. The 3D geometrical models of somas, neurites, and glial cells are constructed based on the electron microscopic image stack. The gray matter image in fig. 1.4(a) contains somas and numerous dendrites. The white matter image in fig. 1.4(c) is composed of a large number of myelinated axons. For clarity, we only show the 3D geometrical models of the colorized cellular components in fig. 1.4(b) and fig. 1.4(d). Even with just a few cellular components, it is clear that the cellular organization in the brain is very complex. Neurons and glial cells are intertwined, with neurites pointing in all directions and the extracellular fluid filling the entire space, making up the intricate microstructure of the human brain.

Knowledge about the brain microstructure can help understand brain functions and neurological disorders. For example, finding the connectivities between different gray matter areas allows us to understand how different brain regions work together. Moreover, numerous neurological disorders are linked to changes in brain microstructure. Multiple sclerosis, for instance, is caused by damage to the myelin sheath of axons [13]. The progressive loss of structure or function of neurons causes Alzheimer's disease, frontotemporal dementia, Parkinson's disease, etc. [14]. Other diseases, such as strokes and brain tumors, also alter the brain microstructure [15, 16]. Therefore, measuring parameters that quantify brain microstructure is helpful in the diagnosis and treatment of neurological disorders.

Histological methods based on microscopic imaging can give precise microstructural measurements. However, they are invasive and costly. For example, in the brain histology study mentioned above [12], the electron microscopic images of the $1\ mm^3$ brain tissue occupy 1,400 terabytes. In addition, the brain

[1] `https://h01-release.storage.googleapis.com/explore.html`

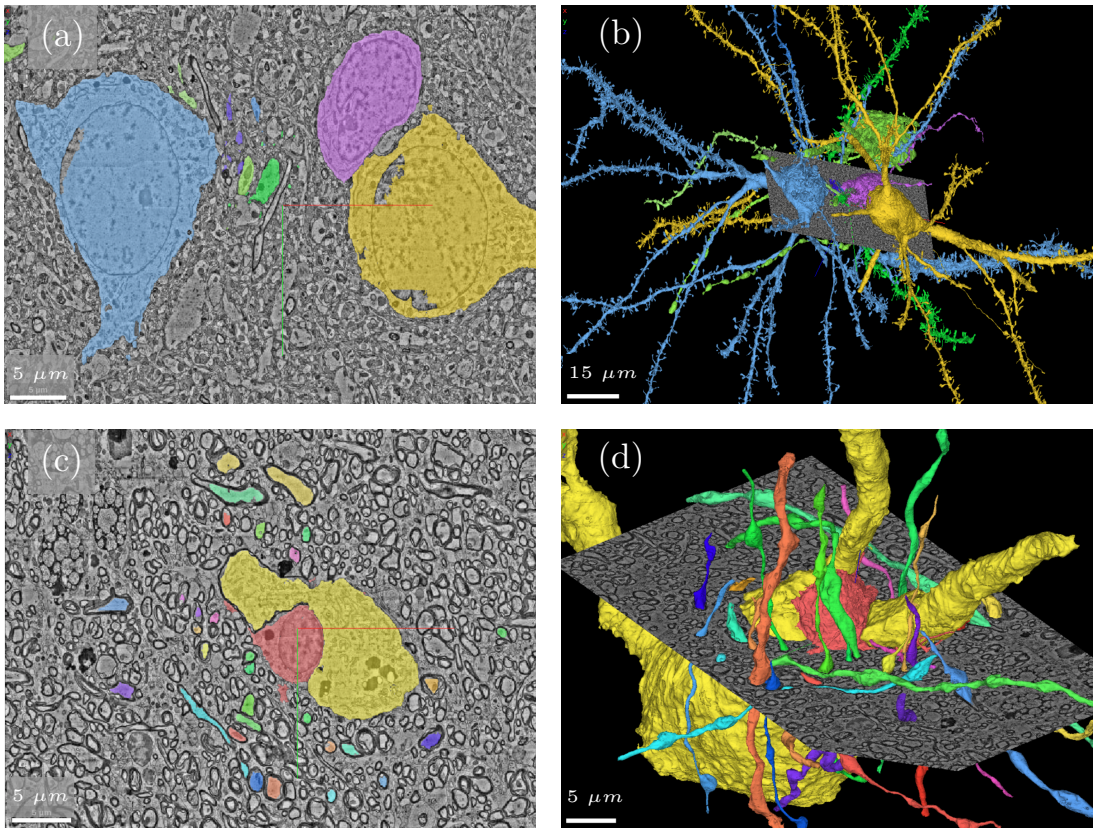


Figure 1.4: Microscopic images and 3D geometrical models of gray matter and white matter tissue. The histological measurements are shared by the work of Shapson-Coe et al. [12]. **(a)** a microscopic image of gray matter tissue. It contains somas, dendrites, and unmyelinated axons. **(b)** the 3D geometrical models of the colorized cellular compartments in (a). The gray plane represents the cross-section where (a) is located. **(c)** a microscopic image of white matter tissue. It mainly contains myelinated axons. **(d)** the 3D geometrical models of the colorized cellular compartments in (c). The gray plane represents the cross-section where (c) is located.

tissue integrity is completely destroyed. These methods can only be performed post-mortem.

Diffusion magnetic resonance imaging (diffusion MRI, dMRI) is a way to probe the diffusive motion of water protons by applying magnetic field gradient to brain tissue [17–19]. Like conventional MRI, dMRI yields a series of grayscale maps representing "virtual" brain slices, as shown in fig. 1.5. Each pixel in the grayscale maps corresponds to a signal from a brain voxel (a cubic region in the human brain). The typical size of a brain voxel is $1\,mm^3$. Because the movement of water protons is restricted by, for example, cell membranes [20], brain voxels

with varying microstructures generate different signals [11, 21], thus forming the contrast in the grayscale maps. A simplified and qualitative explanation for the dMRI contrast is that, in a grayscale map, pixels are brighter (darker) in areas with strong (weak) restrictions on water diffusion. Diffusion MRI settings, such as the intensity of the magnetic field gradient and the diffusion time t_d, also affect the contrast of the grayscale maps. We will explain the dMRI settings in sections 1.2.2 and 3.1.1. Due to the sensitivity to structures below the spatial resolution of MRI scanners, dMRI is a compelling method to probe brain microstructure properties non-invasively.

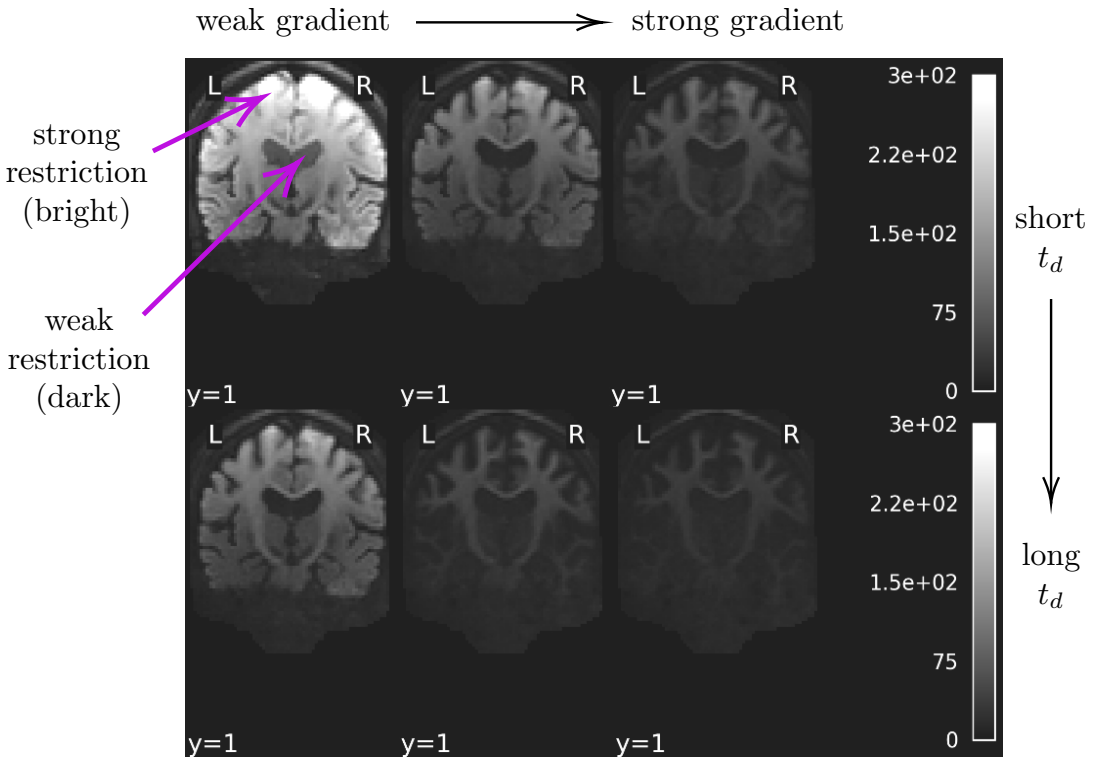


Figure 1.5: Six dMRI grayscale maps. In general, pixels are brighter (darker) in areas with strong (weak) restrictions on water diffusion. Diffusion MRI settings, such as the intensity of the magnetic field gradient and the diffusion time t_d, also affect the contrast of the grayscale maps. The MGH CDMD dataset [22] provides the experimental data.

The above explanation about diffusion MRI is simplified. The following section will give a more comprehensive theory about the dMRI signal formation mechanism. One can design methods for probing brain microstructure based on the understanding of magnetic resonance physics. Section 1.3 presents several existing methods for brain imaging with dMRI. Finally, we describe the thesis

organization in section 1.4 to give an overview of the proposed framework for brain microstructure imaging.

1.2 Introduction to diffusion MRI

Diffusion MRI allows us to encode the diffusive motion of water protons into the dMRI signals. This technique relies on the phenomena of nuclear magnetic resonance (NMR) and diffusion. We will describe the two phenomena to obtain the Bloch-Torrey equation (BT equation) that governs the formation of dMRI signals. Conventional methods for solving the BT equation are listed in section 1.2.2.

1.2.1 Nuclear magnetic resonance

Matter is made of atoms whose nuclei have magnetism and spin. Spin (denoted by a quantum operator $\boldsymbol{S}$) is the intrinsic angular momentum of atomic nuclei. For a nucleus, the nuclear magnetic moment $\boldsymbol{\mu}$ is related to its spin by [23, p. 25]

$$\boldsymbol{\mu} = \gamma \boldsymbol{S}, \tag{1.1}$$

where γ is the *gyromagnetic ratio* of the nucleus. For example, the gyromagnetic ratio of hydrogen nucleus, proton (ignore the rare isotopes), is $\gamma = 0.26752\ rad/(\mu s \cdot mT)$.

The nuclear magnetic moment interacts with magnetic field. NMR allows us to manipulate the nuclear spins and magnetic moment by externally applied magnetic fields. Generating magnetic fields is one of the main functions of MRI scanners. NMR involves three magnetic fields: a static field denoted by $\boldsymbol{B}_0$, a time-varying field called radiofrequency pulse (RF pulse), and an inhomogeneous magnetic field generating a magnetic field gradient denoted by $\boldsymbol{G}$. Next, we describe the role of the three magnetic fields in NMR. Throughout this thesis, we only study the water proton (a spin-1/2 nucleus) due to its abundance in the human body.

Static magnetic field $\boldsymbol{B}_0$

The field $\boldsymbol{B}_0$ is homogeneous and static, applied by an MRI scanner to a sample during the entire experiment. Powerful scanners for human imaging can generate a field of about 10 Tesla. We set a coordinate system by aligning the z-direction (longitudinal direction) with the direction of $\boldsymbol{B}_0$. In the transverse plane, the two axes (x- and y-axis) are fixed in the plane. We call the fixed coordinate system the laboratory frame (lab frame), as shown in fig. 1.6(a). In the lab

frame, we have $\boldsymbol{B}_0 = B_0 \boldsymbol{e}_z$ with B_0 the strength of the static magnetic field and $\boldsymbol{e}_z$ the unit vector in the z-direction.

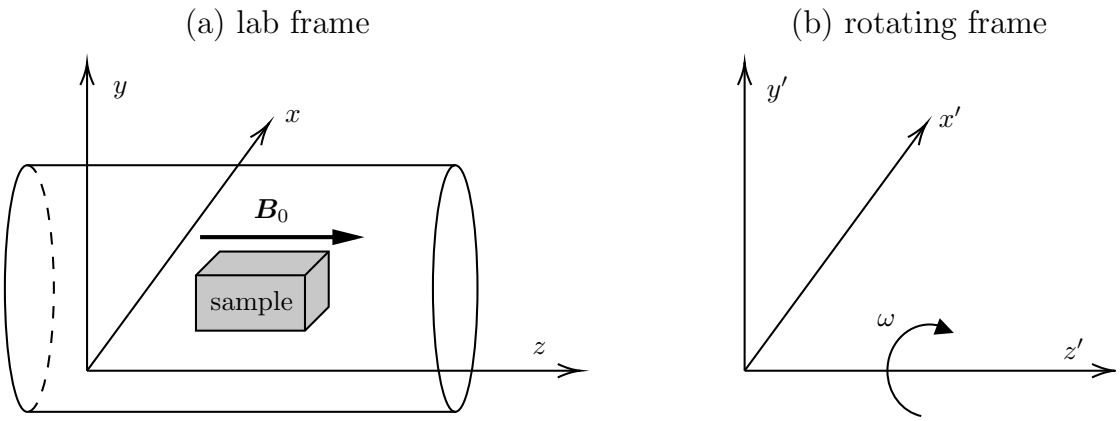


Figure 1.6: Two types of coordinate systems. **(a)** a laboratory frame. The scanner represented by the cylinder applies a static magnetic field $\boldsymbol{B}_0$ to the sample (cuboid). The laboratory frame is fixed based on the magnetic field. **(b)** a rotating frame. The coordinate system rotates around its z-axis with an angular frequency of ω.

The static field plays two roles:

1. it disturbs the orientation distribution of proton spins to make them have a statistical preference to point along the direction of $\boldsymbol{B}_0$ [24];

2. it makes the proton spins precess around $\boldsymbol{B}_0$ [23].

The precession of the spins around $\boldsymbol{B}_0$ is called Larmor precession. The angular frequency of the precession is

$$\omega_0 = \gamma B_0. \tag{1.2}$$

For example, the precession frequency of a proton in a 10 Tesla magnetic field is 425 MHz.

We do not aim to provide a quantum explanation for NMR, we refer to the book of Malcolm Levitt [23] for that. The behavior of proton spins is inherently probabilistic. Thanks to the spin density operator governing the quantum state of an astronomically large number of spins, we can obtain a magnetization vector

$$\boldsymbol{m} = m_x \boldsymbol{e}_x + m_y \boldsymbol{e}_y + m_z \boldsymbol{e}_z \tag{1.3}$$

that describes the net magnetic moment (net spin polarization) of the spins [23, 25], as illustrated in fig. 1.7.

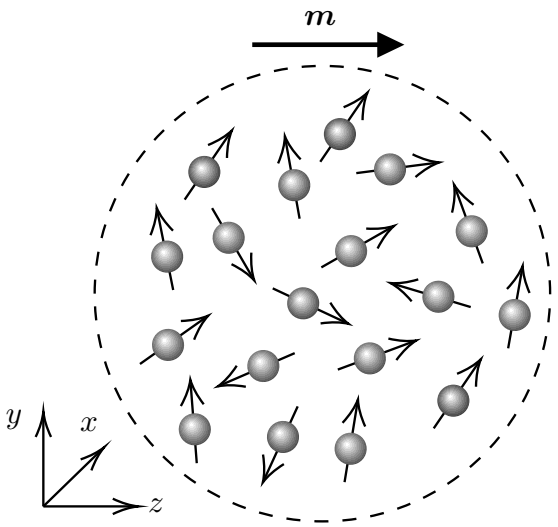


Figure 1.7: Illustration of a magnetization vector $\boldsymbol{m}$ that describes the net magnetic moment of a large number of spins. The static magnetic field $\boldsymbol{B}_0$ in the z-direction perturbs the distribution of the spin orientation, forming a net magnetic moment described by the magnetization vector $\boldsymbol{m}$ parallel to $\boldsymbol{B}_0$.

The magnetization vector $\boldsymbol{m}$ is classical. Under a magnetic field $\boldsymbol{B}$, it follows a differential equation [25, p. 35]:

$$\frac{d\boldsymbol{m}}{dt} = \gamma \boldsymbol{m} \times \boldsymbol{B}, \tag{1.4}$$

where $\times$ is the cross product of two vectors. When $\boldsymbol{B} = \boldsymbol{B}_0$, the solution of eq. (1.4) describes the Larmor precession of the magnetization vector $\boldsymbol{m}$ [25]. From now on, we employ the magnetization vector to describe NMR.

RF pulse

The RF pulse is a short-duration magnetic field $\boldsymbol{B}_1$ rotating around the z-axis in the x-y plane at the Larmor frequency ω_0. For protons whose gyromagnetic ratio is positive, the RF pulse rotates clockwise, namely,

$$\boldsymbol{B}_1(t) = B_1 \cos(\omega_0 t)\boldsymbol{e}_x - B_1 \sin(\omega_0 t)\boldsymbol{e}_y \tag{1.5}$$

where B_1 is a constant representing the strength of the RF pulse.

The evolution of $\boldsymbol{m}$ is described by eq. (1.4) with $\boldsymbol{B} = \boldsymbol{B}_0 + \boldsymbol{B}_1$. In addition, the initial condition is $\boldsymbol{m}(0) = m_0 \boldsymbol{e}_z$ with m_0 the initial magnetization. In this case, the solution of eq. (1.4) is [25, p. 36]

$$\boldsymbol{m}(t) = m_0 \left[\sin(\omega_1 t)\sin(\omega_0 t)\boldsymbol{e}_x + \sin(\omega_1 t)\cos(\omega_0 t)\boldsymbol{e}_y + \cos(\omega_1 t)\boldsymbol{e}_z\right], \tag{1.6}$$

with $\omega_1 = \gamma B_1$.

Let us introduce a rotating coordinate system (rotating frame, see fig. 1.6(b)) spinning around the z-axis with an angular frequency of ω_0. The three unit vectors in the axes of the rotating frame are

$$\begin{aligned} \boldsymbol{e}'_x &= \cos(\omega_0 t)\boldsymbol{e}_x - \sin(\omega_0 t)\boldsymbol{e}_y, \\ \boldsymbol{e}'_y &= \sin(\omega_0 t)\boldsymbol{e}_x + \cos(\omega_0 t)\boldsymbol{e}_y, \\ \boldsymbol{e}'_z &= \boldsymbol{e}_z. \end{aligned} \tag{1.7}$$

In the rotating frame, the field $\boldsymbol{B}_1$ is always aligned with $\boldsymbol{e}'_x$ and the magnetization vector eq. (1.6) is

$$\boldsymbol{m}(t) = m_0 \left[\sin(\omega_1 t)\boldsymbol{e}'_y + \cos(\omega_1 t)\boldsymbol{e}'_z\right], \tag{1.8}$$

which is a clockwise precession around the x-axis of the rotating frame ($\boldsymbol{e}'_x$) with an angular frequency of ω_1, as shown in fig. 1.8. The rotating frame makes the interpretation easier. We will use it in subsequent chapters.

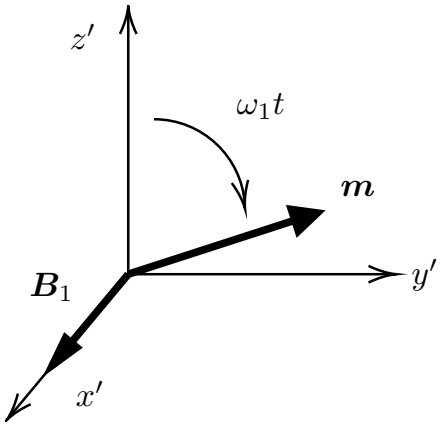


Figure 1.8: The precession of the magnetization vector around the RF pulse $\boldsymbol{B}_1$ in the rotating frame. The angular frequency of the precession is $\omega_1 = \gamma B_1$.

The primary role of the RF pulse is to rotate the magnetization vector by a certain angle around $\boldsymbol{B}_1$. The angle, $\omega_1 t = \gamma B_1 t$, can be controlled by the strength and duration of the RF pulse. Common angles are $90°$ and $180°$. A $90°$ RF pulse flips a magnetization vector aligned with $\boldsymbol{B}_0$ to the transverse plane. It is the coherent precession of a group of magnetization vectors that induces a current above the noise level in the receiving antenna and is solely responsible for generating the MRI signal. It is worth noting that the RF pulse delivers energy to the sample, thus having a heating effect [11, p. 16].

RF pulses usually have a brief duration. Without the energy supply, the magnetization vector will fall back to its thermal equilibrium state, which means it

will realign with $\boldsymbol{B}_0$. The restoration of the thermal equilibrium state involves two relaxation mechanisms: longitudinal and transverse relaxation. The longitudinal relaxation means that the component m_z of the magnetization vector returns exponentially to its initial value (m_0). This relaxation is caused by the energy loss from excited spins to their external environment [26]. We also refer to the longitudinal relaxation as T1-relaxation because it is characterized by a relaxation time denoted by T1.

Transverse relaxation concerns the exponential decay of transverse magnetization. The spin-spin interaction that destroys spin phase coherence is a source of transverse relaxation [27]. Transverse relaxation, also known as T2-relaxation, is characterized by T2 relaxation time. For brain tissue, T1 is on the order of 1 second, and T2 is about 100 milliseconds [11, p. 12]. The T1 and T2 relaxation times of human tissue can be obtained using NMR. They are valuable biomarkers for structural MRI [28].

One important application of RF pulses is the spin echo proposed by Erwin Hahn [17]. In the ideal case, all proton spins in the static magnetic field process with the same Larmor frequency. However, in actual experiments, the magnetic field is inhomogeneous, i.e., the magnetic field strength varies from place to place. Therefore, the position of a spin determines its precession frequency, which broadens the spectrum of Larmor frequencies. The inhomogeneity originates from instrumental imperfections, susceptibility effects, chemical shifts, etc. [23, 25]. The broadening of spectrum destroys the coherent precession of $\boldsymbol{m}$ at different spatial regions, resulting in a fast decay of the signals. The spin echo method can recover a strong echo signal despite magnetic field inhomogeneity.

The spin echo relies on a 180° RF pulse following the 90° RF pulse that flips the longitudinal magnetization vector to the transverse plane. Figure 1.9 illustrates the principle of spin echo. After the 90° RF pulse, magnetization vectors from different points in space precess at different frequencies due to the field inhomogeneity. The spreading of the magnetization vectors is called dephasing. The 180° RF pulse reverses the orientation of magnetization vectors, forming a spin echo at the echo time TE.

Figure 1.10 shows an experimental spin echo signal [29, p. 91]. The received signal can be treated as an amplitude-modulated signal whose carrier wave oscillates at Larmor frequency.

The refocusing of the magnetization vectors is not perfect if the spins move in an inhomogeneous magnetic field. The motivation of diffusion MRI is to leverage the dephasing due to the spin movements in a controlled inhomogeneous magnetic field.

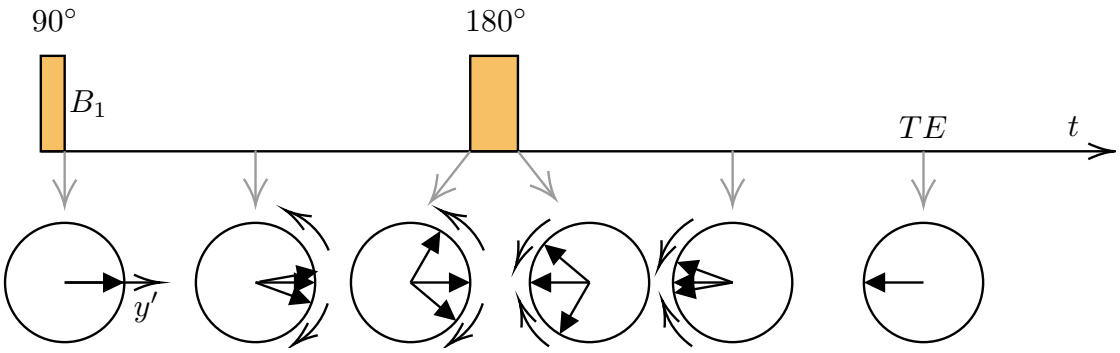


Figure 1.9: Illustration of the principle of spin echo. The magnetization vectors from different spatial regions are plotted in the transverse plane of the rotating frame. The angles between vectors demonstrate the phase differences between them. For clarity, we ignore the relaxation effects. The refocusing of magnetization vectors at time TE is called spin echo.

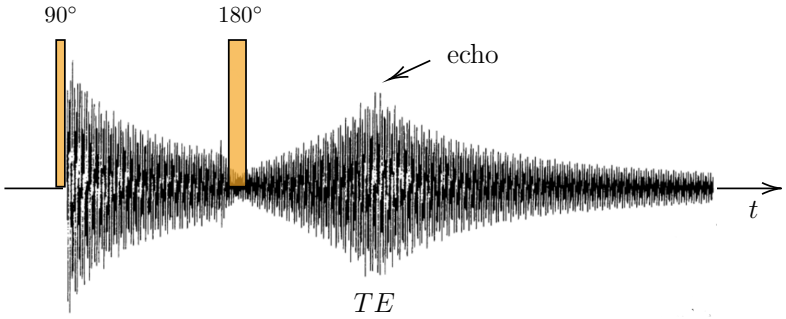


Figure 1.10: Experimental echo signal. The figure is adapted based on the image of the book of A.E. Derome [29, p. 91].

Magnetic field gradient

By adding a magnetic field gradient $\boldsymbol{G}$ to the static magnetic field, we purposely create an inhomogeneous magnetic field

$$\boldsymbol{B}(\boldsymbol{x}) = (B_0 + \boldsymbol{G} \cdot \boldsymbol{x})\, \boldsymbol{e}_z, \tag{1.9}$$

where $\boldsymbol{x}$ represents a point in space and $\boldsymbol{G} = G_x\boldsymbol{e}_x + G_y\boldsymbol{e}_y + G_z\boldsymbol{e}_z$. Due to Gauss's law for magnetism ($\nabla \cdot \boldsymbol{B} = \boldsymbol{0}$), we have $G_z = 0$, which means the gradient must lie in the transverse plane. We can tilt the magnetic field slightly at a certain angle to obtain a gradient pointing in the z-direction. The tilt angle is small because $B_0/\|\boldsymbol{G}\|$ is larger than the voxel size [30]. For human imaging, a modern scanner can generate a magnetic field gradient up to $300\ mT/m$. Figure 1.11 shows the gradients parallel and perpendicular to the static magnetic field.

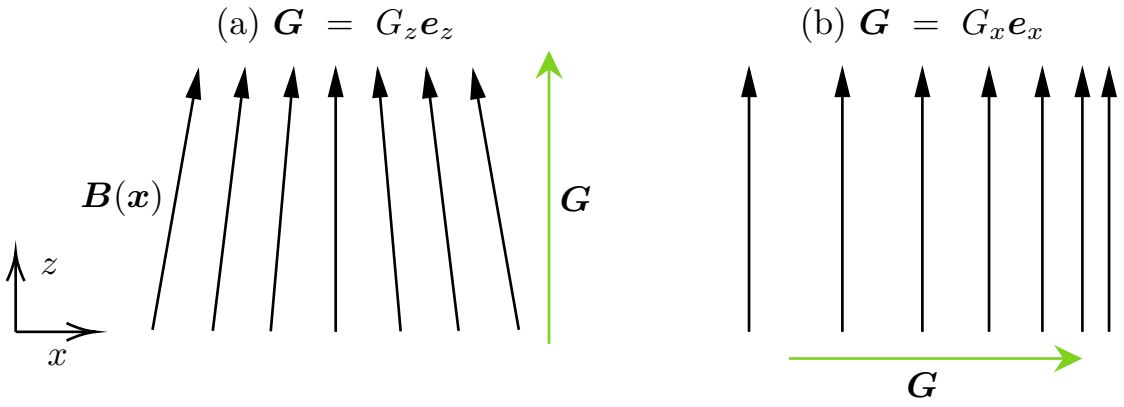


Figure 1.11: Magnetic field gradients parallel and perpendicular to the z-direction. **(a)** a gradient in the z-direction. The density of the magnetic field lines indicates the field strength. The tilt angles are exaggerated. **(b)** a gradient in the x-direction.

Magnetic field gradient has three main functions: (1) slice selection, (2) image encoding, (3) diffusion encoding [11, p. 13]. We describe the first two functions in this section. The last function will be presented in section 1.2.2.

Slice selection occurs by simultaneously applying a magnetic field gradient with a $90°$ RF pulse for a short period. The magnetic field is inhomogeneous due to the additional magnetic field gradient. Therefore, the Larmor frequency changes along the direction of the gradient. When a $90°$ RF pulse is applied, only the magnetization vectors in a certain perpendicular slice are flipped ninety degrees to generate MRI signals. Figure 1.12 demonstrates the slice selection. We notice that the magnetization vectors near the selected slice are also rotated by a certain angle, giving a certain thickness to the selected slice. In practice, the RF pulses and the gradients have more complex time profiles than the rectangles shown in fig. 1.12 to improve the spatial selectivity. The technical details are out of the scope of this thesis.

After the slice selection, magnetization vectors at different regions evolve differently due to the heterogeneous micro-environment experienced by spins. For example, the T2 relaxation time varies across brain voxels, resulting in a non-uniform distribution of transverse magnetization. The spatial distribution of magnetization can produce an image of the selected slice. Applying magnetic field gradient pulses in the transverse plane can help obtain this image.

Let us focus on the selected slice and study the distribution of transverse magnetization *density* in the rotating frame. The magnetic gradient is $\boldsymbol{G}(t) = G_x(t)\boldsymbol{e}_x + G_y(t)\boldsymbol{e}_y$. We denote $M_x(\boldsymbol{x}, t)$ ($M_y(\boldsymbol{x}, t)$) the x-component (y-component) of the magnetization density at point $\boldsymbol{x}$ in the slice. Measuring MRI signals using different magnetic field gradients allows us to obtain the initial dis-

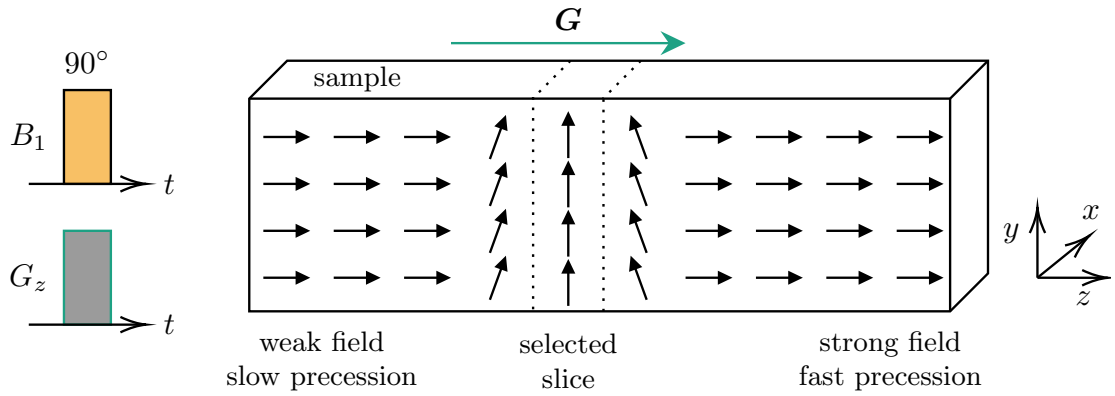


Figure 1.12: Illustration of slice selection by a magnetic field gradient and a $90°$ RF pulse. The cuboid represents a sample. The gradient produces a linear distribution of Larmor frequencies in the z-direction. The $90°$ RF pulse with a frequency ω selects a slice in which magnetization vectors precess with the same frequency ω.

tribution $M_x(\boldsymbol{x}, 0)$ and $M_y(\boldsymbol{x}, 0)$.

Substituting eq. (1.9) into eq. (1.4) and including the T2 relaxation, we obtain the Bloch equation [27] governing $M_x(\boldsymbol{x}, t)$ and $M_y(\boldsymbol{x}, t)$:

$$\frac{\partial M_x}{\partial t} = (\omega_0 + \gamma \boldsymbol{G} \cdot \boldsymbol{x})\, M_y - \frac{M_x}{T2}, \tag{1.10}$$

$$\frac{\partial M_y}{\partial t} = -(\omega_0 + \gamma \boldsymbol{G} \cdot \boldsymbol{x})\, M_x - \frac{M_y}{T2}. \tag{1.11}$$

Let $M'_x(\boldsymbol{x}, t)$ and $M'_y(\boldsymbol{x}, t)$ denote the x- and y-component of the transverse magnetization in the rotating frame. In addition, we define a complex-valued function M' to represent the transverse magnetization

$$M' \equiv M'_x - \imath M'_y, \tag{1.12}$$

with $\imath$ the imaginary unit.

By reformulating eqs. (1.10) and (1.11) in the rotating frame, we obtain the following equation

$$\frac{\partial M'(\boldsymbol{x}, t)}{\partial t} = -\imath\gamma \boldsymbol{G}(t) \cdot \boldsymbol{x} M'(\boldsymbol{x}, t) - \frac{M'(\boldsymbol{x}, t)}{T2(\boldsymbol{x})}, \tag{1.13}$$

with an initial magnetization density distribution $f(\boldsymbol{x})$ being

$$f(\boldsymbol{x}) \equiv M'(\boldsymbol{x}, 0) = M_x(\boldsymbol{x}, 0) - \imath M_y(\boldsymbol{x}, 0). \tag{1.14}$$

The target is to find the unknown initial density distribution $f(\boldsymbol{x})$ using MRI signals.

We turn on the magnetic field gradient in a certain direction for a short duration. Suppose the gradient direction is $\boldsymbol{u}_g$, the gradient intensity is a constant g, and the duration is η. The MRI signal at time η is

$$s = \int M'(\boldsymbol{x}, \eta) d\boldsymbol{x}, \tag{1.15}$$

which represents the total transverse magnetization in the selected slice.

The solution of eq. (1.13) at the time η is [31]

$$\begin{aligned} M'(\boldsymbol{x}, \eta) &= f(\boldsymbol{x}) e^{-\eta/T2(\boldsymbol{x})} e^{-\imath\gamma \boldsymbol{x} \cdot \int_0^\eta \boldsymbol{G}(t) dt} \\ &= f(\boldsymbol{x}) e^{-\eta/T2(\boldsymbol{x})} e^{-\imath g\gamma\eta \boldsymbol{u}_g \cdot \boldsymbol{x}}. \end{aligned} \tag{1.16}$$

Let us introduce a new variable $\boldsymbol{k} = g\gamma\eta\boldsymbol{u}_g/2\pi$. The MRI signal is a function of the new variable

$$s(\boldsymbol{k}) = \int f(\boldsymbol{x}) e^{-\eta/T2(\boldsymbol{x})} e^{-2\pi\imath \boldsymbol{k}\cdot\boldsymbol{x}} d\boldsymbol{x}. \tag{1.17}$$

If the gradient duration is short, the T2 relaxation term is negligible. In this case, an MRI signal is a point in the Fourier spectrum of the unknown magnetization density distribution $f(\boldsymbol{x})$. One can sample the spectrum by taking the gradient in various directions in the transverse plane. An inverse Fourier transform of the sampled spectrum gives the magnetization density distribution $f(\boldsymbol{x})$. If T2 relaxation is not negligible, the inverse Fourier transform yields the magnetization density distribution attenuated by T2 relaxation.

In practice, gradient sequences, such as echo planar imaging (EPI) pulse sequences and spiral pulse sequences, are proposed to improve the imaging quantity and speed up the acquisition [11, p. 17].

1.2.2 Diffusion and Bloch-Torrey equation

In the description presented in the above section, the effect of spin diffusion is largely ignored. This is acceptable if the magnetic field gradient is weak, so the dephasing caused by spin diffusion is negligible. Diffusion MRI purposely employs a time-varying magnetic field gradient to make spins in motion out of phase. The dephasing reflects the spins' displacement in a medium. This section introduces the Bloch-Torrey equation, a formal description of diffusion MRI.

Diffusion

When two solutions of different concentrations are mixed, the solution concentration will gradually converge even without stirring. This transport phenomenon without bulk motion is called diffusion. In fluids, diffusion is caused by disordered collisions between an enormous number of particles. Due to the

randomness of collisions, the movement of a particle is a stochastic process described by Brownian motion [32].

From a continuum point of view, the randomness is averaged out, leaving a deterministic description of a continuous function. Let us take solution mixing as an example. Suppose the solution concentration is a function c, which is inhomogeneous in a domain Ω. The concentration difference generates a diffusive flux $\boldsymbol{J}$ described by Fick's first law

$$\boldsymbol{J} = -D_0 \nabla c(\boldsymbol{x}, t), \quad \boldsymbol{x} \in \Omega, \tag{1.18}$$

where D_0 is the diffusion coefficient of the solvent molecule. In addition, we have the conservation law

$$\frac{\partial c(\boldsymbol{x}, t)}{\partial t} + \nabla \cdot \boldsymbol{J} = 0, \quad \boldsymbol{x} \in \Omega. \tag{1.19}$$

Combining the two equations, we get the diffusion equation that describes the evolution of concentration

$$\frac{\partial c(\boldsymbol{x}, t)}{\partial t} = D_0 \nabla^2 c, \quad \boldsymbol{x} \in \Omega. \tag{1.20}$$

Diffusion MRI detects the random motion of spins in an inhomogeneous magnetic field. The magnetization precession, the T2 relaxation, and the diffusion give rise to the Bloch-Torrey equation governing diffusion MRI.

Bloch-Torrey equation

The introduction of a diffusion term to the Bloch equation eq. (1.13) is proposed by H.C. Torrey [33]. For clarity, we drop the superscript of the complex-valued transverse magnetization eq. (1.12) and refer to it as magnetization. We concentrate on the water proton diffusion inside a domain Ω, e.g., a neuron.

By taking diffusion into account, the complex-valued magnetization M in the rotating frame is governed by the Bloch-Torrey partial differential equation (Bloch-Torrey equation, BT equation) [33]:

$$\frac{\partial}{\partial t} M(\boldsymbol{x}, t) = \left(D_0 \nabla^2 - \imath \gamma \boldsymbol{x} \cdot \boldsymbol{G}(t) - \frac{1}{T2} \right) M(\boldsymbol{x}, t), \ \boldsymbol{x} \in \Omega, \tag{1.21}$$

with D_0 the diffusion coefficient. The self-diffusion coefficient of water molecules at $37\ ^\circ C$ is $3 \times 10^{-3}\ \mu m^2/\mu s$ [34].

Inside the domain, the diffusion of water protons is restricted by, for instance, cellular membranes. The motion restriction is reflected by a boundary condition

$$D_0 \nabla M(\boldsymbol{x}, t) \cdot \boldsymbol{n}(\boldsymbol{x}) = \kappa M(\boldsymbol{x}, t), \ \boldsymbol{x} \in \partial\Omega, \tag{1.22}$$

where $\partial\Omega$ denotes the boundary of Ω, $\boldsymbol{n}(\boldsymbol{x})$ is the unit outward pointing normal vector of the point $\boldsymbol{x}$ in the boundary, and κ is the permeability or surface relaxivity of the boundary.

The magnetic field gradient is typically turned on after the 90° RF pulse. We set the time origin at the time when the magnetic field gradient starts. The initial condition is

$$M(\boldsymbol{x}, 0) = \rho, \ \boldsymbol{x} \in \Omega, \tag{1.23}$$

where ρ is the initial magnetization density, which is assumed to be homogeneous.

Diffusion is an attenuation mechanism that we aim to leverage to reveal microstructure properties below the scanner resolution. We achieve this by applying a time-varying magnetic field gradient between the slice selection and image encoding. Figure 1.13 illustrates a simplified version of a dMRI experiment setting. First, we perform the slice selection using a 90° RF pulse and a magnetic field gradient in the z-direction. Second, spin diffusion is encoded using magnetic field gradients. The gradient time profile (between 0 and $\delta + \Delta$) presented in fig. 1.13 is called the pulsed-gradient spin echo (PGSE) sequence. A constant magnetic field gradient is turned on for a duration of δ. Then at time Δ, the gradient is turned back on for the same duration. A 180° RF pulse is applied between them to generate a spin echo. Finally, we perform image encoding using magnetic field gradients in the transverse plane. The dMRI signal is measured at echo time TE. Similar to eq. (1.15), the dMRI signal is

$$s = \int_\Omega M(\boldsymbol{x}, TE) d\boldsymbol{x}. \tag{1.24}$$

The BT equation does not have a simple solution, especially when the shape of the domain Ω is irregular. To simulate the dMRI signals, we need numerical methods. The following section presents some conventional simulation methods. It is worth noting that the structural properties we wish to probe contribute to the dMRI signal through boundary conditions. Consequently, the geometrical modeling of the domain Ω is essential for dMRI simulation.

The diffusion MRI signal generally depends on the gradient $\boldsymbol{G}$ and the echo time TE. If the domain is an open space where water protons can diffuse freely in all directions, the dMRI signal has a simple form [35]

$$s = s_0 \, e^{-D_0 b}, \tag{1.25}$$

where s_0 is the signal without the diffusion-encoding gradient and b called b-value is a diffusion-weighting factor. The signal s_0 mainly reflects the effect of T2 relaxation. Using the PGSE sequence presented in fig. 1.13, the b-value is

$$b = \gamma^2 g^2 \delta^2 \left(\Delta - \delta/3\right), \tag{1.26}$$

where $g = \|\boldsymbol{G}(0)\|$ is the intensity of the magnetic field gradient.

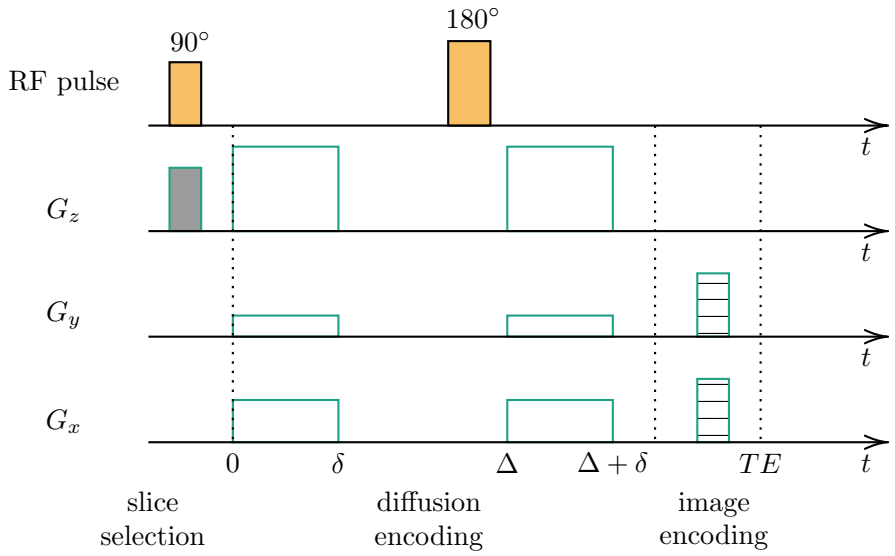


Figure 1.13: Illustration of the RF pulses and magnetic field gradients for a dMRI experiment. The slice selection is performed with a $90°$ RF pulse and a magnetic field gradient in the z-direction. Then, we turn on the PGSE sequence to encode spin diffusion. A $180°$ RF pulse is turned on between the diffusion-encoding pulses to generate a spin echo. Finally, we perform image encoding using magnetic field gradients in the transverse plane.

Numerical methods

The predominant numerical methods to solve the BT equation include the Monte-Carlo method [36–42], the finite difference method [43], the finite element method (FEM) [44–46], and the matrix formalism (MF) method [47–50].

Numerical methods to solve the Bloch-Torrey equation with arbitrary temporal profiles have been proposed in [43, 45, 46, 51]. The computational domain is discretized either by a Cartesian grid [43, 51, 52] or by finite elements [45, 46, 53–55]. The unstructured mesh of a finite element discretization appeared to be better than a Cartesian grid in both geometry description and signal approximation [45]. For time discretization, both explicit and implicit ODE solvers have been used. The efficiency of diffusion MRI simulations is also improved by either a high-performance FEM computing framework [56, 57] for large-scale simulations on supercomputers or a discretization on manifolds for thin-layer and thin-tube media [58]. Finite elements diffusion MRI simulations can be seamlessly integrated with cloud computing resources [59]. A MATLAB Toolbox called

SpinDoctor [44] is a diffusion MRI simulation framework based on solving the BT equation using finite elements and an adaptive time-stepping method.

The Matrix formalism methods [47, 48], which decompose the solution of the BT equation onto Laplacian eigenbases, provide an exciting perspective to the diffusion MRI signal. A numerical matrix formalism method that is adapted to irregular geometries is provided by the SpinDoctor toolbox [49, 50]. In this thesis, all FEM and MF simulations are performed with the SpinDoctor toolbox.

Monte-Carlo methods use random walkers to mimic the diffusion process in a geometrical configuration. The implementations of Monte-Carlo method include [36–42, 60, 61]. Some GPU-based Monte-Carlo simulators are also available [62, 63]. Software packages using this approach include

1. Camino Diffusion MRI Toolkit [64], developed at UCL;
2. disimpy [65], a GPU-based Monte-Carlo simulator, developed at UCL;
3. Diffusion Microscopist Simulator, [37] developed at Neurospin, CEA;
4. A CUDA-based Monte-Carlo simulator [62].

1.3 Diffusion MRI for brain imaging

In this section, we introduce several brain imaging methods using dMRI signals. The simplest method is to present the raw image whose contrast is given by signal intensity, as shown in fig. 1.5. However, the signal intensity results from a combination of diffusion and other effects like relaxation. Therefore, interpreting the raw image could be difficult [66].

Numerous dMRI methods are designed to obtain more specific information from voxelwise signals. Most rely on modeling the dMRI signals from a brain voxel to gain sensitivity to the underlying brain microstructure. This section describes four common methods for dMRI brain imaging.

Apparent diffusion coefficient

We can assume a simple expression for the dMRI signal

$$s(b)/s_0 = e^{-D_e b}. \tag{1.27}$$

The effective diffusion coefficient D_e, also known as the apparent diffusion coefficient (ADC), can be computed using two signals from a brain voxel. The quantity $s(b)$ denotes a signal obtained with a magnetic field gradient whose diffusion-weighting factor is b. The second quantity s_0 is the signal measured without the

diffusion-encoding gradient. We refer to the fraction $s(b)/s_0$ as the normalized signal or signal attenuation. The ADC can be empirically estimated by

$$D_e = \frac{-\ln\left(s(b_2)/s(b_1)\right)}{b_2 - b_1}, \tag{1.28}$$

where $s(b_2)$ and $s(b_1)$ are the dMRI signals obtained with the b_2 and b_1 b-values, respectively [67].

We can perform the above computation voxel by voxel to obtain a brain map whose contrast is given by voxelwise ADCs. However, the simple signal formula eq. (1.27) is accurate only if water protons can diffuse freely [67]. The condition is generally not true due to the structural complexity of brain tissue. Nonetheless, eq. (1.27) is a satisfactory approximation to the dMRI signals when the Gaussian phase approximation (GPA) is applicable [68, 69]. The GPA depends on a perturbative method to expand the logarithm of the normalized signal at low b-values (cumulant expansion) [70]. One can obtain eq. (1.27) by keeping the first term of the cumulant expansion (when the higher-order terms are negligible).

ADC maps have been commonly used in clinical practice, such as the detection of cerebral ischemia [71] and monitoring tumor progress [72, 73].

Diffusion tensor imaging

A major application of diffusion MRI is tractography. For brain tissue with organized fiber structure, such as white matter, the dMRI signal is anisotropic [20], which means the signal depends on the direction of the magnetic field gradient. The diffusion anisotropy is better handled using diffusion tensor imaging (DTI) [74]. The signal expression used in DTI is

$$s(b, \boldsymbol{u}_g)/s_0 = e^{-\boldsymbol{u}_g^T \boldsymbol{D} \boldsymbol{u}_g b}. \tag{1.29}$$

where $\boldsymbol{D}$ is the diffusion tensor represented by a 3×3 symmetric matrix, $\boldsymbol{u}_g$ is the magnetic field gradient direction (a unit vector), and the superscript ^{T} denotes transpose. The product $\boldsymbol{u}_g^T \boldsymbol{D} \boldsymbol{u}_g$ represents the apparent diffusion coefficient in direction $\boldsymbol{u}_g$. Because $\boldsymbol{D}$ is symmetric, it has six independent numbers. Estimating a diffusion tensor requires ADCs measured in at least six non-collinear directions.

Diffusion tensors can reveal the orientation of the underlying fiber structure. We achieve this by performing an eigendecomposition of the diffusion tensor. Because $\boldsymbol{D}$ is a real symmetric matrix, eigendecomposition is always feasible. We denote the three eigenvalues by $\lambda_1,\ \lambda_2$ and λ_3. The corresponding eigenvectors are $\boldsymbol{v}_1,\ \boldsymbol{v}_2$ and $\boldsymbol{v}_3$. The direction of the eigenvector with the largest eigenvalues indicates the major fiber direction in a voxel [75].

The eigenvalues allow for the definition of useful biomarkers. The trace of the diffusion tensor is related to the mean diffusivity ($\overline{\lambda}$)

$$\overline{\lambda} = \frac{1}{3} tr(\boldsymbol{D}) = \frac{\lambda_1 + \lambda_2 + \lambda_3}{3}. \tag{1.30}$$

Fractional anisotropy (FA) and relative anisotropy (RA) are two biomarkers to quantify diffusion anisotropy. They are defined as [11, p. 95]

$$FA = \sqrt{\frac{3}{2}} \frac{\sqrt{(\lambda_1 - \overline{\lambda})^2 + (\lambda_2 - \overline{\lambda})^2 + (\lambda_3 - \overline{\lambda})^2}}{\sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}}, \tag{1.31}$$

$$RA = \sqrt{\frac{1}{3}} \frac{\sqrt{(\lambda_1 - \overline{\lambda})^2 + (\lambda_2 - \overline{\lambda})^2 + (\lambda_3 - \overline{\lambda})^2}}{\overline{\lambda}}. \tag{1.32}$$

For clinical practice, an increase in mean diffusivity indicates diseases such as necrosis [76]. Fractional anisotropy is useful for assessing the maturation of infants' cerebral white matter fiber [77] and reading ability [78]. Besides, DTI is suitable for detecting diseases due to white matter abnormalities, such as multiple sclerosis [79] and Alzheimer's disease [80].

Biophysical modeling - Standard Model

The two methods mentioned above do not make assumptions about the underlying tissue microstructure. To improve sensitivity to brain microstructure properties, biophysical models of diffusion MRI have become more and more popular. The basic idea of biophysical modeling is "compartmentalization", which means the essential components (compartment) of a brain voxel are studied separately to get a signal expression for each compartment. The dMRI signal from a voxel is a weighted sum of the compartmental signals.

We first describe a class of methods referred to as the "Standard Model" of diffusion in neuronal tissue [69]. This model assumes that a brain voxel has three compartments: intra-neurite, extra-neurite, and CSF.

A collection of long cylinders models the intra-neurite compartment. The orientation of a cylinder is represented by a unit vector $\boldsymbol{n}$, and a fiber *orientation distribution function* (ODF) characterizes the orientation distribution of the cylinders. The ODF is denoted by $P(\boldsymbol{n})$, which is normalized to one ($\int P(\boldsymbol{n}) d\boldsymbol{n} = 1$). For diffusion properties, a cylinder is characterized by a longitudinal diffusion coefficient denoted by $D^{\parallel}_{\text{cyl}}$. The transverse diffusion inside a cylinder is assumed to be negligible. The signal from a cylinder is [69]

$$s_{\text{cyl}} = s_0 \, e^{-bD^{\parallel}_{\text{cyl}}(\boldsymbol{g}\cdot\boldsymbol{n})^2}, \tag{1.33}$$

with $\boldsymbol{g}$ the direction of magnetic field gradients.

The diffusion in an extra-neurite space around a cylinder is characterized by longitudinal and transverse diffusion coefficients denoted by $D^{\parallel}_{\text{ext}}$ and $D^{\perp}_{\text{ext}}$, respectively. The signal from the extra-neurite space is [69]

$$s_{\text{ext}} = s_0 \, e^{-bD^{\perp}_{\text{ext}} - b(D^{\parallel}_{\text{ext}} - D^{\perp}_{\text{ext}})(\boldsymbol{g}\cdot\boldsymbol{n})^2}, \tag{1.34}$$

Water molecules are assumed to diffuse freely inside the last compartment, CSF, which gives rise to the CSF signal

$$s_{\text{csf}} = s_0 \, e^{-bD_{\text{csf}}}, \tag{1.35}$$

with D_{csf} the diffusion coefficient in CSF.

Equations (1.33) to (1.35) are the signal expressions describing the diffusion inside and around a cylinder. They give a kernel function [69]

$$\begin{aligned} \mathcal{K}(b, \boldsymbol{g}\cdot\boldsymbol{n}) &= f_{\text{cyl}} s_{\text{cyl}} + (1 - f_{\text{cyl}} - f_{\text{csf}}) s_{\text{ext}} + f_{\text{csf}} s_{\text{csf}} \\ &= s_0 \left[f_{\text{cyl}} e^{-bD^{\parallel}_{\text{cyl}}(\boldsymbol{g}\cdot\boldsymbol{n})^2} + (1 - f_{\text{cyl}} - f_{\text{csf}}) e^{-bD^{\perp}_{\text{ext}} - b(D^{\parallel}_{\text{ext}} - D^{\perp}_{\text{ext}})(\boldsymbol{g}\cdot\boldsymbol{n})^2} + f_{\text{csf}} e^{-bD_{\text{csf}}} \right], \end{aligned} \tag{1.36}$$

where f_{cyl} and f_{csf} are the *signal fractions* of the intra-neurite and CSF compartments. It is worth noting that all compartments are assumed to be non-exchanging with this formulation.

Finally, the signal from a brain voxel is

$$s(b, \boldsymbol{g}) = \int P(\boldsymbol{n}) \mathcal{K}(b, \boldsymbol{g}\cdot\boldsymbol{n}) d\boldsymbol{n}. \tag{1.37}$$

Equation (1.37) is a forward model with an explicit expression to describe the signal formation mechanism in a brain voxel. One can fit eq. (1.37) to measured signals to estimate model parameters, such as f_{cyl}, f_{ext}, and $D^{\parallel}_{\text{cyl}}$. However, parameter estimation is not trivial. We refer to the review paper [69] for more details about the parameter estimation. This model and its variants are used to quantify neurite density and dispersion for gray and white matter in the literature [81–84].

Biophysical modeling - SANDI

The above method assumes the contribution of cell bodies (neurons and glial cells) is integrated into the extra-neurite compartment. However, some studies suggest that this assumption is not valid in the gray matter at high b-values [85–87]. Here, we describe the *soma and neurite density imaging* (SANDI) model [88], which incorporates the soma size and density into biophysical models.

Unlike the Standard Model, SANDI focuses on the *direction-averaged signal*. We denote the direction of magnetic field gradients as $\boldsymbol{u}_g$. The direction-averaged signal is

$$\overline{s} \equiv \int_{\|\boldsymbol{u}_g\|=1} s d\boldsymbol{u}_g. \tag{1.38}$$

SANDI assumes a brain voxel has three compartments: intra-neurite, intra-soma, and extracellular space (ECS). The signal from the intra-neurite compartment follows the Standard Model (eq. (1.33)). By taking the directional average, the intra-neurite signal is [88, 89]

$$\overline{s}_{\text{in}} = s_0 \sqrt{\frac{\pi}{4bD_{\text{in}}}} erf\left(\sqrt{bD_{\text{in}}}\right). \tag{1.39}$$

SANDI represents a soma as a sphere, and a group of somas is assumed to be represented by an "average" sphere. Using PGSE sequences, the intra-soma signal is [88]

$$\overline{s}_{\text{is}} = s_0\, e^{-D'_{\text{is}} b}. \tag{1.40}$$

where D'_{is} is a function of δ, Δ, the soma radius r_{is} and the intra-soma diffusion coefficient D_{is}. The explicit form of D'_{is} is [90, 91]

$$\begin{aligned} D'_{\text{is}} =& \frac{2}{D_s \delta^2 (\Delta - \frac{\delta}{3})} \times \\ & \sum_{m=1}^{\infty} \frac{\alpha_m^{-4}}{\alpha_m^2 r_s^2 - 2} \left(2\delta - \frac{2 + e^{-\alpha_m^2 D_s(\Delta-\delta)} + e^{-\alpha_m^2 D_s(\Delta+\delta)} - e^{-\alpha_m^2 D_s \delta} - e^{-\alpha_m^2 D_s \Delta}}{\alpha_m^2 D_s} \right) \end{aligned} \tag{1.41}$$

where α_m is the m-th root of $(\alpha r_s)^{-1} J_{\frac{3}{2}}(\alpha r_s) = J_{\frac{5}{2}}(\alpha r_s)$, with J_n the Bessel functions of the first kind.

The ECS is assumed to be a free diffusion space, which gives

$$\overline{s}_{\text{ecs}} = s_0\, e^{-D_{\text{ecs}} b}, \tag{1.42}$$

with D_{ecs} the water diffusion coefficient in ECS.

Finally, a weighted sum of compartmental signals gives the direction-averaged signal from a brain voxel

$$\begin{aligned} \overline{s} =& (1 - f_{\text{in}} - f_{\text{ecs}}) \overline{s}_{\text{is}} + f_{\text{in}} \overline{s}_{\text{in}} + f_{\text{ecs}} \overline{s}_{\text{ecs}} \\ & s_0 \left[(1 - f_{\text{in}} - f_{\text{ecs}}) e^{-D'_{\text{is}} b} + f_{\text{ecs}} e^{-D_{\text{ecs}} b} + f_{\text{in}} \sqrt{\frac{\pi}{4bD_{\text{in}}}} erf\left(\sqrt{bD_{\text{in}}}\right) \right]. \end{aligned} \tag{1.43}$$

Similar to the Standard Model, one can fit the measured signals to the explicit signal expression eq. (1.43) to get the model parameters. In the original paper of

SANDI [88], the intra-soma diffusion coefficient is fixed to $3 \times 10^{-3}\ \mu m^2/\mu s$ and a random forest regression is used to estimate the remaining five parameters: f_{in}, D_{in}, f_{ecs}, D_{ecs}, and r_s. The estimation of the five SANDI parameters requires at least five independent measurements with non-zero b-values [92]. An open-source convex optimization package, AMICO [93], provides a routine for fitting the SANDI model to experimental data.

Summary

The above methods have a similar pattern. The first step involves proposing a forward model to explain the signal from a brain voxel. In the cases above, the forward models are the explicit signal expressions eqs. (1.27), (1.29), (1.37) and (1.43). The second step requires "inverting" the forward model to estimate the model parameters, which serve as the biomarkers of brain microstructure.

The explicit signal expressions require various assumptions. However, these assumptions may not be valid. For example, the cumulant expansion and the GPA can fail with a weak magnetic field gradient whose intensity is lower than $20\ mT/m$ [30, p. 246]. In addition, the validity regimes of several signal expressions depend on microstructure length scales [91]. A brain voxel may exhibit multiple length scales (e.g., various soma radii) so that different validity regimes may co-exist or emerge progressively [94], making model validation difficult. Besides, the complex brain microstructure shown in fig. 1.4 contrasts sharply with the simple biophysical models. Subtle effects, e.g., neurite undulation and soma-neurite water exchange, are not included.

To overcome some drawbacks of the existing methods, we aim to replace simple geometric models with realistic neuron models and explicit signal expressions with diffusion MRI simulations. Indeed, the Bloch-Torrey equation provides a "gold-standard" forward model allowing us to simulate trustworthy intracellular signals using realistic neuron geometrical models. Figure 1.14 compares the new model proposed in this thesis and the two biophysical models.

1.4 Thesis organization

The ultimate goal of this thesis is to facilitate brain microstructure imaging by directly using diffusion MRI simulations. There are three main challenges. First, we need precise geometrical modeling for neurons and glial cells. Existing neuron modeling tools [95–97] are mainly for visualization purposes. The quality of the model is not good enough for simulation. Besides, we need a large number of neuron models to achieve simulation-driven brain microstructure imaging. Therefore, the modeling tool needs to be automatic and robust. To our knowl-

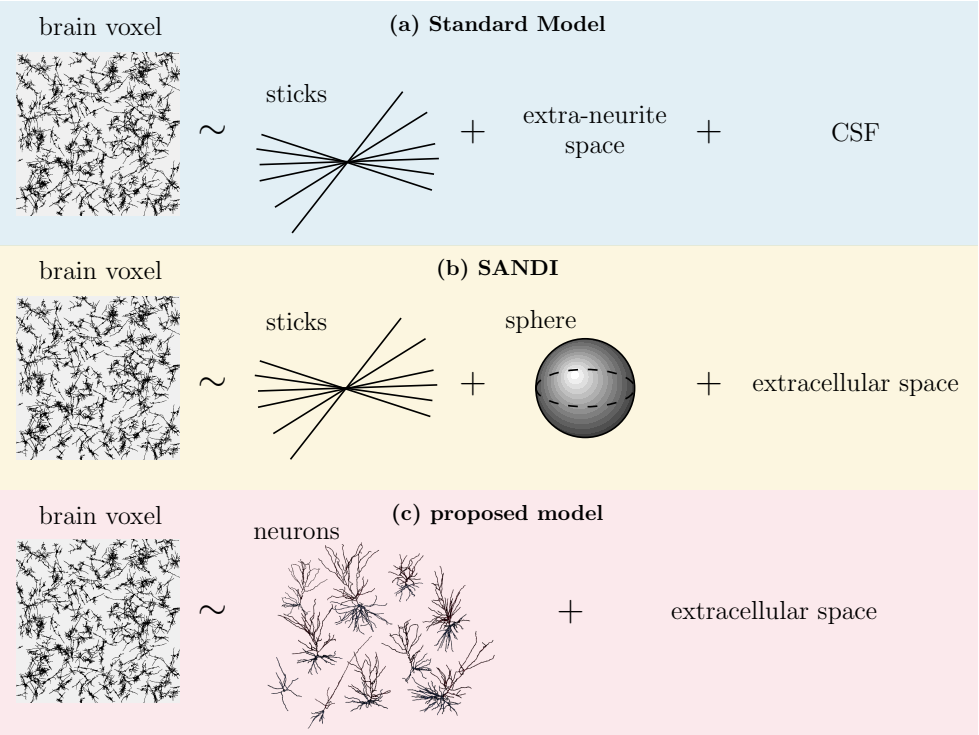


Figure 1.14: Illustration of the two biophysical models and the new model proposed in this thesis. We aim to replace simple geometries with realistic neuron models and explicit signal expressions with dMRI simulations.

edge, no such tool is available to the public. Second, the existing dMRI simulation tools are not efficient enough to perform simulations on a large number of realistic neuron models. Third, inverting the gold-standard forward model (BT equation) is non-trivial. Unlike the four methods mentioned in section 1.3, the simulated signal does not have an explicit formula with specific model parameters. The relationships between the microstructure properties and the simulated signals are implicit, non-parametric, and possibly high-dimensional. Therefore, solving the inverse problem is challenging.

This thesis will present the solution to the three challenges in the following chapters. In chapter 2, we developed a high-performance open-source neuron mesh generator and made over one thousand realistic cellular meshes publicly available. Chapter 3 describes two numerical simulation methods: the finite element method and the numerical matrix formalism method. We increased the computational efficiency of the numerical matrix formalism method by a factor of ten. In chapter 4, a new simulation method that provides a Fourier-type representation of the dMRI signals is described and implemented numerically. Chap-

ter 5 presents the proposed simulation-driven supervised learning framework for dMRI brain microstructure imaging. We conclude the thesis in chapter 6.

Chapter 2

Realistic Neuron Modeling

Diffusion MRI simulations and numerical phantoms can help deepen the understanding of the relationship between the cellular structure and the diffusion MRI signal. They play a significant role in the formulation and validation of appropriate models in order to answer relevant biological questions. Numerical phantoms are less costly and more flexible than physical phantoms [98]. Some recent works that use numerical simulations of the diffusion MRI signal as a part of model validation include [85, 87, 88, 99]. Simulations can help develop, test, and optimize MRI pulse sequences by modeling the response to novel pulse sequences with various tissue features [100–103]. In fact, given the recent availability of vastly more advanced computational resources, simulation frameworks have increasingly been used as standard computational tools for tissue parameter estimation [61, 104].

Despite these advantages, dMRI simulations of realistic brain tissues are still limited due to the lack of available and sophisticated numerical phantoms. Constructing numerical phantoms is challenging. Few open-source phantom-generating toolboxes or pre-generated numerical phantoms are available to the public. The work presented in this chapter aims to change this situation by providing the dMRI community with two sets of neuron meshes suitable for numerical simulation and an open-source neuron mesh generator. Throughout the thesis, we refer to a collection of vertices, edges, and faces that defines the shape of an object as a polygon mesh or simply a mesh. Figure 2.1 illustrates a triangulated surface mesh and a tetrahedral volume mesh. Realistic neuron modeling aims to represent the shape of neurons using polygon meshes on which dMRI simulations can be performed.

Section 2.1 gives a general introduction to neuron tracing, which is the most common way to record neuron morphology in neuroscience. In section 2.2, the difficulties of generating neuron meshes suitable for diffusion MRI simulations are explained. In section 2.3, we present Neuron Module, an open-source dMRI

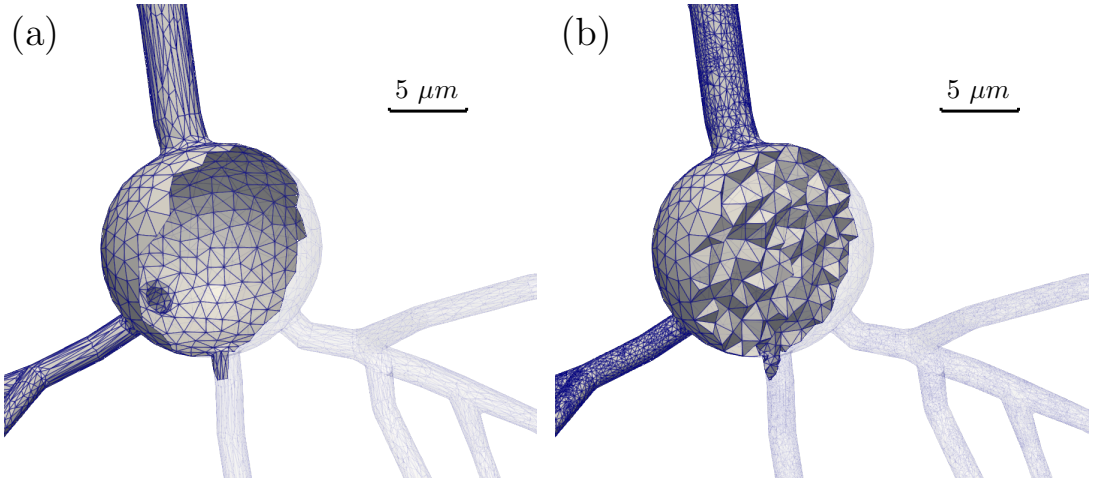


Figure 2.1: Illustration of a triangulated surface mesh **(a)** and a tetrahedral volume mesh **(b)**. Half of the meshes are made transparent to show the inside. A triangulated surface mesh is composed of a set of triangles defining the hull of an object. The space enclosed by the hull is empty. A volume mesh consists of a set of tetrahedra to model the shape of an object.

simulation package containing 65 pre-generated neuron meshes, as our first attempt to model realistic neuron morphology using triangulated surface meshes and tetrahedral volume meshes.

However, the mesh generation pipeline used in Neuron Module was limited because it requires commercial software and significant manual operations. To overcome the drawback, we developed an open-source neuron mesh generator called *swc2mesh*. We explain the implementation of *swc2mesh* and showcase four main functionalities in section 2.4. We show that *swc2mesh* is automatic, robust, versatile, and user-friendly. The neuron mesh generator allows us to build a large-scale neuron mesh dataset, NeuronSet, that contains 1163 realistic neuron meshes and 50 glial meshes.

We recall that the ultimate goal of the thesis is to facilitate simulation-driven brain microstructure estimation. Realistic neuron modeling, as the first contribution of this thesis, lays a solid foundation for achieving this goal.

2.1 Introduction to neuron tracing

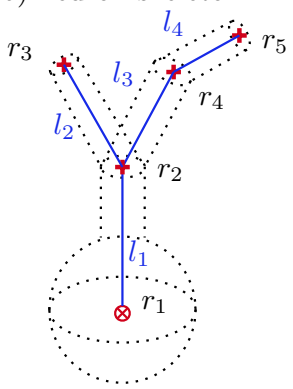
Neuron tracing or neuron reconstruction is a fundamental technique used in neuroscience to record neuronal morphology based on neuron microscopic images. Nowadays, neuron tracing has become fully digital and increasingly au-

tomated [105]. Specialized software [96, 97, 106] can semi-automatically trace neuron microscopic images to obtain 3D neuronal reconstructions [107], which are typically encoded into a tabular format called SWC[1] [108, 109].

(a) SWC format

index	type	x	y	z	r	parent
1	1	0	0	0	5	-1
2	4	0	0	8	2	1
3	4	-5	-3	10	1	2
4	4	3	5.2	9	2	2
5	4	6	3	12	1	4

(b) neuron skeleton



(c) rounded cone

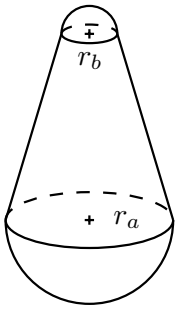


Figure 2.2: **(a)** an illustrative example of an SWC file. The example is only for illustration purposes. It does not originate from any real neurons. From left to right, the columns represent node index, type of neuronal compartment, x, y, z coordinates, radius, and parent node index, respectively. For example, the second node represents a section of apical dendrite (type 4) located at [0, 0, 8] with a radius of 2 μm. It connects to its parent (the first node). If the parent index is -1, the current node is the root. **(b)** a visual representation of the neuron skeleton defined by the SWC file in (a). The red $\otimes$ symbol represents a soma node, and neurite nodes are denoted by red $+$. The blue lines whose lengths are $l_1, ..., l_4$ sketch the neuron skeleton. The dashed lines illustrate the neuron morphology in 3D. **(c)** the rounded cone that can be used to connect two consecutive nodes.

An example of an SWC file is presented in fig. 2.2(a), including compartment types, spatial coordinates, radius, and connectivities of each node. Each row of the table defines a soma or neurite node. The first column is the node index. The second column represents the type of neuronal compartment. For instance, soma, axon, and apical dendrite are encoded by 1, 2, and 4, respectively. The following three columns are the node's coordinates. The next column is the node radius, whose interpretation depends on the type of neuronal compartment. For a neurite node, the value is the neurite radius at the node position. If the node represents a spherical soma[2], the sixth column is the sphere radius. The

[1] SWC stands for the initials of the last names of E.W. Stockley, H.V. Wheal, and H.M. Cole, who developed a system for generating morphometric reconstructions of neurons [108].

[2] Not all somas are represented as spheres. Soma format representation varies across databases. A summary of soma format representation can be found in `https://neuromorpho.org/SomaFormat.html`.

last column records the parent node index to which the current node is connected. More details about the SWC format can be found in the work of Cannon et al. [109]. Note that the term "neuron" often includes glial cells when it comes to neuron tracing. For the sake of simplicity, we keep this convention. Moreover, we refer to a digital neuronal reconstruction stored in SWC format as a "neuron skeleton" because the set of nodes defines the "skeleton" of a neuron (see fig. 2.2(b)).

Thanks to the collective effort of the neuroscience community, there are several single-neuron skeleton databases accessible to the public. The most extensive database is NeuroMorpho.Org [110], which contains around 230,000 neurons of over 40 species contributed by more than 800 laboratories worldwide. In addition to neuron skeletons, NeuroMorpho.Org provides neurons' morphometric measurements and associated metadata. Databases like NeuroMorpho.Org are employed to study synaptic integration, signal transmission, network connectivity, and circuit dynamics. Computational simulation is a promising application of neuron skeletons [107]. However, most mesh-based simulation methods require neuron models as tetrahedral volume or triangular surface meshes instead of neuron skeletons.

One can build a 3D neuron model based on an SWC file by connecting nodes with cylinders or rounded cones (see fig. 2.2(c)). Numerous software packages offer such functionality for visualization purposes [95–97]. Imperfections such as intersection, separate subcompartments, and broken surfaces are generally ignored as long as these defects do not significantly alter the morphology of neurons for visualization. However, building simulation-ready meshes is much more difficult. Next, we discuss the challenges of building simulation-ready neuronal meshes.

2.2 Challenges of neuron mesh construction

This chapter aims to model realistic neurons by triangulated surface and tetrahedral volume meshes. We concentrate on the surface mesh generation because there are well-established tools, such as Tetgen [111], Gmsh [112], and CGAL [113], that can automatically tetrahedralize surface meshes. In addition to reconstructing neuron morphology, the meshes should be suitable for diffusion MRI simulations.

A simulation-ready surface mesh should be a 2-manifold mesh without boundary edges, also known as *watertight* surface mesh [114]. More concretely, a set of triangles, i.e., a triangle soup, must form one closed surface so that the inside volume is well-defined. The defects that destroy the watertightness of a

surface mesh include T-vertex, non-manifold vertex, intersection, non-manifold edge, and hole, as illustrated in fig. 2.3 [114].

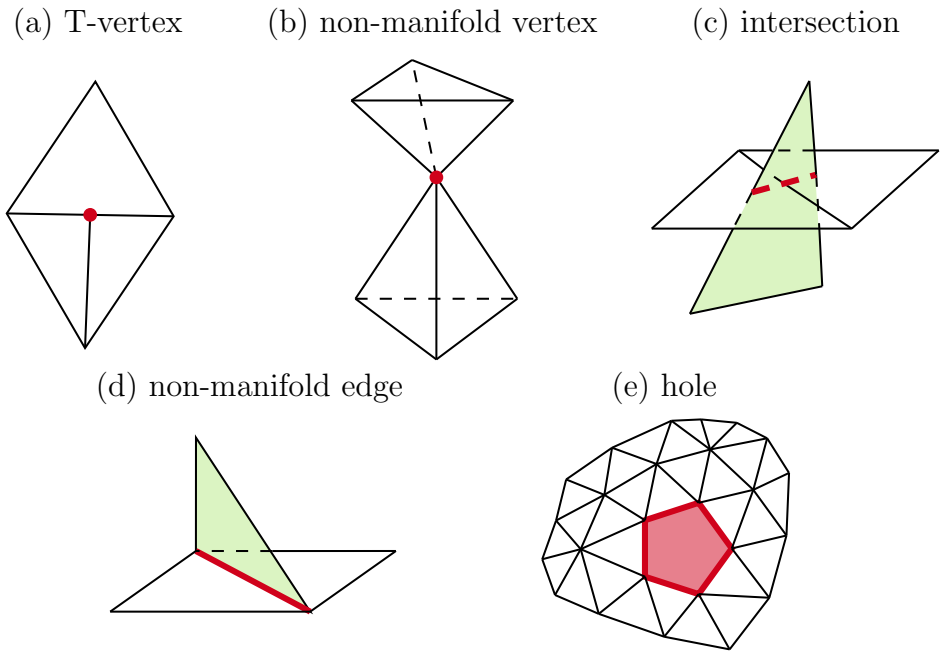


Figure 2.3: Five common defects that destroy the watertightness of a triangular surface mesh: **(a)** a T-vertex marked by a red dot. **(b)** a non-manifold vertex marked by a red dot. **(c)** the intersection of faces. **(d)** a non-manifold edge marked by a solid red line. **(e)** a hole marked by the red pentagon.

Building a watertight neuron surface mesh is highly challenging due to the complexity of neuron structure. For example, intersections inevitably exist in areas where the bifurcations occur. Those defects can be eliminated through several published tools [98, 115]. We also proposed a pipeline [116] that utilizes commercial software from the ANSA-BETA CEA system [117] to remove the mesh defects manually. The pipeline with commercial software is presented in section 2.3.

However, the meshing tools mentioned above [98, 115, 116] require manual operations through graphical interfaces to obtain a watertight neuron mesh. The manual labor could be immense if hundreds of neuron meshes are needed. The number of manual operations must be reduced to make the large-scale neuron mesh generation practical. Moreover, the number of faces (triangles) should be small[3]. Otherwise, dMRI simulations would be slow. Finally, the triangular el-

[3]Typically, the number of triangles (vertices) should be less than 300,000 (150,000) for dMRI simulations to end within a few hours.

ements should not be too elongated to minimize numerical error [118]. The quality of a triangular element can be quantified by the aspect ratio ς defined as two times the ratio of inradius and circumradius. Figure 2.4 shows the decrease of aspect ratio as triangles become elongated. Aspect ratios range from 0 (a segment) to 1 (an equilateral triangle). The triangle quality, which is a key factor affecting simulation accuracy, has been widely ignored in previous studies.

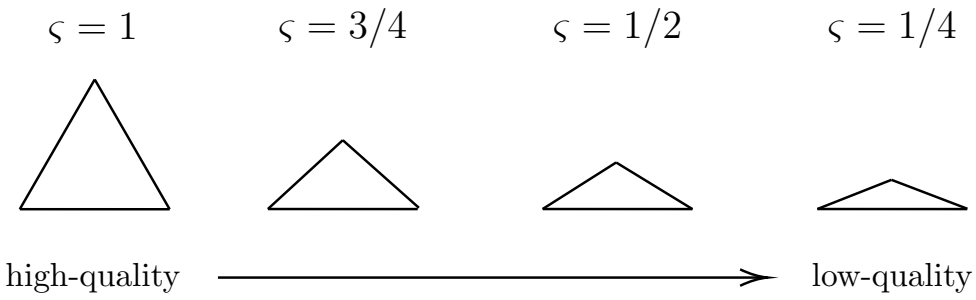


Figure 2.4: Decrease of aspect ratio as triangles become elongated.

In summary, there are four requirements for neuron surface mesh generation:

1. meshes must be watertight;
2. the mesh generation should be robust and almost automatic, requiring minimal manual operations;
3. the number of triangles should be small to reduce computational cost;
4. the triangular elements should be high-quality to minimize numerical errors.

All four requirements should be simultaneously satisfied, which is nontrivial due to the structural complexity of neurons. Specifically, neuron structure is multiscale. The neurite radius is about $1\ \mu m$, while the total neurite length can be more than $1\ mm$. These multiscale 3D structures typically require a large number of triangles. However, as mentioned above, one must make a trade-off with the number of triangles for computational efficiency.

In addition, there is another tradeoff between the number and the quality of triangles. Because neurites are typically cylindrical, triangles tend to be elongated in the axial direction as one reduces the number of triangles by simplifying a neuron mesh. We must balance triangle quantity, mesh quality, and computational efficiency. All the above difficulties add up, making automatic neuron mesh generation extremely hard. The remainder of this chapter is devoted to addressing these difficulties.

2.3 Neuron Module

Our first attempt to overcome the difficulties mentioned in the previous section involves exploiting available software. We constructed 65 realistic watertight neuron meshes using three open-source packages and commercial software. These meshes and the program related to dMRI simulations are integrated into the SpinDoctor toolbox [44] as an independent block called Neuron Module [116].

The mesh generation starts with the neuron skeletons stored in the archive *Allman* [119] in NeuroMorpho.Org. We convert the neuron skeletons to surface descriptions using two packages: *swc2vtk* [95] and *vtk2stl* [120]. These surface descriptions are problematic because they contain many intersections and proximities (see fig. 2.5, left). We used commercial software from ANSA-BETA CEA Systems [117] to manually correct and improve the quality of the neuron surface meshes (in STL format) and produced new surface triangulations (see fig. 2.5, right) that are watertight. The new surface meshes are passed into the software Gmsh [112] to obtain the tetrahedral volume meshes.

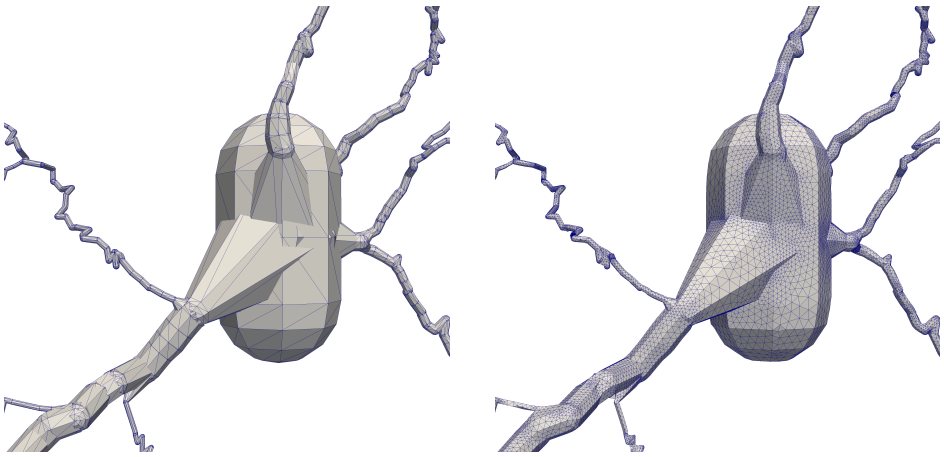


Figure 2.5: Left: a surface description of a pyramidal neuron, *02a_pyramidal2aFI*, containing many intersections and other mesh defects. Right: a watertight surface mesh with mesh defects being fixed.

Figure 2.6 summarizes the pipeline that converts neuron skeletons to the tetrahedral volume meshes in the MSH format that the users of the Neuron Module will take as the input geometrical description to perform diffusion MRI simulations. One can stop at the third step if only surface meshes are required. In Neuron Module, we provide both surface and volume meshes.

To facilitate further study, we broke the neurons into disjoint geometrical components: the soma and the dendrite branches. We manually rotated the

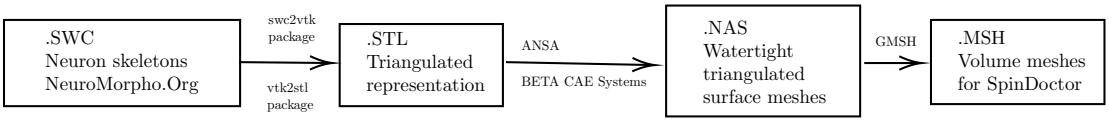


Figure 2.6: Neuron mesh generation pipeline used in Neuron Module. Neuron skeletons are converted to surface meshes (in STL format) by using *swc2vtk* [95] and *vtk2stl* [120]. Then ANSA was used to generate watertight surface meshes (in NAS format). Finally, the NAS files were converted to tetrahedral volume meshes in the MSH format by the software Gmsh [112].

tetrahedral volume mesh of a neuron so that it lies as much as possible in the x-y plane. In this orientation, we cut the tetrahedral volume mesh into sub-meshes of the soma and the dendrite branches. As an illustration, we show in fig. 2.7 the spindle neuron, *03a_spindle2aFI*[4], split into sub-meshes of the soma and the two dendrite branches.

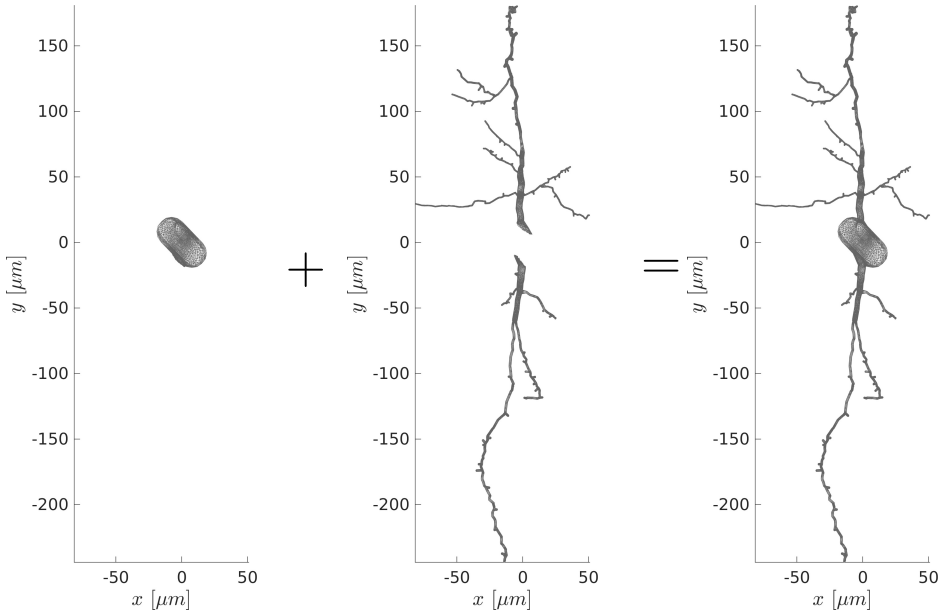


Figure 2.7: The tetrahedral volume mesh of the spindle neuron *03a_spindle2aFI* is split into three disconnected geometrical components: the soma and two dendrite branches.

In the Neuron Module, we have a group of 36 pyramidal neurons and a group of 29 spindle neurons found in the anterior frontal insula (aFI) and the anterior cingulate cortex (ACC) of the neocortex of the human brain. These neurons constitute, respectively, the most common and the largest neuron types in the hu-

[4]NeuroMorpho.Org ID of *03a_spindle2aFI* is NMO_01078.

man brain [121, 122]. They share some morphological similarities, such as having a single soma and dendrites branching on opposite sides. We list the metadata and some measurements of the 65 neurons in the appendix section A.2. Figure 2.8 presents several selected neuron meshes.

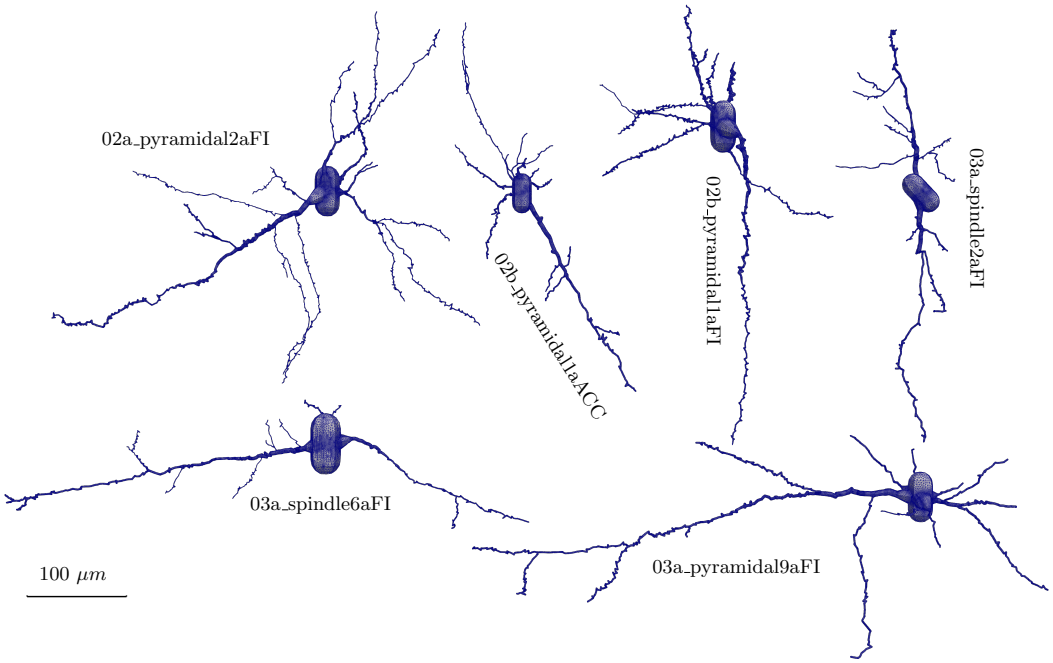


Figure 2.8: Six realistic neuron meshes in the Neuron Module.

Requiring extensive manual operations is the main drawback of the above pipeline (fig. 2.6). We repair non-watertight neuron meshes manually using commercial software. Splitting soma and neurites also requires manual annotation of the location of soma and neurites. Even though neuron meshes need to be constructed only once, the significant amount of manual operations dramatically limits the number of meshes the pipeline can build. In the next section, we present an automatic neuron mesh generator that needs minimal manual operations and can satisfy the four requirements discussed in section 2.2.

2.4 swc2mesh: an automatic mesh generator

The mesh generation pipeline used in Neuron Module started with a non-watertight surface description and produced a simulation-ready neuron mesh by manually removing the mesh defects. This is not a good strategy because

the quality of the initial mesh largely influences the performance of subsequent steps. The initial mesh can be arbitrarily "bad" and never be fixed to be watertight. This section adopts a new strategy: we first build a watertight surface mesh with a large number of triangles and then gradually simplify the mesh to reduce the mesh size. This new strategy can robustly generate simulation-ready neuron surface meshes.

2.4.1 Mesh generation pipeline

Figure 2.9 illustrates the new mesh generation pipeline. We have implemented it in a python package called *swc2mesh*[5], which will be released as an open-source project.

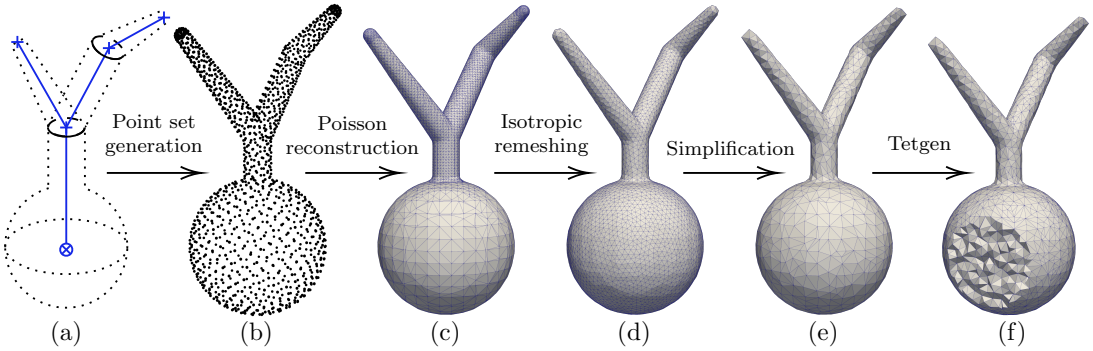


Figure 2.9: Mesh generation pipeline. **(a)** the neuron skeleton defined by the SWC file in fig. 2.2(a). **(b)** the point set densely covers the neuron's surface. Each point is associated with an appropriate out-pointing normal vector (not shown in the figure). **(c)** the watertight dense surface mesh built by the screened Poisson surface reconstruction method [123, 124]. **(d)** the isotropic mesh obtained by remeshing the dense mesh (c). Most triangles of the isotropic mesh have aspect ratios close to one (almost equilateral). **(e)** the final surface mesh obtained by simplifying the mesh (d) using an algorithm based on quadric error metrics proposed by Garland et al. [125]. **(f)** the tetrahedral volume mesh generated by Tetgen [111]. Part of the volume mesh is cut out to show the internal tetrahedra.

As the name *swc2mesh* suggests, it converts neuron skeletons in SWC format to meshes. To satisfy the four requirements discussed in section 2.2, we leverage three well-established computer graphics algorithms: the screened Poisson surface reconstruction method [123, 124], an isotropic explicit remeshing method [126–128], and a surface simplification algorithm using quadric error metrics [125].

[5] `https://github.com/fachra/swc2mesh`

The screened Poisson surface reconstruction method (or Poisson method) [123, 124] takes an oriented point set as input to produce a watertight triangulated surface mesh. Any member in the point set contains two vectors: one vector being the coordinate of a point on the desired surface and the other being the outward-pointing (from inside to outside) normal vector. The Poisson method treats the oriented point set as a vector field $\mathbf{V}$ and aims to find an indicator function χ whose gradient is $\mathbf{V}$, i.e., $\nabla\chi = \mathbf{V}$. The indicator function can be obtained by solving a Poisson equation ($\nabla \cdot \nabla\chi = \nabla\mathbf{V}$). The desired surface can be reconstructed by extracting an appropriate isosurface of χ. As long as the input point set densely covers a closed surface and each point has an appropriate normal vector, the Poisson method can ensure that the reconstructed surface mesh is watertight.

Thanks to the Poisson method, our task is reduced to creating oriented point sets covering neuron surfaces. We implemented a routine in *swc2mesh* to create oriented point sets using the Python programing language [129]. The basic idea of point set creation is connecting the nodes defined in SWC files by rounded cones (see fig. 2.2(c)). Tens of millions of points are sampled on the surfaces of cones. The samplings follow the Fibonacci lattice pattern [130] to achieve a uniform distribution.

Point set manipulation is easy because there are no connectivities between points. We can easily remove the points inside the neuron and carefully compute the normal vectors in bifurcation areas to prevent intersection. Once we obtain an oriented point set that densely covers the neuron surface, the Poisson method can produce a watertight surface mesh. The drawback is that the output mesh typically has more than one million triangles, which is excessively dense for diffusion MRI simulations. Besides, the surface mesh may contain numerous elongated triangles.

An isotropic explicit remeshing method [126–128] is then applied to the dense mesh. The remeshing method can reduce the number of triangles and improve their quality by repeatedly applying edge flip, collapse, relax and refine operations to regularize the size and aspect ratio of triangulated surface meshes. High-quality triangles are nearly equilateral, with aspect ratios close to one. On average, neuron meshes are composed of hundreds of thousands of high-quality triangles after this step.

Further mesh simplification is required to reduce the number of triangles and make subsequent dMRI simulations efficient. We employ a simplification method proposed by Garland et al. [125] implemented in PyMeshLab [127]. The algorithm adopts an edge-collapse strategy based on quadric error metrics. It allows us to aggressively simplify soma and thick neurites while keeping thin neurites almost unchanged. Moreover, we gradually apply the simplifica-

tion algorithm to meshes until they can no longer be watertight or the number of low-quality ("bad") triangles whose aspect ratios are inferior to $1/3$ exceeds 20% of the total face number. The above two numbers are determined empirically. The simplification algorithm can commonly reduce the triangle number to about 100,000.

It is worth stressing that the pipeline (from fig. 2.9(a) to fig. 2.9(e)) implemented in *swc2mesh* is fully automatic. However, due to the complexity of the neuron morphology, around 10% neuron surface meshes still need manual post-cleaning after the simplification.

Once a watertight mesh is created, Monte-Carlo dMRI simulations can directly run on it. Tetgen [111] can robustly convert watertight surface meshes to volume meshes based on constrained Delaunay tetrahedralization.

2.4.2 Implementation of swc2mesh

We implemented *swc2mesh* using the Python programing language [129] and PyMeshLab package [127]. Algorithm 1 is a pseudo code summarizing the main steps of *swc2mesh*.

The input is an SWC file defining the morphology of a neuron. We developed a parser (swc_parser) to extract the nodes stored in the SWC file. Based on these nodes, we build an oriented point set that densely covers the neuron surface. The remaining steps follow the pipeline explained in section 2.4.1. It is worth noting that we iteratively simplify the surface mesh until it can no longer be watertight or the "bad" triangle ratio is greater than 0.2. The last step helps maintain the balance between the quantity and the quality of triangular faces.

Here, we briefly present the core functions used in *swc2mesh*. By changing the input parameters, these functions can perform a variety of functionalities. In the next section, we demonstrate four essential functionalities of *swc2mesh*.

Algorithm 1: Implementation of *swc2mesh*

```
Data: swcfile (an SWC file)
Result: surfmesh (a surface mesh in the format of PLY, STL, OBJ, etc.)
begin
    nodes = swc_parser(swcfile)

    # build an oriented point set
    pointset = point_set_builder(nodes)

    # construct a watertight surface mesh
    smesh = Poisson_surface_constructor(pointset)
    smesh = isotropic_remeshing(smesh)
    surfmesh = deepcopy(smesh)

    # iteratively simplify the surface mesh
    while (bad_triangle_ratio(smesh)<0.2) and is_watertight(smesh) do
        surfmesh = deepcopy(smesh)
        # surface simplification using quadric error metrics
        smesh = QEM_simplify(smesh)
    end
    if not is_watertight(surfmesh) then
        Warning("Mesh is not watertight.")
    end

    return surfmesh
end
```

2.4.3 Functionalities of swc2mesh

This section showcases four main functionalities of *swc2mesh* and how to achieve them using Python. We sort them in the order of complexity:

1. reconstructing a complete neuron mesh with simplification routine being activated;

2. changing the shape of soma;

3. building a mesh for a particular cellular compartment;

4. building an incomplete neuron mesh with several selected cellular compartments.

First, using three lines of code, *swc2mesh* can automatically build and save a neuron surface mesh based on an input SWC file. Listing 2.1 demonstrates a simple use case for constructing a single neuron mesh with the simplification routine being activated. It is worth noting that our package provides integrated functions to the users. Only the input and output files and appropriately selected settings are required from the users.

The saved surface mesh is in the Polygon file format (PLY). Other formats, such as STL, OBJ, OFF, are also supported. We refer to the documentation[6] of PyMeshLab [127] for an exhaustive list of supported formats.

```
from swc2mesh import Swc2mesh
mesh = Swc2mesh("neuron_skeleton.swc")
mesh.build("neuron.ply", simplification=True)
```

Listing 2.1: A code snippet for generating a neuron surface mesh. The input is an SWC file named "neuron_skeleton.swc". The generated neuron surface mesh is saved in a file named "neuron.ply". The mesh generator automatically determines the mesh format according to the file extension. In this case, the mesh format is PLY.

In fig. 2.10, a representative neuron surface mesh overlaps the original microscopic image of the neuron to demonstrate the realisticity of the reconstructed neuron mesh. The Allen Institute for Brain Science [8] provides the microscopic image of the neuron[7]. It is found that the triangulated surface mesh digitally reproduces the neuron morphology with extreme precision. For visualization purposes, different colors are assigned to each neuronal compartment. The choice of color follows the convention adopted by NeuroMorpho.Org[8]. When coupling the neuron meshes with advanced numerical simulators, we could expect different physical properties, e.g., permeability, to be assigned to the sub-regions.

Second, *swc2mesh* supports four shapes of somas: sphere, ellipsoid, cylinder, and customized contour. Soma format representation varies across databases and has more than four types of shapes. A summary of soma format representation can be found in the frequently asked questions (FAQs) in NeuroMorpho.Org[9]. The first three shapes, known as single contours, account for 80 percent of the neuron skeletons archived in NeuroMorpho.Org. We give an example for each shape in fig. 2.11. A soma whose shape is customized contour is defined as a group of stacked cylinders. The convex hull of the group of cylinders forms

[6] https://pymeshlab.readthedocs.io/en/latest/io_format_list.html#save-mesh-parameters

[7] http://celltypes.brain-map.org/mouse/experiment/morphology/545612828

[8] Please check the FAQ in NeuroMorpho.Org: How are different parts of the cells color coded in visualization?

[9] Please check the FAQ in NeuroMorpho.Org: How is the soma format represented in the standardized (CNG.swc) files?

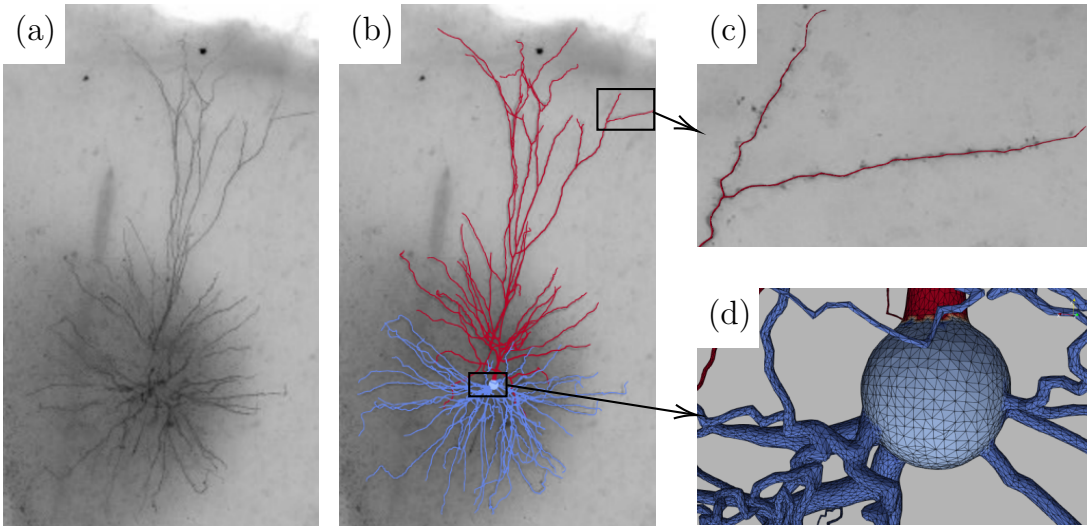


Figure 2.10: Overlapping of a real human neuron mesh with its original microscopic image. **(a)** the neuron's microscopic image. In the Allen Institute Cell Types database [8], the neuron ID is 545612828, and its donor is H16.03.010. The neuron ID in NeuroMorpho.Org is NMO_102562. **(b)** the neuron surface mesh overlapped on the microscopic image. The red neurites are apical dendrites, the purplish blue neurites are basal dendrites, and the bright blue part is the soma. The mesh digitally reproduces the morphology of the real neuron. **(c)** the end of a neurite. The neuron mesh perfectly reproduces the finest neurite structure. **(d)** some neurites around the soma. Note that the neuron surface is triangulated.

the customized contour. The three examples of single contours in fig. 2.11 correspond to a pyramidal neuron, 07b_pyramidal14aACC[10], in the archive *Allman* in NeuroMorpho.Org. The example of customized contour corresponds to a rat cell[11] in the archive *Spruston*.

With *swc2mesh*, changing the soma shape is easy. Users can specify the keyword argument `soma_shape` to control the soma shape (see listing 2.2).

```
from swc2mesh import Swc2mesh
# the keyword argument some_shape has four options:
# "sphere" (default), "ellipsoid", "cylinder", and "contour"
mesh = Swc2mesh("neuron_skeleton.swc", soma_shape="contour")
mesh.build("neuron_with_customized_soma.obj", simplification=
True)
```

Listing 2.2: A code snippet for generating a neuron whose soma is defined by a customized contour. The neuron mesh is saved in OBJ format.

[10]The NeuroMorpho.Org ID of 07b_pyramidal14aACC is NMO_01066.
[11]The rat cell is ri05 whose NeuroMorpho.Org ID is NMO_00885

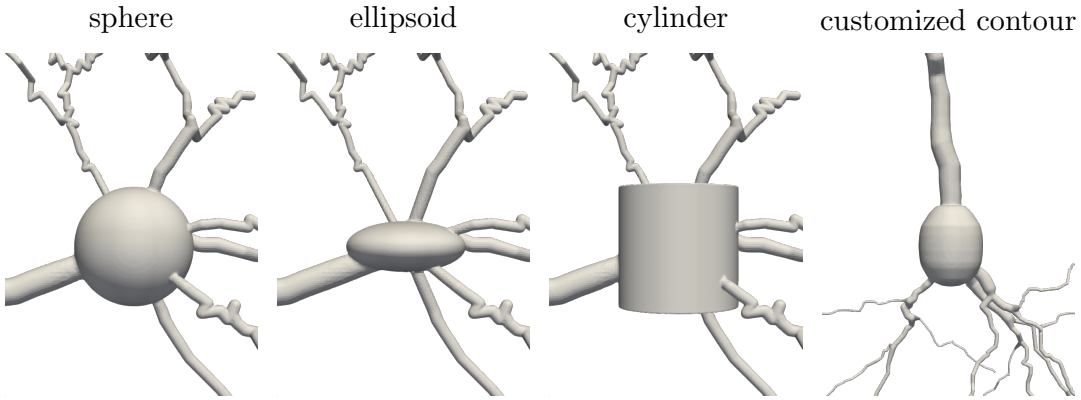


Figure 2.11: Four types of soma shapes supported by *swc2mesh*.

Third, *swc2mesh* can build a mesh for a particular cellular compartment. For example, users can construct a mesh for a single dendritic arborization. Line 7 of listing 2.3 presents the way to build meshes of apical dendrites. It is also possible to separately build a group of meshes for all cellular compartments by setting the keyword argument `compartment` to `"all"` (see line 10 of listing 2.3). In fig. 2.12, we demonstrate a neuron mesh as a whole and its cellular compartmental meshes. We note that splitting compartments requires manual operations in Neuron Module, while it is fully automatic using *swc2mesh*.

```
1  from swc2mesh import Swc2mesh
2  mesh = Swc2mesh("neuron_skeleton.swc", soma_shape='ellipsoid')
3
4  # other compartments include: "undefined", "soma", "axon",
5  # "basal_dendrite", "apical_dendrite", "custom",
6  # "unspecified_neurites", "glia_processes"
7  mesh.build("apical_dendrites.stl", simplification=True,
   compartment="apical_dendrite")
8
9  # build a group of meshes for all cellular compartments
10 mesh.build("cell_compartments.stl", simplification=True,
   compartment="all")
```

Listing 2.3: A code snippet for generating cellular compartmental meshes. All meshes are saved in STL format.

The fourth functionality is that *swc2mesh* allows us to subtract cellular compartments from a neuron or construct an incomplete neuron using several selected compartments. Listing 2.4 demonstrates two examples. The syntax "cell-apical_dendrite" indicates the subtraction of apical dendrites from the whole neuron. Similarly, the syntax "soma+apical_dendrite" indicates that we build an incomplete neuron mesh using only soma and apical dendrites. The two exam-

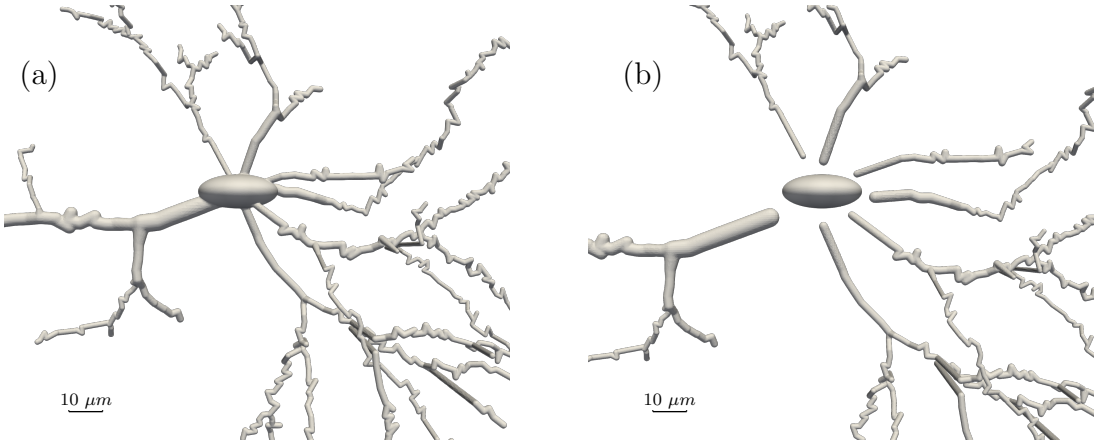


Figure 2.12: **(a)** a neuron mesh as a whole. All cellular compartments are connected. **(b)** the meshes of cellular compartments automatically generated by *swc2mesh*. Each compartment has an individual surface mesh. They are disconnected.

ples are illustrated in fig. 2.13, respectively.

```
from swc2mesh import Swc2mesh
mesh = Swc2mesh("neuron_skeleton.swc", soma_shape='ellipsoid')

# subtract apical dendrites from the neuron
mesh.build("incomplete_neuron_1.ply", simplification=True,
compartment="cell-apical_dendrite")

# build an incomplete neuron using soma and apical dendrites
mesh.build("incomplete_neuron_2.ply", simplification=True,
compartment="soma+apical_dendrite")
```

Listing 2.4: A code snippet for generating incomplete neuron meshes.

We demonstrate the excellent modeling capabilities of *swc2mesh* through the above use cases. We believe the neuron mesh generator can facilitate diffusion MRI simulations and thus help understand the formation of intracellular signals and develop methods for microstructure imaging. Next, we present the first application of *swc2mesh*, which is building a large-scale neuron mesh dataset.

2.5 NeuronSet: a large-scale neuron mesh dataset

With the neuron mesh generator, we can build a large-scale neuron mesh dataset by converting neuron skeletons stored in NeuroMorpho.Org to meshes. We refer to the neuron mesh dataset as "NeuronSet", which contains 1213 real-

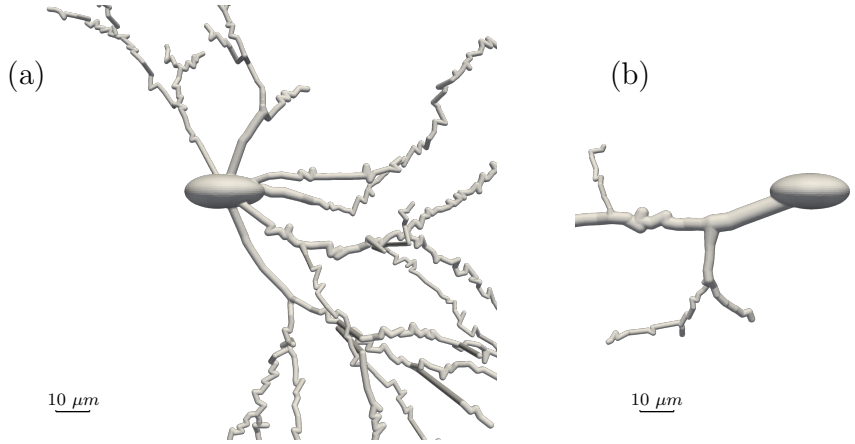


Figure 2.13: **(a)** an incomplete neuron mesh constructed by subtracting an apical dendrite from the neuron. The mesh corresponds to "incomplete_neuron_1.ply" in listing 2.4. **(b)** an incomplete neuron mesh built with only soma and an apical dendrite. The mesh corresponds to "incomplete_neuron_2.ply" in listing 2.4.

istic human cellular meshes, including 1163 neurons and 50 glia[12]. Each neuron mesh corresponds to a real human neuron or glial cell. These cells were independently traced by 11 laboratories, stored in 11 archives, and reported on 22 papers [8, 119, 131–150]. Table 2.1 lists the number of neurons in each archive included in NeuronSet.

Table 2.1: The number of cells in each archive included in NeuronSet.

Archive	Number of cells in NeuronSet	Total number of cells in archive	Cell types	Age classes	References
Allen Cell Types	161	264	neurons	adult	[8, 134, 137]
Allman	65	65	neurons	adult	[119]
DeFelipe	126	126	neurons	adult	[133, 138]
Falcone	50	50	glia	adult, infant	[136, 139]
Hrvoj-Mihic_Semendeferi	85	85	neurons	adult	[135]
Jacobs	505	2649	neurons	neonatal, adult, old	[132, 140–146]
Lewis	142	142	neurons	adult	[131]
Linaro	14	14	neurons	not reported	[147]
Segev	6	6	neurons	adult	[148]
Semendeferi_Muotri	45	45	neurons	adult	[149]
Vuksic	14	14	neurons	infant	[150]

Each archive has a different protocol for neuron tracing. Key differences between archives are related to the soma format and the integrity of axons. To unify these archives, we make the following choices.

[12]NeuronSet is publicly available at `https://github.com/fachra/NeuronSet`.

First, soma shapes are fixed to be spherical in NeuronSet because most somas are initially saved as spheres in NeuroMorpho.Org. For example, neurons in archives *Allen Cell Types*, *Hrvoj-Mihic_Semendeferi*, *Defelipe*, and *Falcone* have spherical somas.

Second, long axons are not modeled by the cellular meshes. Of the eleven archives, only the *Allen Cell Types* provides incomplete axons. Indeed, axonal arborization is highly complex and variable. Some axons can extend as long as 1 m [6]. The diameter of unmyelinated axons in the mammalian brain varies between 0.08 and 0.4 μm [151, 152]. Axons of hippocampal CA3 pyramidal cells have around 150 bifurcation points with a total axonal length of 150-300 mm [153–155]. Moreover, a single cell may contact 30,000-60,000 neurons through axon [6]. The digital reconstruction of realistic axonal arborization is still challenging for neuron tracing, not to mention generating surface meshes. For diffusion MRI application, long axons are believed to have a small impact on the measured intracellular diffusion [88, 156]. We, therefore, do not include axons in modeling.

In the following sections, we first display several representative cellular meshes in the dataset. Then some statistics about mesh quality and mesh size are given. Finally, we conduct neuroanatomical measurements on the cellular meshes.

2.5.1 Mesh visualization and statistics

We demonstrate our realistic neuron models by comparing a reconstructed neuron mesh with its original microscopic image in fig. 2.10. Here, we present one neuron mesh for each archive in fig. 2.14. Glial cells in the archive *Falcone* are tiny compared to the neurons. For clarity, we show them separately in fig. 2.15.

In addition to being realistic, all neuron surface meshes are watertight and simulation-ready. The mesh quality, a commonly disregarded factor affecting the simulation precision, has been considered. We refer to a triangle as low-quality or "bad" if its aspect ratio is inferior to $1/3$. For each triangulated surface mesh, we evaluate the mesh quality using the proportion of bad triangles, i.e., the number of bad triangles divided by the total number of triangles. Figure 2.16(a) is the histogram of the bad triangle proportion for all meshes in NeuronSet. Seventy-five percent of neuron meshes have a bad triangle proportion below 0.2, meaning that most meshes generated by *swc2mesh* have good quality.

While maintaining the mesh quality, we reduce the mesh size as much as possible using the simplification routine described in section 2.4. Figure 2.16(b) and fig. 2.16(c) are the histograms of the number of triangles and vertices. The maximum number of triangles (vertices) is 300,000 (150,000).

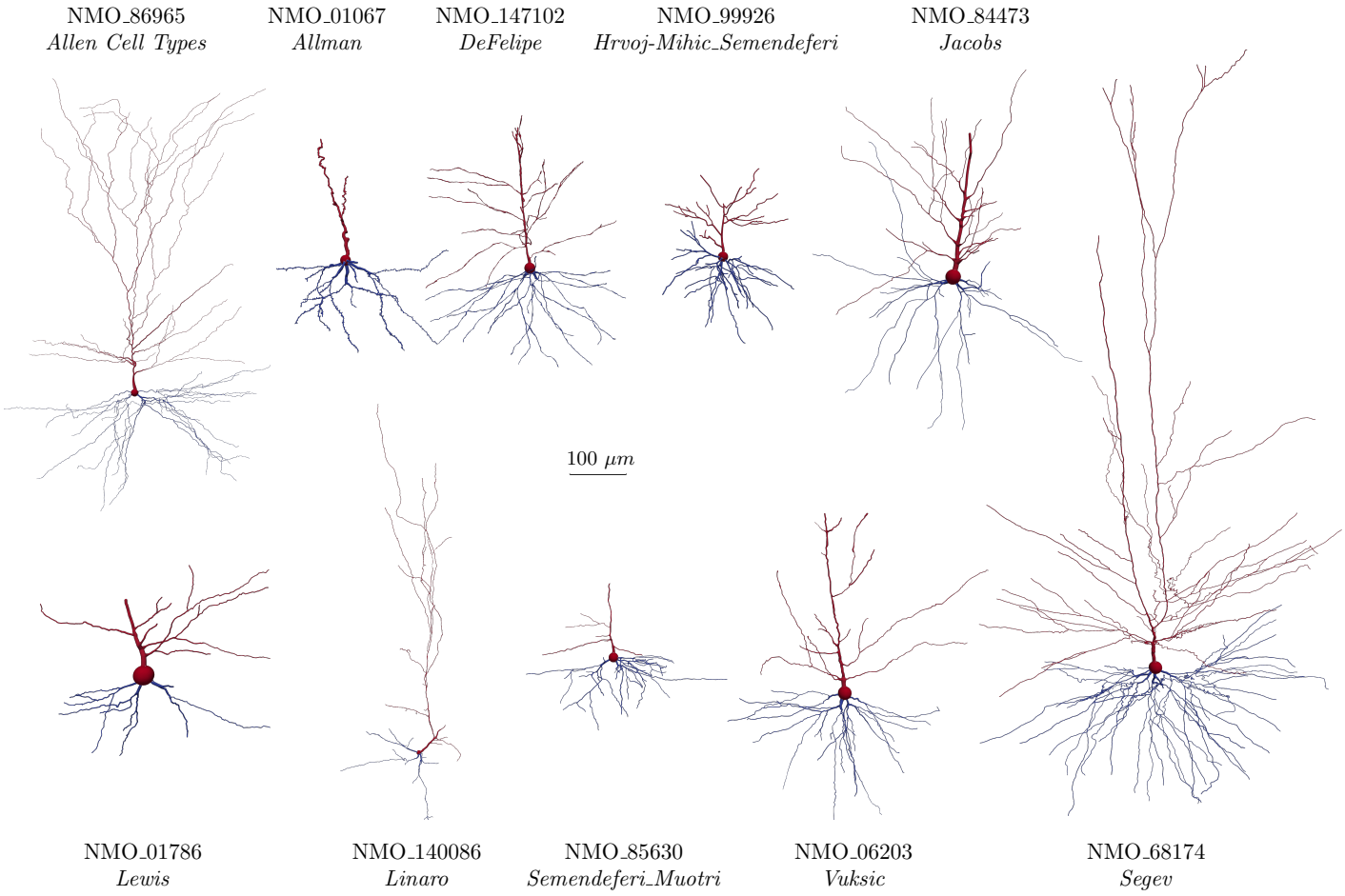


Figure 2.14: Visualization of one representative neuron surface mesh for each archive. The NeuroMorpho.Org IDs and the archive names are annotated around the neurons.

Besides, cellular meshes in NeuronSet correspond to human cells extracted from a wide range of brain regions. Figure 2.17 shows the regional distribution of the cells included in NeuronSet.

2.5.2 Neuroanatomical measurements

The neuron meshes and skeletons enable the measurements of various neuroanatomical parameters. L-measure[13] is widely used to measure neuron morphology based on skeletons [157]. However, L-measure cannot accurately measure neurite area and volume because it treats neurite segments as cylinders. For example, the neurite area given by L-measure is the sum of the lateral area of many cylinders. This simplified treatment omits the spatial variation of neu-

[13] `http://cng.gmu.edu:8080/Lm/`

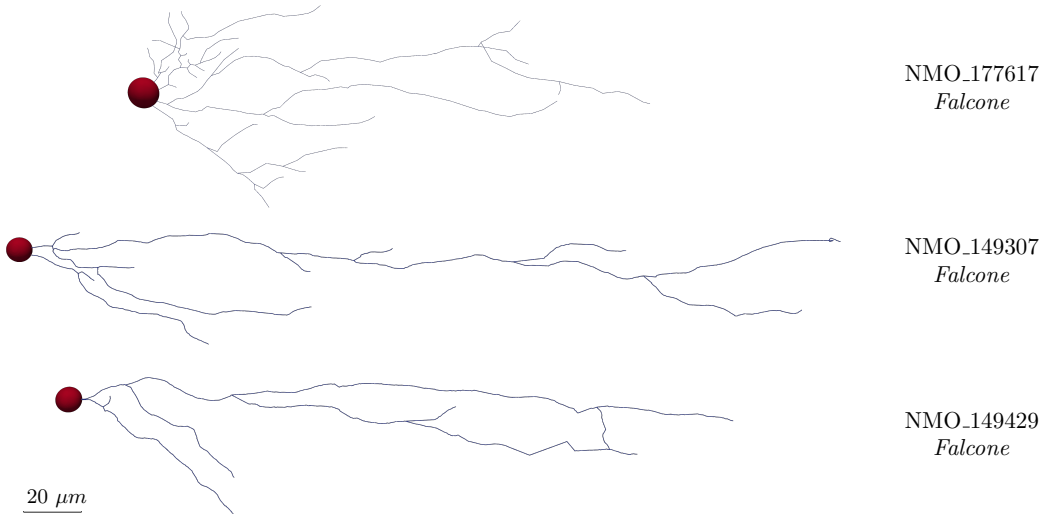


Figure 2.15: Visualization of three glial meshes in the archive *falcone*. The NeuroMorpho.Org IDs and the archive name are annotated around the cells.

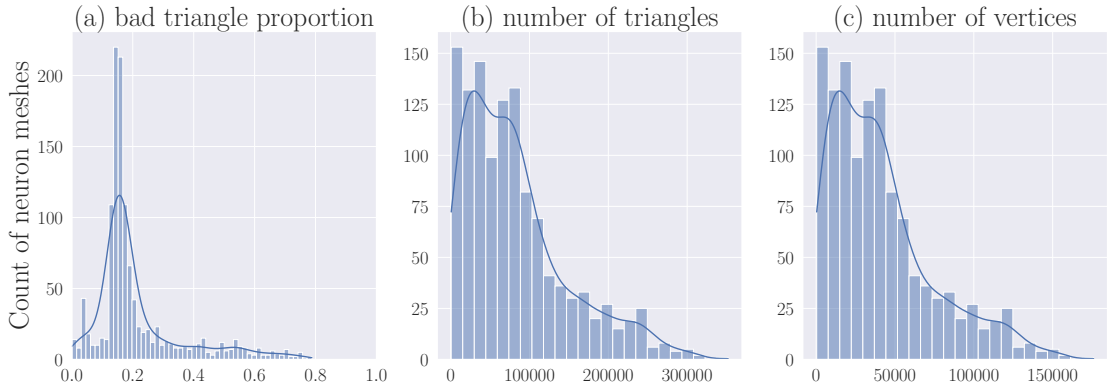


Figure 2.16: Distribution of cellular mesh quality and size in NeuronSet. **(a)** histogram of bad triangle proportion. 75% meshes have a bad triangle proportion below 0.2. **(b)** histogram of triangle numbers of meshes. The maximum number is 300,000. **(c)** histogram of vertex numbers of meshes. The maximum number is 150,000.

rite radius and the complex structure in bifurcation regions. We remedy the defect by measuring the surface and volume on neuron meshes. We can get over 30 neuroanatomical parameters of biophysical interest based on neuron meshes and skeletons. In this section, we present how to measure the area, volume, soma radius, and neurite length using the example given in fig. 2.2. The definitions of other neuroanatomical parameters are listed in the appendix sec-

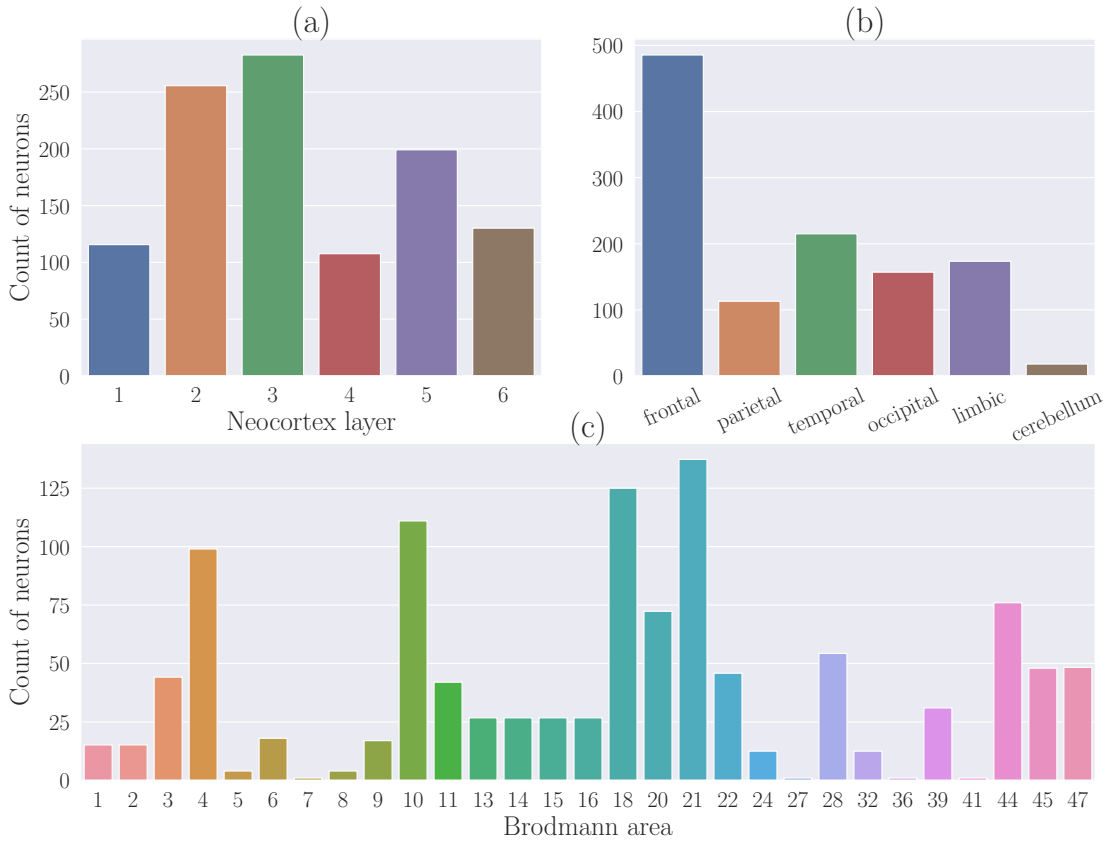


Figure 2.17: Cell distribution in brain regions. **(a)** distribution in neocortex layers. **(b)** distribution in brain lobes. **(c)** distribution in Brodmann areas.

tion A.1.

The soma radius r_{soma} is recorded in SWC files (typically in the sixth column of the first row). Stems are the branches attached to a soma. We denote by N_{stems} the number of stems. The sum of the distances between nodes subtracted by $N_{\text{stems}} \times r_{\text{soma}}$ is the total neurite length L_{neurite}. For example, fig. 2.2(b) has one stem ($N_{\text{stems}} = 1$) and the total neurite length is $\sum_{i=1}^{4} l_i - N_{\text{stems}} \times r_{\text{soma}}$.

With a watertight neuron surface mesh, the area A_{neuron} and volume V_{neuron} can be efficiently computed [158]. Besides, the soma area is $A_{\text{soma}} \simeq 4\pi r^2$ and volume is $V_{\text{soma}} \simeq \frac{4}{3}\pi r^3$. It is straightforward that the area of neurites is $A_{\text{neurite}} = A_{\text{neuron}} - A_{\text{soma}}$ and volume is $V_{\text{neurite}} = V_{\text{neuron}} - V_{\text{soma}}$.

These direct measurements give rise to some secondary neuroanatomical parameters. For example, the neuronal neurite area (volume) fraction can be defined as $A_{\text{neurite}}/A_{\text{neuron}}$ ($V_{\text{neurite}}/V_{\text{neuron}}$).

Finally, we show the distributions of these neuroanatomical parameters in

fig. 2.18. The distributions are consistent with a large-scale study of brain cell morphometry also conducted on NeuroMorpho.Org [159].

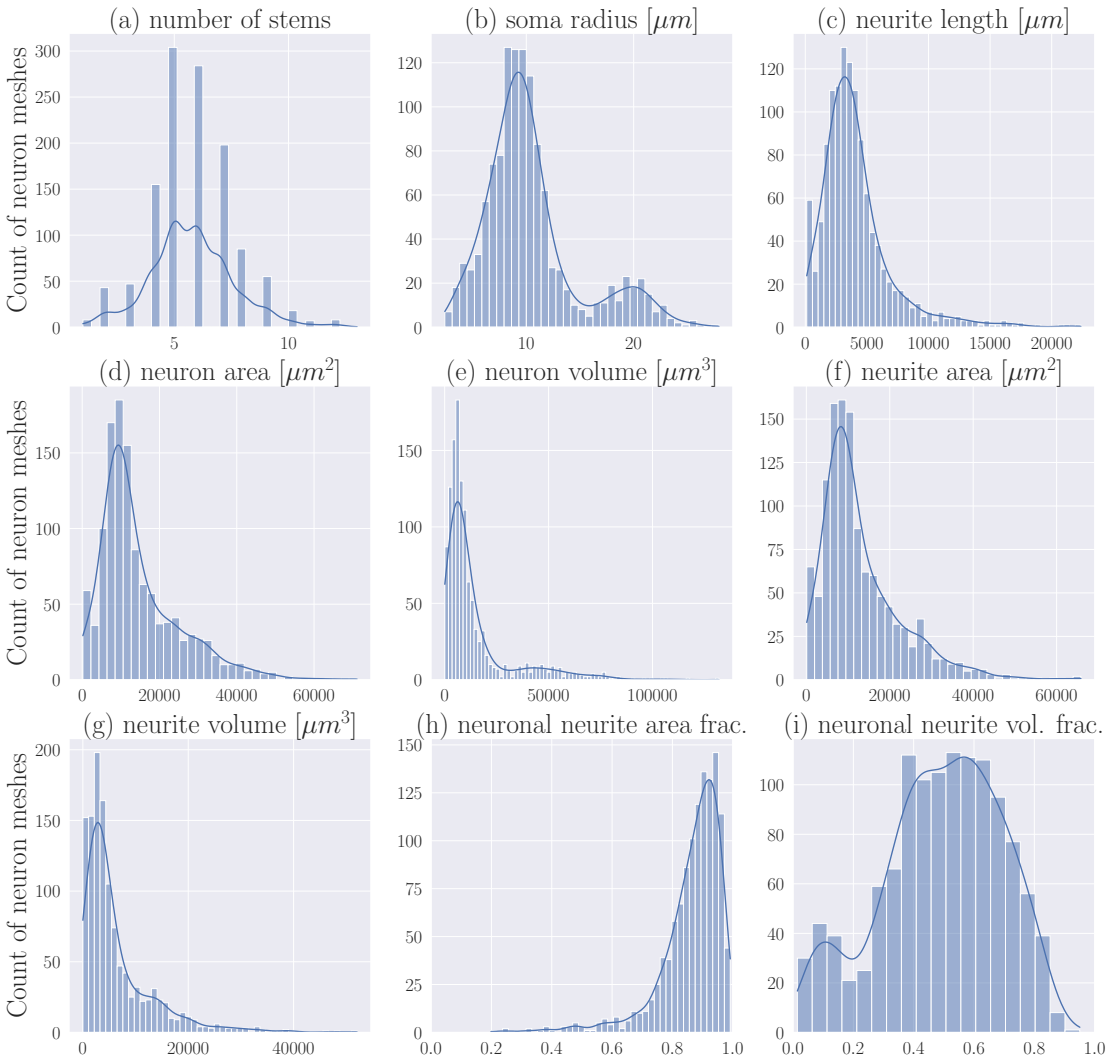


Figure 2.18: Distribution of some neuroanatomical parameters for the mesh database.

2.6 Summary

Sophisticated numerical phantoms, the indispensable elements required by diffusion MRI simulations, have been lacking in the community. Hence, some numerical simulations are limited to spherical and cylindrical geometries whose phantoms are easy to build. For the same reason, dMRI simulations are still

treated as just a basic tool to validate some simplified biophysical models. Even though diffusion MRI simulations based on the BT equation or Brownian motion are more accurate than biophysical models, the lack of realistic numerical phantoms substantially restricts the widespread use of simulation methods. As the first contribution of this thesis, we provide a large number of realistic cellular meshes and a high-performance open-source neuron mesh generator to the community. We emphasize that these neuron meshes are compatible with Monte Carlo and finite element simulation methods.

Our first attempt is Neuron Module which includes 65 realistic neuron meshes. They are enough for gaining insights into diffusion MRI physics. In the next chapter, we will demonstrate several usages of the Neuron Module by conducting numerical simulations on the meshes. Nonetheless, Neuron Module is insufficient.

The main drawback of the mesh generation pipeline used in the Neuron Module is the low automation level. We implemented an open-source neuron mesh generator, *swc2mesh*, using three well-established computer graphics algorithms to overcome the drawback. As the name suggests, *swc2mesh* converts neuron skeletons to simulation-ready meshes. Additionally, we have shown that *swc2mesh* is automatic, robust, versatile, and easy to use. Finally, we built a large-scale neuron mesh dataset, NeuronSet, as the first application of the mesh generator. Meshes in NeuronSet are of good quality and appropriate size, and they play an essential role in the new microstructure imaging method proposed in chapter 5.

We cite two previous works in [98, 160] that describe new algorithms for generating relevant tissue and cell geometries for diffusion MRI simulations. These two works are similar in spirit to ours, namely, the common idea is to provide synthetic but realistic cell/tissue geometries.

Regarding the synthetic tissue/cell mesh generation problem, the work in [160] is more about the brain white matter. The work closer to ours is [98], which is about creating 3-dimensional neuron meshes based on neuron skeletons. That paper contained detailed information about generating artificial neuron skeletons, which do not correspond to actual neurons but are analogous to realistic neuron skeletons obtained by tracing real neurons. In addition, they used the software Blender[14] and a Blender add-on "SWC Mesh"[15] to generate neuron surface meshes. The salient points of our work, contrasted with [98], are the following:

1. *swc2mesh* aims to convert neuron skeletons in SWC format to meshes. Generating artificial neuron skeletons is not our objective. We note that

[14] `https://www.blender.org/`
[15] `https://github.com/mcellteam/swc_mesher`

realistic neuron skeletons from NeuroMorpho.Org and artificial neuron skeletons in SWC format are all acceptable inputs of *swc2mesh*.

2. *swc2mesh* is robust, automatic, versatile, and user-friendly. Unlike Blender, *swc2mesh* does not require manipulation in a graphical interface, making large-scale mesh generation possible.

3. The entire mesh generation pipeline, including the mesh simplification routine, is already open source.

4. We provide the simulation-ready realistic neuron meshes in publicly available repositories.

However, we have not entirely solved the problem regarding numerical phantom generation. The final goal is to build brain voxel phantoms. For that, we need to densely pack the neurons so that ECS has a reasonable volume fraction and neurons must not intersect. The neuron packing is still an open question that is probably more challenging than neuron mesh generation, especially considering that the width of ECS can be as narrow as $30\,nm$ [161, 162]. Nonetheless, *swc2mesh* is a substantial advance toward brain voxel phantom generation. Even though we cannot wrap various neurons with an ECS compartment yet, it is possible to cover a single neuron mesh with a thin envelope to achieve ECS modeling with reasonable volume fractions. Figure A.1 and fig. A.2 in section A.3 give an example of a neuron wrapped by an ECS whose volume fraction is 31%.

Another major challenge that prevents us from reaching the ultimate goal of this thesis is the low efficiency of dMRI simulators. For example, our work [116] shows that a FEM simulator needs around 10 seconds to compute a single dMRI signal using a realistic neuron mesh. To leverage data-driven methods, hundreds of dMRI signals are needed for each neuron. It requires several months to finish the simulations on all neurons in NeuronSet, which is impractical. The next chapter presents two simulation approaches: the finite-element method and the numerical matrix formalism method. Numerical matrix formalism proposed by Li et al. [49, 50] is a method that implements classic matrix formalism [47, 48] on FE discretization. Integrating matrix formalism with a finite element method brings significant potential for improving computing efficiency. The second contribution of this thesis, the improvement of dMRI simulation efficiency, will be presented in the next chapter.

Chapter 3

Solving Bloch-Torrey Equation with Finite Element Discretization

One main objective of diffusion MRI is to non-invasively probe the brain microstructure by encoding the motion of water protons with externally applied magnetic field gradient [18, 163]. This is highly challenging due to the structural complexity of brain tissue and the intricate diffusion MRI mechanism. Existing methods that estimate brain microstructure properties mostly rely on explicit forward models relating the dMRI signals with some microstructure properties. These forward models often require various assumptions and simplified geometrical representations of brain tissue, which are not always valid. A gold-standard forward model for describing the dMRI signal formation mechanism is the Bloch-Torrey partial differential equation (BT equation).

The BT equation describes the time evolution of magnetization carried by spin-bearing particles inside a medium under the influence of diffusion, externally applied magnetic field gradient, relaxation, and boundary restriction. The mathematical formulation of BT equation will be presented in section 3.1.

Solving the BT equation is an effective way to simulate dMRI signals from brain tissue. The previous chapter geometrically modeled the most important components of brain tissue, i.e., neurons and glia. This chapter concerns simulating dMRI signals from them.

The predominant numerical methods of solving the BT equation include the finite element method (FEM) [44–46], the Matrix Formalism (MF) method [47–49], and the Monte-Carlo method [36–42]. The FEMs have demonstrated their potential, including in high-performance computing settings [44, 49, 56, 57] and in manifolds settings for thin-layer and thin-tube geometries [58]. The Matrix Formalism method [47, 48], which decomposes the BT equation onto a Laplacian eigenbasis, provides a compact matrix formulation of dMRI signals.

Monte-Carlo methods numerically simulate the motion of random walkers

in a geometrical configuration. Software packages of Monte-Carlo methods include *Camino* [64], *DIFSIM* [38], and *disimpy* [65].

Section 3.2 introduces a finite element method and a matrix formalism method implemented in the SpinDoctor toolbox [44]. Contrary to classic MF, which is limited to simple geometries, SpinDoctor's MF implementation relies on finite element (FE) discretization of the computational domain. The combination of MF with FE discretization enables MF simulation on complex geometries. We refer to SpinDoctor's MF implementation as numerical matrix formalism (numerical MF).

The main shortcoming of existing simulation methods is the high computational cost. Current methods may take several months to run dMRI simulations on every neuron mesh in NeuronSet. We succeeded in improving the computational efficiency of the numerical MF by a factor of ten. Compared with other simulation methods, the new numerical MF is **20** times faster than SpinDoctor's FEM implementation and **65** times faster than *disimpy* [65]. With the new numerical MF, we finished the simulations on 1213 neuron meshes in a few weeks[1]. The optimization of the numerical MF is the second contribution of this thesis that will be presented in section 3.3.

To illustrate some potential uses of dMRI simulations, in section 3.4, we show numerical examples of the simulated diffusion MRI signals in multiple diffusion directions from whole neurons as well as from the soma and dendrite branches. The segmentation of neuron meshes, shown in fig. 2.7, allows us to study the water exchange inside a neuron and the power-law scaling pattern of dendrite signals. Furthermore, by performing dMRI simulations on the 65 neurons in Neuron Module, we show that six markers defined on the simulated dMRI signals can be related to the soma size. The preliminary studies in section 3.4 inspire us to develop a simulation-based brain microstructure imaging method that will be presented in chapter 5.

3.1 Bloch-Torrey equation

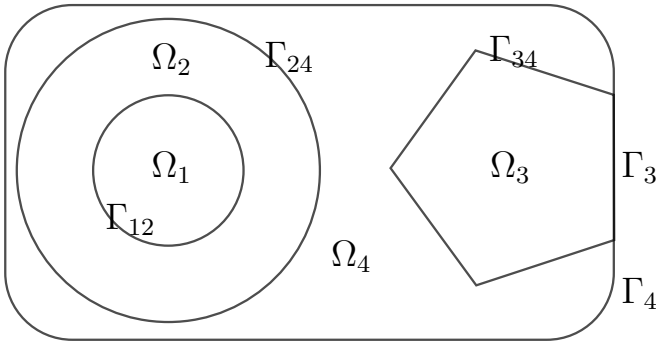
This section explains the theoretical framework of diffusion MRI simulation. We specify the geometrical configuration, the formulation of BT equations, and the diffusion-encoding sequences.

Suppose one would like to simulate dMRI signals due to water protons inside a medium. A domain $\Omega = \cup_{i=1}^{N} \Omega_i$ describes the medium that comprises

[1] We did not record in detail the time used for the simulation alone. However, in just three weeks, we completed the generation of the 1213 neuron meshes, the code development for an automated simulation pipeline, and the computation of dMRI signals for every neuron mesh.

N non-overlapping compartments. Ω_i denotes the *i*-th compartment which is a connected subset of $\mathbb{R}^n$ (n is 2 or 3), and $\partial\Omega_i$ its boundary. The interface between two compartments is denoted by $\Gamma_{ij} = \Gamma_{ji} = \Omega_i \cap \Omega_j$ for $i \neq j$, and $\Gamma_{ij} = \emptyset$ if $i = j$. The outer boundary of the whole medium is $\partial\Omega$ and of the *i*-th compartment is $\Gamma_i = \partial\Omega_i \cap \partial\Omega$. Figure 3.1 illustrates a 2-dimensional example of the geometrical configuration. Throughout this chapter, a domain Ω represents a 3-dimensional artificial brain voxel of $N-1$ intracellular compartments (ICCs) wrapped by an extracellular space (ECS), and the boundaries and interfaces model cellular membranes.

Typical numerical descriptions of the geometrical configuration are polygon meshes. For example, the neuron meshes constructed in chapter 2 serve as the intracellular compartments in numerical simulations.



$$\Omega = \bigcup_{i=1}^{4} \Omega_i \ , \ \partial\Omega = \Gamma_3 \cup \Gamma_4$$

Figure 3.1: A 2-dimensional example of a rectangular domain Ω composed of four compartments. Ω_1 is the disk surrounded by the ring Ω_2. The pentagon Ω_3 is the third compartment. The last compartment Ω_4 is the remaining area in the rounded rectangle. The small circle Γ_{12} is the interface between Ω_1 and Ω_2. The big circle Γ_{24} is the interface between Ω_2 and Ω_4. The rightmost edge of the pentagon, Γ_3, is the outer boundary of the third compartment, and the other edges Γ_{34} are the interface between Ω_3 and Ω_4. Γ_4 is the outer boundary of the fourth compartment. $\Gamma_1, \Gamma_2, \Gamma_{13}, \Gamma_{14}$, and Γ_{23} are empty sets.

3.1.1 General form of BT equation

In diffusion MRI, a time-varying magnetic field gradient is applied to encode the diffusive motion of water protons. Denoting the effective diffusion-encoding magnetic field gradient by $\boldsymbol{G}(t)$, the complex transverse water proton magnetization in the rotating frame satisfies the Bloch-Torrey partial differential equation (BT equation) [33]:

$$\frac{\partial}{\partial t} M_i(\boldsymbol{x}, t) = \left(\nabla \cdot D_i \nabla - \imath\gamma \boldsymbol{x} \cdot \boldsymbol{G}(t) - \frac{1}{T_i} \right) M_i(\boldsymbol{x}, t), \ \boldsymbol{x} \in \Omega_i, \ t \in [0, TE], \quad (3.1)$$

where D_i and T_i are respectively the self-diffusion coefficient and transverse relaxation time of water protons in the compartment Ω_i ($i \in \{1, ..., N\}$), $\gamma = 0.26752\ rad/(\mu s \cdot mT)$ is the gyromagnetic ratio of the water proton [164], TE is the echo time, and $\imath$ is the imaginary unit. Magnetization, the density of proton transverse magnetic moment, is a complex-valued function of position $\boldsymbol{x}$ and time t. M_i denotes the magnetization in Ω_i. The externally applied magnetic field gradient $\boldsymbol{G}(t)$ encodes the water protons' displacements by making the magnetization out of phase.

The initial magnetization is assumed to be equilibrial in the brain voxel:

$$M_i(\boldsymbol{x}, 0) = \rho,\ \boldsymbol{x} \in \Omega_i,\ i \in \{1, ..., N\}, \tag{3.2}$$

where ρ is the initial magnetization constant.

The outer boundaries are subject to surface relaxation [165, 166], which gives the boundary conditions:

$$D_i \nabla M_i(\boldsymbol{x}, t) \cdot \boldsymbol{n}_i(\boldsymbol{x}) = \kappa_i M_i(\boldsymbol{x}, t),\ \boldsymbol{x} \in \Gamma_i,\ i \in \{1, ..., N\}, \tag{3.3}$$

where $\boldsymbol{n}_i$ is the unit outward pointing normal vector of $\partial\Omega_i$, and κ_i is the surface relaxivity of Γ_i. Finally, we account for the permeability and guarantee the flux continuity across the interfaces through the interface conditions [50, 167]:

$$D_i \nabla M_i(\boldsymbol{x}, t) \cdot \boldsymbol{n}_i(\boldsymbol{x}) = \kappa_{ij}(M_j(\boldsymbol{x}, t) - M_i(\boldsymbol{x}, t)),\ \boldsymbol{x} \in \Gamma_{ij}, \tag{3.4}$$

$$D_i \nabla M_i(\boldsymbol{x}, t) \cdot \boldsymbol{n}_i(\boldsymbol{x}) = -D_j \nabla M_j(\boldsymbol{x}, t) \cdot \boldsymbol{n}_j(\boldsymbol{x}),\ \boldsymbol{x} \in \Gamma_{ij}, \tag{3.5}$$

where i, j belong to $\{1, ..., N\}$, and κ_{ij} characterizes the permeability of the interface Γ_{ij} with respect to water. Due to the flux continuity eq. (3.5), we have $\kappa_{ij} = \kappa_{ji}$. Compartments are coupled by the interface conditions.

The order of magnitude of some physical quantities is: the cell membrane permeability to water protons is around $5 \times 10^{-6}\ \mu m/\mu s$ [167–169]; the transverse relaxation time of water protons in brain tissue is in the order of $100\ ms$ [170]; the self-diffusion coefficient of water protons at $37\ ^\circ C$ is $3 \times 10^{-3}\ \mu m^2/\mu s$ [34].

The magnetic field gradient $\boldsymbol{G}$ is a 3D vector function whose intensity $g(t) = \|\boldsymbol{G}(t)\|$ and direction $\boldsymbol{u}_g(t) = \boldsymbol{G}(t)/g(t)$ can vary with time. The vector function $\boldsymbol{G}$ is often referred to as diffusion-encoding sequence, gradient sequence, or just sequence. It is indispensable in encoding the water proton diffusion that is restricted or hindered by the boundaries or interfaces into the magnetization M. The aggregate magnetization in the entire domain Ω at time TE yields a measurable quantity, a dMRI signal. MRI scanners can vary the intensity, direction, and time profile of the diffusion-encoding sequence to obtain a series of dMRI signals. One main objective of diffusion MRI is to infer the geometrical properties of the compartments delimited by their boundaries (cell membranes) using

a set of dMRI signals. To achieve this objective, the gradient sequence design is crucial.

Three common types of diffusion-encoding sequences include the pulsed-gradient spin echo (PGSE) sequences [18], the double pulsed-gradient spin echo (d-PGSE) sequences [171, 172], the oscillating gradient spin echo sequences of type sine (sin-OGSE) and cosine (cos-OGSE) [173, 174]. Their magnetic field gradients $\boldsymbol{G}$ are the product of the gradient intensity g, the gradient direction $\boldsymbol{u}_g(t)$ and a time profile $f(t)$. In other word, $\boldsymbol{G}(t) = g \cdot f(t)\boldsymbol{u}_g(t)$.

Figure 3.2 presents the time profiles of these sequences. The parameters that define these sequences are the following,

1. PGSE: the pulse duration δ, the inter-pulse duration Δ, the gradient intensity g, and the direction of the magnetic field gradient $\boldsymbol{u}_g$. The time profile of PGSE is
$$f_1(t) = \begin{cases} 1, & 0 \leq t \leq \delta, \\ -1, & \Delta \leq t \leq \Delta + \delta, \\ 0, & \text{otherwise;} \end{cases} \tag{3.6}$$

2. d-PGSE: the pulse durations δ_1 and δ_2, the inter-pulse durations Δ_1 and Δ_2, the mixing time t_m, the gradient intensities $g_1 = g$ and $g_2 = \alpha g$, and the directions of the magnetic field gradient $\boldsymbol{u}_{g_1}$ and $\boldsymbol{u}_{g_2}$. The time profile of d-PGSE is
$$f_2(t) = \begin{cases} 1, & 0 \leq t \leq \delta_1, \\ -1, & \Delta_1 \leq t \leq \Delta_1 + \delta_1, \\ -\alpha, & \Delta_1 + t_m \leq t \leq \Delta_1 + \delta_2 + t_m, \\ \alpha, & \Delta_1 + \Delta_2 + t_m \leq t \leq \Delta_1 + \Delta_2 + \delta_2 + t_m, \\ 0, & \text{otherwise;} \end{cases} \tag{3.7}$$

3. OGSE: the oscillating pulse duration δ, the period of an oscillation τ, the inter-pulse duration Δ, the gradient intensity g, and the direction of the magnetic field gradient $\boldsymbol{u}_g$. The time profile of sin-OGSE and cos-OGSE are
$$f_3(t) = \begin{cases} \sin\left(\frac{2\pi t}{\tau}\right), & 0 \leq t \leq \delta, \\ -\sin\left(\frac{2\pi(t-\Delta)}{\tau}\right), & \Delta \leq t \leq \Delta + \delta, \\ 0, & \text{otherwise;} \end{cases} \tag{3.8}$$
and
$$f_4(t) = \begin{cases} \cos\left(\frac{2\pi t}{\tau}\right), & 0 \leq t \leq \delta, \\ -\cos\left(\frac{2\pi(t-\Delta)}{\tau}\right), & \Delta \leq t \leq \Delta + \delta, \\ 0, & \text{otherwise.} \end{cases} \tag{3.9}$$

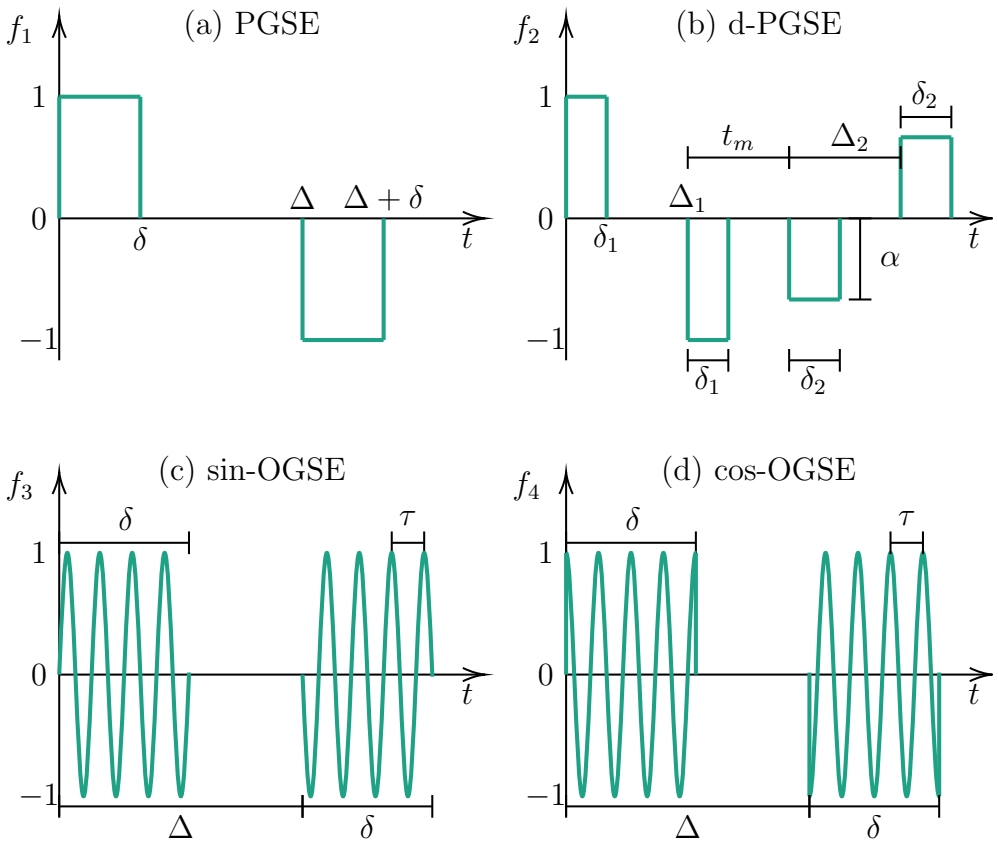


Figure 3.2: Effective time profiles of some commonly used diffusion-encoding sequences. The effect of refocusing $180°$ RF pulses is taken into account. **(a)** Pulsed-Gradient Spin Echo (PGSE) sequence. **(b)** double Pulsed-Gradient Spin Echo (d-PGSE) sequence. **(c)** Oscillating Gradient Spin Echo sequence of type sine (sin-OGSE). **(d)** Oscillating Gradient Spin Echo sequence of type cosine (cos-OGSE).

The most commonly used sequence type is PGSE, proposed by Stejskal and Tanner [18]. It has a wide range of applications in oncology [175], tractography [176], ischemic stroke identification [177], etc. The d-PGSE sequences are good at preserving the diffusion-diffraction patterns that are shown to be closely related to microstructural information [172, 178]. The OGSE sequences are believed to have short diffusion times, allowing us to measure finer structures. This thesis mainly focuses on dMRI using PGSE sequences.

Equations (3.1) to (3.5) govern the time evolution of magnetization in all compartments. Once we obtained the magnetization by solving the BT equation with a given gradient sequence, the dMRI signal s from the voxel Ω measured at TE

is the spatial integration of the magnetization:

$$s = \sum_{i=1}^{N} \int_{\boldsymbol{x} \in \Omega_i} M_i(\boldsymbol{x}, TE) d\boldsymbol{x}. \tag{3.10}$$

In practice, the dMRI signal is normalized by the signal in the absence of gradient sequence, i.e., $\boldsymbol{G} = \boldsymbol{0}$, to get the signal attenuation

$$E = \frac{s}{s_0}, \tag{3.11}$$

where s_0 denotes the signal without magnetic field gradient. The normalization mainly leaves the attenuation due to diffusion and eliminates other effects such as transverse relaxation [11, p. 6]. The dMRI signal s and the signal attenuation E generally depend on the entire magnetic field gradient $\boldsymbol{G}$. With one of the common sequences shown in fig. 3.2, s and E are functions of the corresponding sequence parameters listed above. For example, the signal attenuation of PGSE is a function of δ, Δ, g and $\boldsymbol{u}_g$. In the literature, the first three parameters are usually replaced by two quantities: diffusion time t_d and bvalue (or simply *b*).

Diffusion time is proposed to quantify the duration of water proton diffusion encoded by the magnetic field gradient. However, various diffusion time definitions exist in the literature for PGSE sequence, such as $\Delta - \delta/3$ [179], Δ [11, 180], and $\Delta + \delta$ [181]. These definitions converge to Δ in the limit of $\frac{\delta}{\Delta} \to 0$, but it is still not clear whether the notion of diffusion time is well-defined for long pulses (i.e., $\delta \sim \Delta$) [182]. We adopt the widely used definition $t_d = \Delta - \delta/3$ and only take it as a rough estimation of the diffusion time.

For PGSE sequence, the bvalue is defined as

$$b = \gamma^2 g^2 \int_0^{TE} \left(\int_0^t f_1(s) ds \right)^2 dt = \gamma^2 g^2 \delta^2 (\Delta - \delta/3). \tag{3.12}$$

The reason for this definition is that in free diffusion or the case when the Gaussian phase assumption (GPA) is applicable, the signal attenuation is or can be approximated by $e^{-D_e \cdot b}$ where D_e is an effective diffusion coefficient, also known as ADC. The ADC can be approximated using signal attenuation:

$$D_e(g, \boldsymbol{u}_g, \delta, \Delta) = \frac{-ln\left(E(g, \boldsymbol{u}_g, \delta, \Delta)\right)}{b}. \tag{3.13}$$

For isotropic free diffusion, the ADC is a constant that coincides with the self-diffusion coefficient. This phenomenon enables one to measure the self-diffusion coefficient of liquid molecules using diffusion MRI [34, 183]. For unrestricted diffusion in an anisotropic medium, such as a polymer solution, characterized by a diffusion tensor $\boldsymbol{D}$, the ADC depends only on the gradient direction

and $D_e(\boldsymbol{u}_g) = \boldsymbol{u}_g^T \boldsymbol{D} \boldsymbol{u}_g$. Measuring ADCs in at least six noncollinear directions allows for diffusion tensor imaging [74]. For restricted diffusion, one can measure the ADC when the Gaussian phase approximation is applicable so that the ADC is physically meaningful [68]. ADC is a useful clinical marker for distinguishing low and high diffusivity regions. For example, the decrease of the ADC may indicate ischemic stroke [184]. However, the apparent diffusion coefficient is often misused in the literature. We refer to Grebenkov's work [68] for clarification.

We stress that the Bloch-Torrey equation system (eqs. (3.1) to (3.5)) and the signal computation (eqs. (3.10) and (3.11)) do not depend on the definitions of diffusion time, bvalue, and ADC.

3.1.2 A simplified form of BT equation

The general form of BT equation provides a complete description of the attenuation mechanisms related to dMRI. Although numerically solving the general form is already feasible, the computation is time-consuming and resource-intensive. We refer to the papers of Nguyen et al. [185] and Agdestein et al. [50] for the approaches to solve the general form of BT equation. This section introduces two assumptions for simplifying the BT equation.

First, the boundaries and interfaces are assumed to be impermeable, i.e., $\kappa_i = 0$ and $\kappa_{ij} = 0$ for $i, j \in \{1, ..., N\}$, meaning that we ignore the effect of transcytolemmal water exchange. For human brain tissue, the cell membrane permeability is around $5 \times 10^{-6}\ \mu m/\mu s$, which corresponds to an intracellular water residence time of $\sim 600\ ms$ and an extracellular water residence time of $\sim 120\ ms$ [169, 186, 187]. The impermeability assumption is reasonable when the diffusion time is significantly less than the water residence time. Based on the extracellular water residence time, Yang et al. conservatively estimate the diffusion time should be of $\sim \mathbf{12\ ms}$ or less for the valid application of impermeability assumption to MR studies [187]. Palombo et al. extend the limit to $\mathbf{20\ ms}$ for diffusion MRI application [88]. Jelescu et al. estimate the inter-compartment water exchange time to be $\mathbf{15 - 60\ ms}$ in the rat cortex and hippocampus based on the NEXI model [188]. Our experiments on clinical dMRI data presented in chapter 5 suggest that the water exchange effect is minor when the diffusion time is $\sim 46\ ms$.

Second, we assume the transverse relaxation is homogeneous in all compartments within a brain voxel, i.e., $T_i = T$ for $i \in \{1, ..., N\}$ where T is the transverse relaxation time constant measured by, for example, Hahn echoes [17]. We stress that T can vary across brain voxels.

The first assumption decouples the magnetization in different compartments giving N independent BT equations. In addition, we introduce a new variable φ representing the magnetization unattenuated by relaxation and rescaled

by initial magnetization. The new variable φ is defined as

$$M_i(\boldsymbol{x}, t) = \rho e^{-\frac{t}{T}} \varphi_i(\boldsymbol{x}, t),\ \boldsymbol{x} \in \Omega_i,\ t \in [0, TE], \tag{3.14}$$

where φ_i is the restriction of φ to Ω_i. Equation (3.14) is similar to Torrey's treatment [33]. We refer to the new variable φ as nonrelaxed magnetization. The BT equation that governs φ_i is

$$\frac{\partial}{\partial t}\varphi_i(\boldsymbol{x}, t) = (D_i \nabla \cdot \nabla - \imath\gamma \boldsymbol{x} \cdot \boldsymbol{G}(t))\varphi_i(\boldsymbol{x}, t), \qquad \boldsymbol{x} \in \Omega_i,\ t \in [0, TE], \tag{3.15}$$

$$\varphi_i(\boldsymbol{x}, 0) = 1, \qquad \boldsymbol{x} \in \Omega_i, \tag{3.16}$$

$$D_i \nabla \varphi_i(\boldsymbol{x}, t) \cdot \boldsymbol{n}_i(\boldsymbol{x}) = 0, \qquad \boldsymbol{x} \in \partial\Omega_i,\ t \in [0, TE], \tag{3.17}$$

where i is an integer ranging from 1 to N.

Equation (3.15) is a simplified form of the BT equation that we obtained by making the two assumptions. The numerical methods that will be presented in section 3.2 and chapter 4 focus on solving the simplified form.

Let us reformulate the signal attenuation E with φ. We denote by $\varphi(\boldsymbol{x}, t; g = 0)$ the nonrelaxed magnetization in the absence of magnetic field gradient and V_i the volume of the compartment Ω_i. According to the divergence theorem, we obtain[2]

$$V_i = \int_{\Omega_i} \varphi_i(\boldsymbol{x}, TE; g = 0) d\boldsymbol{x}. \tag{3.18}$$

Dividing s by s_0 cancels the initial magnetization and the homogeneous transverse relaxation. Let E_i denote the signal attenuation of the i-th compartment, and we have

$$E_i = \frac{\int_{\Omega_i} \rho e^{-\frac{TE}{T}} \varphi_i(\boldsymbol{x}, TE) d\boldsymbol{x}}{\int_{\Omega_i} \rho e^{-\frac{TE}{T}} \varphi_i(\boldsymbol{x}, TE; g = 0) d\boldsymbol{x}} = \frac{\int_{\Omega_i} \varphi_i(\boldsymbol{x}, TE) d\boldsymbol{x}}{V_i}. \tag{3.19}$$

Finally, the signal attenuation of a voxel is

$$E = \frac{\sum \int_{\Omega_i} \rho e^{-\frac{TE}{T}} \varphi_i(\boldsymbol{x}, TE) d\boldsymbol{x}}{\sum \rho e^{-\frac{TE}{T}} V_i} = \frac{\sum V_i E_i}{\sum V_i} = \sum f_i^v E_i, \tag{3.20}$$

where $f_i^v = V_i / \sum V_j$ is the volume fraction of Ω_i.

Given the two assumptions, eq. (3.20) reveals that the voxel signal attenuation is the volume fraction weighted sum of the compartmental signal attenuations. Since compartments are decoupled, we can focus on solving eqs. (3.15) to (3.17) for a single compartment Ω_i. The following section presents two methods for solving the simplified BT equation.

[2] The derivation of eq. (3.18) is in section B.1.

3.2 Numerical methods with FE discretization

This section introduces two well-established numerical methods, i.e., finite element method (FEM) and numerical matrix formalism (numerical MF), for solving the BT equation based on finite element discretization. We will describe the main steps of each method and give the parameters that control the simulation precision. However, the goal is not to provide a comprehensive manual for the methods. We refer to several works related to the SpinDoctor toolbox [44, 45, 49, 50] for more details about these two methods.

3.2.1 Finite element method

The finite element method is a conventional method for numerically solving differential equations. It relies on discretization in space to subdivide a large system into smaller and simpler parts. Typically, one performs spatial discretization through segmentation (1D), triangulation (2D), or tetrahedralization (3D) of computational domains, as we did in chapter 2. The discretization is usually achieved by meshing tools such as Tetgen [111], CGAL [113], and Gmsh [112].

As explained in the previous section, we focus on solving eqs. (3.15) to (3.17) for a single compartment $\Omega_i,\ i \in \{1, ..., N\}$. Suppose the compartment Ω_i is discretized into a tetrahedral volume mesh consisting of K vertices $\{\boldsymbol{v}_k\}_{k\in\{1,...,K\}}$. We adopt a FE space that is spanned by a set of continuous piecewise linear functions $\{\eta_k\}_{k\in\{1,...,K\}}$ (called P1 elements in FEM literature). The mesh vertices are also called FE nodes.

To give an intuition about how P1 elements are defined, we first provide an example of one-dimensional P1 elements in fig. 3.3. Suppose a line is discretized into four segments delineated by five vertices $\{v_1, ..., v_5\}$. The P1 elements defined on the line are the set of piecewise linear functions $\{h_1, ..., h_5\}$ shown in fig. 3.3. Each P1 element corresponds to a vertex and is defined based on the position of the vertex. For example, the function h_3 has a value of 1 at the vertex v_3 and 0 at other vertices. Together with the fact that h_3 is a piecewise linear function, it is completely determined. Other functions are defined in the same way.

The P1 elements defined on the tetrahedral volume mesh of Ω_i are the extension of h's to three dimensions. For example, the value of η_k is one at $\boldsymbol{v}_k$ and 0 at other vertices. Since η_k is piecewise linear, it is non-zero on the tetrahedra that touch $\boldsymbol{v}_k$. The set of P1 elements forms a basis of the FE space defined on the mesh of Ω_i. We call it a FE basis. Three properties of the FE basis functions are the following:

1. the value of $\eta_k(\boldsymbol{v}_l)$ is 1 if $k = l$, 0 otherwise;

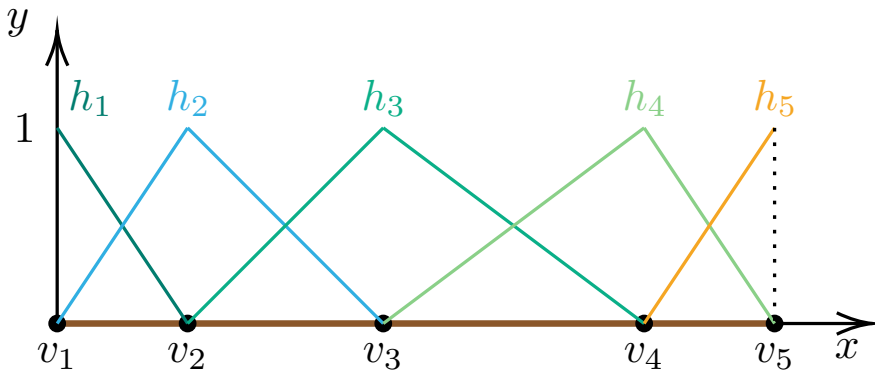


Figure 3.3: Example of one-dimensional P1 elements defined on a line discretized into four segments.

2. the value of $\eta_k(\boldsymbol{x})$ is non-zero if and only if $\boldsymbol{x}$ is inside a tetrahedron with $\boldsymbol{v}_k$ as a vertex;

3. the product $\eta_k\eta_l$ $(k, l \in \{1, ..., K\})$ is non-zero (not a zero-valued constant function) if and only if $\boldsymbol{v}_k$ and $\boldsymbol{v}_l$ are the vertices of the same tetrahedron.

FEM searches for a solution in the FE space. To achieve this, we need the weak form of the BT equation. Letting u be a test function, the weak form is

$$\frac{d}{dt}\int_{\Omega_i}\varphi_i(\boldsymbol{x},t)u(\boldsymbol{x})d\boldsymbol{x} = -D_i\int_{\Omega_i}\nabla\varphi_i(\boldsymbol{x},t)\nabla u(\boldsymbol{x})d\boldsymbol{x} - \imath\gamma\int_{\Omega_i}\boldsymbol{x}\cdot\boldsymbol{G}(t)\varphi_i(\boldsymbol{x},t)u(\boldsymbol{x})d\boldsymbol{x}. \tag{3.21}$$

An approximate solution of the equation system (eqs. (3.15) to (3.17)) is sought for in the form

$$\varphi_i(\boldsymbol{x},t) = \sum_{k=1}^{K}\xi_k(t)\eta_k(\boldsymbol{x}) = \boldsymbol{\xi}(t)^T\boldsymbol{H}(\boldsymbol{x}), \tag{3.22}$$

where $\boldsymbol{H}(\boldsymbol{x}) = [\eta_1(\boldsymbol{x}), ..., \eta_K(\boldsymbol{x})]^T$ is the vector of finite element basis functions, $\{\xi_k\}_{k\in\{1,...,K\}}$ are complex-valued time-dependent coefficients, and $\boldsymbol{\xi}(t) = [\xi_1(t), ..., \xi_K(t)]^T$. The k-th coefficient coincides with the value of φ_i at vertex $\boldsymbol{v}_k$, i.e., $\varphi_i(\boldsymbol{v}_k, t) = \xi_k(t)$. We note that the superscript notation ^{T} only denotes *transpose* instead of the complex conjugate transpose denoted by †.

We choose K test functions which are $u = \eta_k, k \in \{1, ..., K\}$. Substituting eq. (3.22) into eq. (3.21), we obtain an ODE in the time interval $t \in [0, TE]$ that governs the vector function $\boldsymbol{\xi}(t)$

$$\boldsymbol{M}\frac{d\boldsymbol{\xi}(t)}{dt} = -\left(D_i\boldsymbol{S} + \imath\gamma\boldsymbol{J}(t)\right)\boldsymbol{\xi}(t), \tag{3.23}$$

with $\boldsymbol{M}$, $\boldsymbol{S}$, and $\boldsymbol{J}(t)$ three real-valued matrices of size $K \times K$. The matrix elements are

$$[\boldsymbol{M}]_{mk} = \int_{\Omega_i} \eta_m(\boldsymbol{x})\eta_k(\boldsymbol{x})d\boldsymbol{x}, \tag{3.24}$$

$$[\boldsymbol{S}]_{mk} = \int_{\Omega_i} \nabla\eta_m(\boldsymbol{x}) \cdot \nabla\eta_k(\boldsymbol{x})d\boldsymbol{x}, \tag{3.25}$$

$$[\boldsymbol{J}(t)]_{mk} = \int_{\Omega_i} \boldsymbol{x} \cdot \boldsymbol{G}(t)\eta_m(\boldsymbol{x})\eta_k(\boldsymbol{x})d\boldsymbol{x}, \tag{3.26}$$

with $m, k \in \{1, ..., K\}$. The gradient directions of the three common types of sequences do not vary with time ($\boldsymbol{G}(t) = g \cdot f(t)\boldsymbol{u}_g$). In these cases, the matrix $\boldsymbol{J}(t)$ is a product of a time-independent matrix $\boldsymbol{J}'$, the gradient intensity g, and the time profile $f(t)$, i.e., $\boldsymbol{J}(t) = f(t)g\boldsymbol{J}'$. The elements of $\boldsymbol{J}'$ are

$$[\boldsymbol{J}']_{mk} = \boldsymbol{u}_g \cdot \int_{\Omega_i} \boldsymbol{x}\eta_m(\boldsymbol{x})\eta_k(\boldsymbol{x})d\boldsymbol{x},\ m, k \in \{1, ..., K\}. \tag{3.27}$$

As $\boldsymbol{J}'$ is time-independent, we do not need to compute it at every time step when solving eq. (3.23).

The initial condition related to eq. (3.23) is

$$\boldsymbol{\xi}(0) = \mathbf{1} = [1, ..., 1]^T, \tag{3.28}$$

which is a consequence of the fact $\varphi_i(\boldsymbol{v}_k, 0) = 1 = \xi_k(0)$ for $k \in \{1, ..., K\}$.

Given the tetrahedral volume mesh of Ω_i and the P1 elements, the matrix elements can be computed numerically. In the SpinDoctor toolbox, the matrix construction is performed with a vectorized routine proposed by Rahman and Valdman [189].

The matrices $\boldsymbol{S}$ and $\boldsymbol{M}$ are called stiffness and mass matrices, respectively, in the FEM literature. The matrix $\boldsymbol{M}$ is positive-definite and $\boldsymbol{S}$ is positive semi-definite [190]. Due to the three properties of the P1 elements mentioned above, all three matrices are sparse and symmetric.

The particularity of eq. (3.23) is that it has a mass matrix on the left-hand side, making it a linearly implicit ODE. SpinDoctor calls MATLAB built-in ODE solvers with adaptive time-stepping to solve the ODE eq. (3.23) and obtain $\boldsymbol{\xi}(t)$ for $t \in [0, TE]$. The default ODE solver is *ode15s* [191]. Putting $\boldsymbol{\xi}(t)$ into eq. (3.22), we get the time evolution of φ. The signal attenuation from the compartment Ω_i can be computed with $\varphi(\boldsymbol{x}, TE)$ (see eq. (3.19)).

Finally, we list the simulation parameters of SpinDoctor's FEM implementation, which control the precision of FEM simulation. They are the following:

1. A discretization parameter H controls the maximum volume of the tetrahedra. The software Tetgen [111] is called in SpinDoctor to tetrahedralize surface meshes or refine volume meshes. If one sets H to $h\ \mu m^3$, the maximum volume of tetrahedra will be $h\ \mu m^3$. If one sets H to -1, the default discretization routine[3] of Tetgen is triggered. Tetgen uses an adaptive method to discretize the volume and add new points to improve the mesh quality [111]. Figure 3.5 shows a comparison of meshes with different spatial discretization;

2. A parameter $rtol$ controls the relative residual tolerance of the solution at all points of the mesh at each time step. Roughly speaking, it controls the number of correct digits in all elements of $\boldsymbol{\xi}(t)$;

3. A parameter $atol$ controls the absolute residual tolerance at all points of the FE mesh at each time step. This tolerance is a threshold below which the value of the solution becomes unimportant.

3.2.2 Numerical matrix formalism

Matrix formalism (MF) [47, 48] is a spectral method for solving BT equations with a computational domain Ω. It represents the time evolution of magnetization in an explicit matrix form by projecting BT equations to the Laplacian eigenbasis of the given domain. The method's main advantage is that once the Laplacian eigenbasis for a given computational domain is obtained, further computations with various dMRI settings are much faster [47]. The calculation of the Laplacian eigenbasis, also known as eigendecomposition, for a given computational domain only needs to be performed once. However, the explicit Laplacian eigenbases are only known for some simple geometries, e.g., rectangles, cubes, disks, and spheres. It is nontrivial to find the Laplacian eigenbasis for an arbitrary computational domain.

Li et al. [49, 50] proposed a method to numerically compute Laplacian eigenbases based on finite element discretization of computational domains. This method not only retains the advantage mentioned above but also brings the generality of FEM. We refer to this method as numerical matrix formalism (numerical MF).

We will introduce the numerical MF in two steps: the first gives the representation of the BT equation with Laplacian eigenbases, and the second explains the numerical computation of the eigenbases based on finite element discretization.

[3] `https://wias-berlin.de/software/tetgen/1.5/doc/manual/manual005.html#cmd-q`

Matrix formalism representation

Consider the eigenvalue problem for the Laplace operator $\nabla \cdot \nabla$ in the compartment Ω_i ($i \in \{1, ..., N\}$) with zero Neumann boundary condition:

$$-\nabla \cdot \nabla \psi(\boldsymbol{x}) = \lambda \psi(\boldsymbol{x}), \qquad \boldsymbol{x} \in \Omega_i, \tag{3.29}$$

$$\nabla \psi(\boldsymbol{x}) \cdot \boldsymbol{n}_i(\boldsymbol{x}) = 0, \qquad \boldsymbol{x} \in \partial\Omega_i, \tag{3.30}$$

where λ is an eigenvalue of the Laplace operator and ψ the corresponding eigenfunctions. Soma basic facts of the eigenstates are [192, 193]:

1. The spectrum is discrete.
2. All eigenvalues are real and nonnegative.
3. All eigenfunctions can be chosen to be real-valued and form a complete orthonormal basis in the function space $L_2(\Omega_i)$.

Let $\{(\lambda_j, \psi_j)\}_{j \in \mathbb{N}^*}$ be a set of solutions of eqs. (3.29) and (3.30). We sort the eigenvalues in non-decreasing order:

$$0 = \lambda_1 < \lambda_2 \leq \lambda_3 \leq \dots$$

Since Ω_i is a connected domain subject to the zero Neumann boundary condition, λ_1 is zero and ψ_1 is a constant function [193]. In addition, we choose the eigenfunctions to be real-valued and form a complete orthonormal basis. For numerical purposes, we only keep the J smallest eigenvalues.

We decompose the nonrelaxed magnetization φ_i to the truncated Laplacian eigenbasis $\{\psi_j\}_{j \in \{1,...,J\}}$ to get

$$\varphi_i(\boldsymbol{x}, t) \simeq \sum_{j=1}^{J} c_j(t) \psi_j(\boldsymbol{x}) = \boldsymbol{C}(t)^T \boldsymbol{\Psi}(\boldsymbol{x}), \ \boldsymbol{x} \in \Omega_i, \ t \in [0, TE], \tag{3.31}$$

where $\{c_j\}_{j \in \{1,...,J\}}$ are complex-valued time-dependent coefficients, $\boldsymbol{C}(t) = [c_1(t), ..., c_J(t)]^T$ is the projection of the magnetization to the Laplacian eigenbasis $\boldsymbol{\Psi}(\boldsymbol{x}) = [\psi_1(\boldsymbol{x}), ..., \psi_J(\boldsymbol{x})]^T$.

The magnetization projection satisfies an ordinary differential equation [49, 50]

$$\frac{d}{dt}\boldsymbol{C}(t) = -\left(D_i \boldsymbol{\Lambda} + \imath\gamma \boldsymbol{A}(t)\right) \boldsymbol{C}(t), \ t \in [0, TE]. \tag{3.32}$$

The matrix $\boldsymbol{\Lambda}$ comprises the eigenvalues in its diagonal, i.e., $\boldsymbol{\Lambda} = diag(\lambda_1, ..., \lambda_J)$. $\boldsymbol{A}(t)$ is a $J \times J$ matrix that depends on the magnetic field gradient $\boldsymbol{G}(t)$. The elements in $\boldsymbol{A}(t)$ are defined as

$$[\boldsymbol{A}(t)]_{mk} = \int_{\Omega_i} \boldsymbol{x} \cdot \boldsymbol{G}(t) \psi_m(\boldsymbol{x}) \psi_k(\boldsymbol{x}) d\boldsymbol{x}, \ (m, k) \in \{1, ..., J\}^2. \tag{3.33}$$

For the three common types of sequences, the matrix $\boldsymbol{A}(t)$ is a product of a time-independent matrix $\boldsymbol{A}'$, the gradient intensity g, and the time profile $f(t)$, i.e., $\boldsymbol{A}(t) = f(t)g\boldsymbol{A}'$. The elements of $\boldsymbol{A}'$ are

$$[\boldsymbol{A}']_{mk} = \boldsymbol{u}_g \cdot \int_{\Omega_i} \boldsymbol{x}\psi_m(\boldsymbol{x})\psi_k(\boldsymbol{x})d\boldsymbol{x},\ m,k \in \{1,...,J\}. \tag{3.34}$$

The initial condition related to eq. (3.32) is

$$\boldsymbol{C}(0) = \int_{\Omega_i} \boldsymbol{\Psi}(\boldsymbol{x})d\boldsymbol{x} = [\sqrt{V_i}, 0, ..., 0]^T. \tag{3.35}$$

In general, $\boldsymbol{\Lambda}$ and $\boldsymbol{A}(t)$ do not commute, so one cannot integrate eq. (3.32) unless $\boldsymbol{G}$ is a piecewise constant function. For PGSE sequences which are piecewise constant functions, solving eq. (3.32) does not require discretization in time. The magnetization projection at time TE is

$$\boldsymbol{C}(TE) = e^{-(D_i\boldsymbol{\Lambda} - \imath\gamma g\boldsymbol{A}')\delta} e^{-D_i\boldsymbol{\Lambda}(\Delta-\delta)} e^{-(D_i\boldsymbol{\Lambda} + \imath\gamma g\boldsymbol{A}')\delta} \boldsymbol{C}(0). \tag{3.36}$$

From right to left, we explain the meaning of each term of eq. (3.36):

- $\boldsymbol{C}(0)$ is the initial value of the magnetization projection;
- $e^{-(D_i\boldsymbol{\Lambda} + \imath\gamma g\boldsymbol{A}')\delta}$ describes the joint effect of diffusion $(D_i\boldsymbol{\Lambda})$ and gradient dephasing $(\imath\gamma g\boldsymbol{A}')$ of the first pulse whose duration is δ;
- $e^{-D_i\boldsymbol{\Lambda}(\Delta-\delta)}$ represents the diffusion effect on the magnetization projection during the time interval $[\delta, \Delta]$;
- $e^{-(D_i\boldsymbol{\Lambda} - \imath\gamma g\boldsymbol{A}')\delta}$ is the joint effect of diffusion and gradient dephasing of the second pulse;
- $\boldsymbol{C}(TE)$ is the final value of the magnetization projection at the echo time TE.

Equation (3.36) reflects the theoretical advantages of matrix formalism. First, each attenuation mechanism is represented by a matrix in the explicit expression. Second, the physical parameters, such as diffusivity and gradient intensity, which characterize the strength of the underlying attenuation mechanism, serve as scaling coefficients in front of the corresponding matrices [47]. Third, the joint effect of different mechanisms is formulated as the matrix addition. Fourth, the effect of attenuation mechanisms on the magnetization projection is properly translated into the action of a matrix exponential to $\boldsymbol{C}$.

Similarly, the value of $\boldsymbol{C}$ at time TE with d-PGSE sequences is a time-ordered exponential:

$$\begin{aligned}\boldsymbol{C}(TE) = & e^{-(D_i\boldsymbol{\Lambda}+\imath\gamma g_2\boldsymbol{A}')\delta_2}e^{-D_i\boldsymbol{\Lambda}(\Delta_2-\delta_2)}e^{-(D_i\boldsymbol{\Lambda}-\imath\gamma g_2\boldsymbol{A}')\delta_2}e^{-D_i\boldsymbol{\Lambda}(t_m-\delta_1)} \\ & e^{-(D_i\boldsymbol{\Lambda}-\imath\gamma g_1\boldsymbol{A}')\delta}e^{-D_i\boldsymbol{\Lambda}(\Delta_1-\delta_1)}e^{-(D_i\boldsymbol{\Lambda}+\imath\gamma g_1\boldsymbol{A}')\delta_1}\boldsymbol{C}(0).\end{aligned} \tag{3.37}$$

To solve eq. (3.32) with arbitrary gradient sequences, we can approximate $\boldsymbol{G}(t)$ with piecewise constant functions [47]. We can also compute $\boldsymbol{C}(TE)$ using conventional ODE solvers. Putting $\boldsymbol{C}(TE)$ into eq. (3.31), we find $\varphi_i(\boldsymbol{x}, TE)$, thus the solution of eq. (3.15) as well as the signal attenuation E_i (eq. (3.19)).

Characteristic length and time scales

Laplacian eigenstates can be associated with characteristic length and time scales. On a line segment of length R (represented by an interval $[0, R]$), the j-th eigenvalue λ_j is $\pi^2(j-1)^2/R^2$, and the corresponding eigenfunction is $cos(\sqrt{\lambda_j}x),\ x \in [0, R]$. The half wavelength of the j-th ($j > 1$) eigenfunction is $\pi/\sqrt{\lambda_j}$. In analogy with the 1D case, we define the characteristic length scale ℓ for a Laplacian eigenvalue λ as [49, 50]

$$\ell(\lambda) = \begin{cases} +\infty, & \lambda = 0, \\ \frac{\pi}{\sqrt{\lambda}}, & \lambda > 0. \end{cases} \tag{3.38}$$

The time scale $\tau(\lambda)$ sets a "lifetime" for the eigenstate associated with λ [48],

$$\tau(\lambda) = \begin{cases} +\infty, & \lambda = 0, \\ \frac{1}{D_i\lambda}, & \lambda > 0. \end{cases} \tag{3.39}$$

Indeed, a derivation of eq. (3.36) is that an eigenstate's contribution to the signal is proportional to $e^{-(\Delta-\delta)/\tau} \sim e^{-t_d/\tau}$. If the diffusion time t_d is much greater than $\tau(\lambda)$, the corresponding eigenstate vanishes. Comparing τ with the diffusion time t_d helps decide the number of eigenvalues to keep. Finally, ℓ and τ are related by

$$\ell^2 = \pi^2 D_i \tau. \tag{3.40}$$

Eigenfunctions on FE bases

The above matrix formalism of the magnetization depends on the known Laplacian eigenstates. For complex geometries, it is nearly impossible to find their analytical eigenfunctions. We now present the way, proposed by Li et al. [49, 50], to compute the eigenstates for complex geometries numerically. The idea is to discretize the complex geometries and calculate the eigenstates on FE bases.

Suppose the compartment Ω_i is discretized into a tetrahedral volume mesh consisting of K finite element nodes. Same as section 3.2.1, we adopt a FE space that is spanned by a set of continuous piecewise linear functions $\{\eta_k\}_{k\in\{1,...,K\}}$. We recall that the set of P1 elements is known for a given tetrahedral volume mesh.

To solve the eigenvalue problem (eqs. (3.29) and (3.30)) in the FE space, we project the eigenfunction ψ to the FE basis

$$\psi(\boldsymbol{x}) = \sum_{k=1}^{K} p_k \eta_k(\boldsymbol{x}) = \boldsymbol{p}^T \boldsymbol{H}(\boldsymbol{x}), \tag{3.41}$$

with $\boldsymbol{p} = [p_1, ..., p_K]^T$ and $\boldsymbol{H}(\boldsymbol{x}) = [\eta_1(\boldsymbol{x}), ..., \eta_K(\boldsymbol{x})]^T$.

The projection allows for the conversion from a continuous Laplace operator eigenvalue problem to a generalized matrix eigenvalue problem

$$\boldsymbol{S}\boldsymbol{p} = \lambda \boldsymbol{M}\boldsymbol{p} \tag{3.42}$$

where $\boldsymbol{S}$ is the stiff matrix (eq. (3.25)) and $\boldsymbol{M}$ is the mass matrix (eq. (3.24)).

The generalized matrix eigenvalue problem eq. (3.42) yields K pairs of eigenvalue and eigenvector $\{(\lambda_k, \boldsymbol{p}_k)\}_{k\in\{1,...,K\}}$. Putting the eigenvectors into eq. (3.41), we find the Laplacian eigenfunctions in the FE space.

We recall that $\boldsymbol{M}$ is positive-definite and $\boldsymbol{S}$ is positive semi-definite [190]. These matrix properties ensure that all eigenvalues are real and nonnegative, and all eigenvectors can be chosen to be real-valued and $\boldsymbol{M}$-orthonormal (i.e., $\boldsymbol{p}_m^T \boldsymbol{M} \boldsymbol{p}_k = 0$ if $m \neq k$, 1 if $m = k$, for $m, k \in \{1, ..., K\}$) [194]. The properties of the eigenvalues and eigenvectors guarantee that items 2 and 3 in section 3.2.2 hold.

It is worth stressing that, with the zero Neumann boundary conditions (eq. (3.30)), the Laplacian eigenstates depend only on geometrical configuration. We can understand this from the definition of eigenstates. Equations (3.29) and (3.30) do not involve diffsion coefficient and magnetic field gradient. We can also see this from eq. (3.42) in which the mass and stiff matrices rely only on the P1 elements defined in a volume mesh.

Therefore, for a tetrahedral volume mesh, the eigendecomposition only needs to be performed once. When the eigenstates are known, one can freely change the diffusion coefficient and the magnetic field gradient (sequence, direction, and intensity) to compute dMRI signals with much less computational expense than other methods. This is the main computational advantage of numerical MF.

The simulation parameters that control the precision of numerical MF are the following:

1. A discretization parameter H controls the maximum volume of the tetrahedra. As with the finite element method, the software Tetgen [111] is called to tetrahedralize surface meshes or refine volume meshes. If one sets H to $h\ \mu m^3$, the maximum volume of tetrahedra will be $h\ \mu m^3$. If one sets H to -1, the default discretization routine[4] of Tetgen is triggered. Tetgen uses an adaptive method to discretize the volume and add new points to improve the mesh quality [111].

2. A parameter controls the truncation of the Laplacian spectrum. The parameter can be the number of retained eigenstates n_{eigs}, the minimum length scale ℓ_{min}, or the minimum time scale τ_{min}.

3. (Not for piecewise constant magnetic field gradient) A time step dt or two tolerances $rtol$ and $atol$ that control the precision for solving the ODE eq. (3.32) numerically.

Solving generalized matrix eigenvalue problems eq. (3.42) is difficult and time-consuming [47]. The matrix exponential and multiplication required in eqs. (3.36) and (3.37) can also become expensive if the number of simulations is very large. To perform simulations with neurons in NeuronSet, the previous implementation of numerical MF in SpinDoctor is not efficient enough. The next section focuses on optimizing the numerical matrix formalism by adopting an appropriate eigendecomposition algorithm and GPU computation. The optimization leads to a tenfold improvement in computational efficiency, which is the second contribution of this thesis.

3.3 Optimizing numerical matrix formalism

Simulating a dMRI signal on a realistic neuron mesh can take several seconds or even minutes [116]. We aim to simulate hundreds of dMRI signals from each neuron mesh in NeuronSet. The total computational demand is a substantial challenge for most existing approaches. To meet such a huge demand, we leverage the numerical matrix formalism implemented in the SpinDoctor toolbox [44, 49, 50]. Nonetheless, the previous implementation is still not fast enough. The numerical matrix formalism has two time-consuming operations: matrix eigendecomposition required in eq. (3.42) and matrix multiplication required in eqs. (3.36) and (3.37). We aim to speed them up.

3.3.1 Method optimization

[4] https://wias-berlin.de/software/tetgen/1.5/doc/manual/manual005.html#cmd-q

A shift-and-invert transformation helps accelerate the eigendecomposition [195]. After the transformation, we obtain a standard eigenvalue problem

$$(\boldsymbol{S} - \omega \boldsymbol{M})^{-1} \boldsymbol{M} \boldsymbol{p} = \nu \boldsymbol{p} \text{ where } \nu = \frac{1}{\lambda - \omega}, \tag{3.43}$$

which is equivalent to eq. (3.42). We then employ the krylov-Shur algorithm [196] to solve eq. (3.43) instead of eq. (3.42). This method allows us to obtain only a few eigenvalues close to ω without computing the whole spectrum. We need eigenvalues close to zero, so we set $\omega = -10^{-8}$. Letting ω be slightly smaller than the target value is a trick to avoid the singularity of ν [195].

Regarding the matrix exponential and matrix-vector product required in the computation of the magnetization projection (eqs. (3.36) and (3.37)), the previous implementation of numerical MF in SpinDoctor uses Matlab built-in functions, *expm* [197, 198] and *mtimes*, that execute on CPUs. These basic algebraic operations can be accelerated using GPU computation.

Additionally, the computation of $\boldsymbol{C}(t)$ mainly involves the product of a matrix exponential with a vector, i.e., $\boldsymbol{y} = e^A \cdot \boldsymbol{x}$. For this particular operation, Al-Mohy et al. [199] propose an efficient method to compute the resulting vector $\boldsymbol{y}$ without explicitly evaluating the matrix exponential e^A. It relies on a scaling and squaring method [200] and truncated Taylor series to approximate the final result. A Matlab implementation of this method is called *expmv*[5]. We further speed up *expmv* by making it compatible with GPU matrices. Since the method avoids the explicit computation of matrix exponential, it is also memory-efficient. Saving memory is advantageous when computing large matrices, as memory is usually scarce on GPUs. The computation of $\boldsymbol{C}(t)$ has been accelerated by the GPU-version of *expmv*.

3.3.2 Performance improvement

We made the above optimization to the numerical MF implemented in the SpinDoctor toolbox. To compare the computational efficiency before and after the optimization, we conduct the eigendecomposition for three spheres to find the smallest 1000 eigenvalues using the *eigs* function [195, 196] implemented in Matlab R2021b. The computation is performed on a computer with 20 physical cores (2 Intel(R) Xeon(R) CPU E5-2660 v2 @ 2.20GHz), 256GB RAM, running CentOS Stream release 8.

Table 3.1 lists the computation times before and after the optimization. The optimization achieved a twenty-fold speedup. Moreover, the speedup does not degrade the computational accuracy. We get the same eigenvalues and eigenvectors. A typical neuron mesh has around 80,000 FE nodes and needs about

[5] `https://github.com/higham/expmv`

3,000 eigenvalues. After the optimization, the eigendecomposition for each neuron takes less than 10 minutes.

Table 3.1: Computation times of eigendecomposition for three spheres. The optimization achieved a twenty-fold speedup.

sphere radius	$5\ \mu m$	$10\ \mu m$	$20\ \mu m$
number of FE nodes	5222	18981	80191
before optimization	51.48 s	665.49 s	12944.36 s
after optimization	10.93 s	47.86 s	431.97 s
speedup	4.71	13.90	29.97

Another time-consuming step is calculating the magnetization projection $\boldsymbol{C}(t)$. The previous implementation utilizes Matlab's built-in function *expm* running on CPUs ("CPU+expm"). We replace *expm* with the function *expmv* [199] and modify *expmv* to make it compatible with GPU matrices ("GPU+expmv"). We compare the computational efficiency of the two implementations by conducting simulations on ten neuron meshes. The numbers of FE nodes of the meshes range from 30,000 to 150,000. For each cell, we compute the magnetization projection using PGSE sequences with 2 diffusion times, 50 gradient intensities ranging from 0.5 to 1000 mT/m, and 10 gradient directions.

We utilize two computing platforms for computational efficiency comparison

1. Platform 1: a computer with 64 physical cores (AMD EPYC 7742 64-Core Processor), 512GB RAM, running CentOS Stream release 8;

2. Platform 2: a computer with 20 physical cores (Intel(R) Xeon(R) Silver 4214R CPU @ 2.40GHz), 384GB RAM, one Nvidia A40 48G graphics card, using CUDA 11.7, running CentOS Stream release 8.

We run the previous implementation "CPU+expm" on Platform 1. Simulations with "GPU+expmv" are performed on Platform 2. Figure 3.4 shows the overall computation time for evaluating eq. (3.36) for each neuron mesh as a function of its FE node number. The combination of the *expmv* and the GPU computation brings a ten-fold speedup. Additionally, the new implementation does not reduce the computational accuracy either.

Next, we compare the optimized numerical MF with other simulation methods.

3.3.3 Simulation efficiency comparison

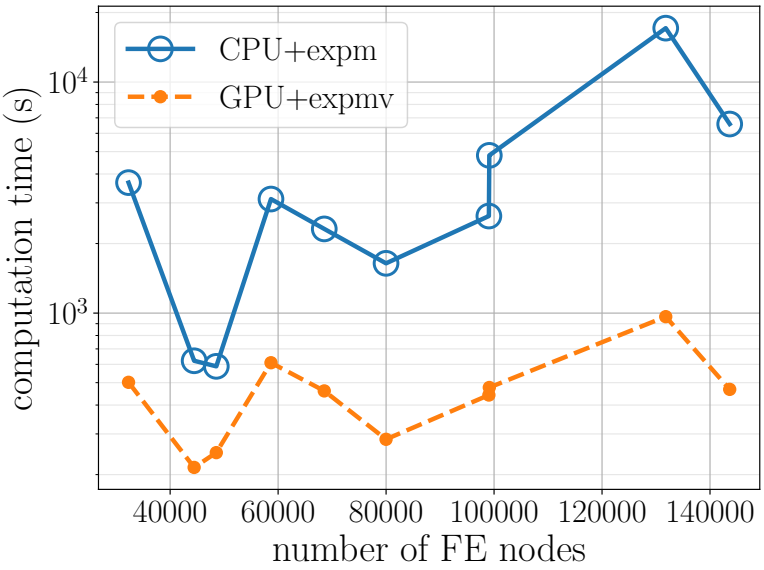


Figure 3.4: Overall computation time for evaluating eq. (3.36) with two diffusion times ($8/19$ and $8/49$ ms), fifty gradient intensities ranging from 0.5 to 1000 mT/m, and ten gradient directions. The x-axis represents the number of FE nodes. The y-axis in the logarithmic scale shows the computation times. The blue line with circular markers corresponds to the previous implementation that runs Matlab's built-in function *expm* on CPUs. The orange dashed line with dot markers corresponds to the new implementation with the GPU-version of *expmv* function. The combination of the *expmv* and the GPU computation brings a ten-fold speedup.

This chapter presented two types of dMRI simulations to solve the BT equation. One can also compute dMRI signals using Monte-Carlo methods to simulate the Brownian motion of random walkers. Since each random walker is independent of the others, Monte-Carlo methods are particularly suitable for parallel computing. On the contrary, it is hard to speed up FEM using GPUs due to the serial nature of FEM. The implementation of FEM in SpinDoctor utilizes multiprocessing to achieve parallel computation. FEM, MF, and Monte-Carlo methods are the three main approaches to dMRI simulations.

The debate about the computational efficiency of these three methods has always existed in the community. FEM suffers from low efficiency when the mesh size becomes huge. Matrix formalism is believed to be fast but was limited to simple shapes due to the difficulty of computing the Laplacian eigenbasis for arbitrary geometries. In addition, Monte-Carlo methods are constrained by their slow convergence rate ($1/\sqrt{N_{\text{walkers}}}$ with N_{walkers} the number of random walkers) [30]. The optimized numerical MF in this thesis is promising to end the debate.

This section compares the execution times of the three methods by conduct-

ing dMRI simulations on a neuron mesh. The ID of the neuron is NMO_85632. It is in the archive *Semendeferi_Muotri* of NeuronSet. We compute the dMRI signals using PGSE sequences with $\delta/\Delta = 8/49\ ms$ and 100 gradient intensities linearly spaced between 3 and 300 mT/m. The gradient direction is fixed to be the x-direction, i.e., $[1,0,0]^T$. The diffusivity is $3 \times 10^{-3}\ \mu m^2/\mu s$.

We choose the following numerical implementation for each method:

1. FEM: the SpinDoctor implementation running on Platform 1 with multiprocessing computation being activated;
2. Numerical MF: the optimized GPU version implemented in SpinDoctor running on Platform 2;
3. Monte-Carlo method: *disimpy* [65], a GPU-based Monte-Carlo simulator, running on Platform 2.

The simulation parameters of the Monte-Carlo method that control the simulation precision are the number of random walkers N_{walkers} and the number of time steps N_{t}.

To quantify the simulation accuracy, we compute reference signals using FEM by refining the discretization in space and time. The simulation parameters for the reference signals are $H = 0.5\ \mu m^3,\ rtol = 10^{-6},\ atol = 10^{-8}$. A comparison between a coarse and a refined mesh is shown in fig. 3.5.

We denote the reference signals by s_{ref} and the simulated signals as s. The relative error ε is defined as

$$\varepsilon = \left| \frac{s - s_{ref}}{s_{ref}} \right| \times 100\%. \tag{3.44}$$

For a fair comparison, we choose the simulation parameters so that the maximum relative errors of the three methods are around 2%. The simulation parameters for FEM are $H = -1,\ rtol = 10^{-4},\ atol = 10^{-6}$, for numerical MF are $H = -1,\ \ell_{min} = 1.5\,\mu m$, and for the Monte-Carlo method are $N_{\text{walkers}} = 10^5,\ N_{\text{t}} = 5 \times 10^4$. Figure 3.6 shows the relative errors of the three methods with the above settings. It can be seen that all three methods have comparable accuracy.

With similar precision, we can measure the computation times. The execution of the three methods can be divided into two steps: preparation and signal computation. The preparation step of FEM involves the computation of FE matrices (e.g., eqs. (3.24) to (3.26)). We treat the eigendecomposition (eq. (3.42)) as part of the preparation step for numerical MF. The preparation step of numerical MF also includes the computation of FE matrices. For the Monte-Carlo method, the preparation step includes calculating the initial positions of random walkers.

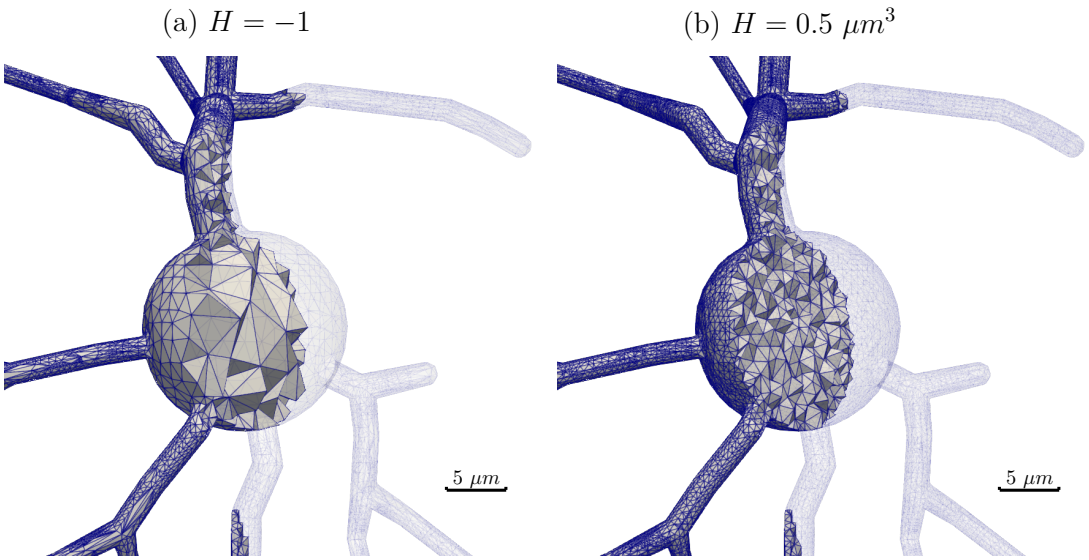


Figure 3.5: Comparison between a coarse and a refined mesh. The two meshes correspond to the selected neuron in this section. The ID of the neuron is NMO_85632. It is in the archive *Semendeferi_Muotri* of NeuronSet. Half of the meshes are made transparent to show the interior tetrahedra. **(a)** a coarse volume mesh generated by Tetgen with $H = -1$. It has 79992 vertices. **(b)** a refined volume mesh generated with $H = 0.5\ \mu m^3$. It has 163905 vertices.

The signal computation of FEM is the step in solving the ODE eq. (3.23). For the numerical MF, this step involves evaluating eq. (3.36). The Monte-Carlo method computes dMRI signals by simulating the Brownian motion of random walkers whose initial positions are set in the preparation step. We refer to the execution times of signal computation as computation times.

We list the execution times of the three methods in table 3.2. The numerical MF is much more efficient than the other two methods. Thanks to the optimization made in section 3.3, the eigendecomposition required in the preparation step of numerical MF takes 219 seconds, which is seven times faster than the Monte-Carlo method. We reiterate that the eigendecomposition only needs to be done once for a given geometrical configuration. Most importantly, the optimized numerical MF is highly advantageous in computing signals. For computing 100 dMRI signals, it is **20** times faster than FEM and **65** times faster than the Monte-Carlo method.

3.4 Numerical experiments

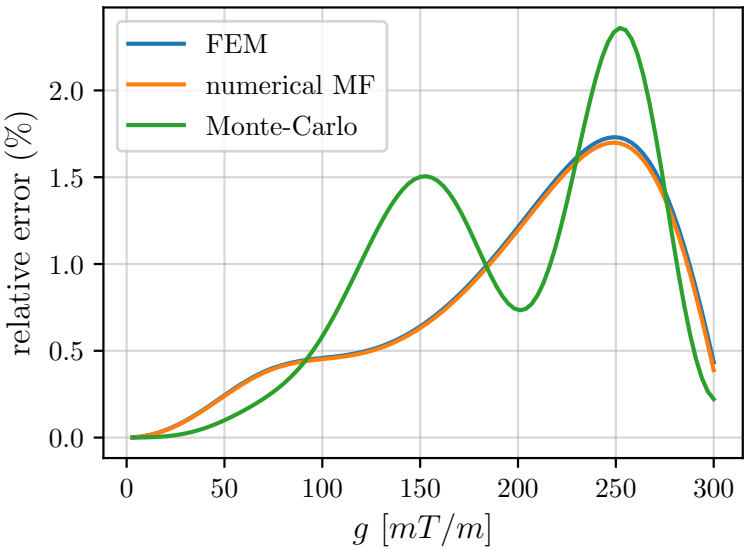


Figure 3.6: The relative errors (in percent) of three simulation methods (FEM, numerical MF, and a Monte-Carlo method) with respect to the gradient intensity. The maximum relative errors are about 2%. We perform the simulations on the neuron whose ID is NMO_85632 using PGSE sequences with $\delta/\Delta = 8/49\ ms$ and 100 gradient intensities linearly spaced between 3 and 300 mT/m. The gradient direction is fixed to be $[1, 0, 0]^T$. The diffusivity is $3 \times 10^{-3}\ \mu m^2/\mu s$. The relative errors are defined as $|s - s_{ref}|/|s_{ref}| \times 100\%$. We compute the reference signals using FEM by refining the discretization in space and time. Then we perform simulations using the three methods on the coarse mesh shown in fig. 3.5(a). The simulation parameters for FEM are $H = -1,\ rtol = 10^{-4},\ atol = 10^{-6}$, for numerical MF are $H = -1,\ \ell_{min} = 1.5\ \mu m$, and for the Monte-Carlo method are $N_{\text{walkers}} = 10^5,\ N_t = 5 \times 10^4$. All three methods converge to the reference solution.

Table 3.2: Execution times of three common dMRI simulation methods. The FEM is running on the CPU platform (Platform 1). The numerical MF and Monte-Carlo method is running on the GPU platform (Platform 2).

Methods	Max. relative error (%)	Preparation times (s)	Computation times (s)
FEM	1.73	0.7	281.9
Numerical MF	1.69	219.9	**13.5**
Monte-Carlo method	2.36	1474.4	902.6

In the previous sections, we introduced the BT equation and presented two ways to solve the BT equation numerically. In particular, we optimized the numerical MF implemented in the SpinDoctor, making it the fastest dMRI simulator. Together with polygon meshes, the numerical simulations allow us to study

dMRI signals on various geometrical configurations with different gradient sequences. We leverage realistic neuron meshes and dMRI simulators to study the water exchange inside a neuron and the power-law scaling. Finally, a statistical analysis shows that we can relate the soma sizes with six markers that can be computed from dMRI signals.

3.4.1 Diffusion directions distributed in two dimensions

We generated 90 directions uniformly distributed on the unit semi-circle lying in the $x-y$ plane (plotting 180 directions on the unit circle because simulated signals are antipodally symmetric) and computed the diffusion MRI signals using FEM in these 180 directions for three sequences:

- PGSE ($\delta/\Delta = 2.5/5\ ms$);
- PGSE ($\delta/\Delta = 10/43\ ms$);
- PGSE ($\delta/\Delta = 10/433\ ms$).

The simulation parameters are $H = 0.5\ \mu m^3,\ rtol = 10^{-3},\ atol = 10^{-5}$. With this choice, we verified that the signal is within 1% of the reference solution for all geometries (the whole neuron, the soma, and the two dendrites branches) for the three gradient sequences simulated.

The results for the spindle neuron *03b_spindle4aACC* in the Neuron Module are shown in fig. 3.7. We plot the signal attenuation in the 180 directions in the $x-y$ plane. The simulated geometries are superimposed on the plots for visualization.

It can be seen that the dendrite branch diffusion signal shape is more like an ellipse at $b = 1000\ \mu s/\mu m^2$, whereas at $b = 4000\ \mu s/\mu m^2$ the shape is non-convex. The signal shape of the soma is like an ellipse except for $b = 4000\ \mu s/\mu m^2$ at the two shorter diffusion times. At the two shorter diffusion times, the soma signal magnitude at $b = 4000\ \mu s/\mu m^2$ is much reduced with respect to the magnitude at $b = 1000\ \mu s/\mu m^2$, in contrast to the dendrite branches, where the difference in the signal magnitude between the two b-values is not nearly as significant. For the soma, at the long diffusion time, there is not a large reduction in the signal magnitude between $b = 1000\ \mu s/\mu m^2$ and $b = 4000\ \mu s/\mu m^2$.

By visual inspection, at the lower b-value, the signal in the whole neuron is close to the volume-weighted sum of the signals from the three cell components (the soma, the upper dendrite branch, and the lower dendrite branch). A quantitative study is conducted in the next section.

3.4.2 Exchange between soma and dendrites

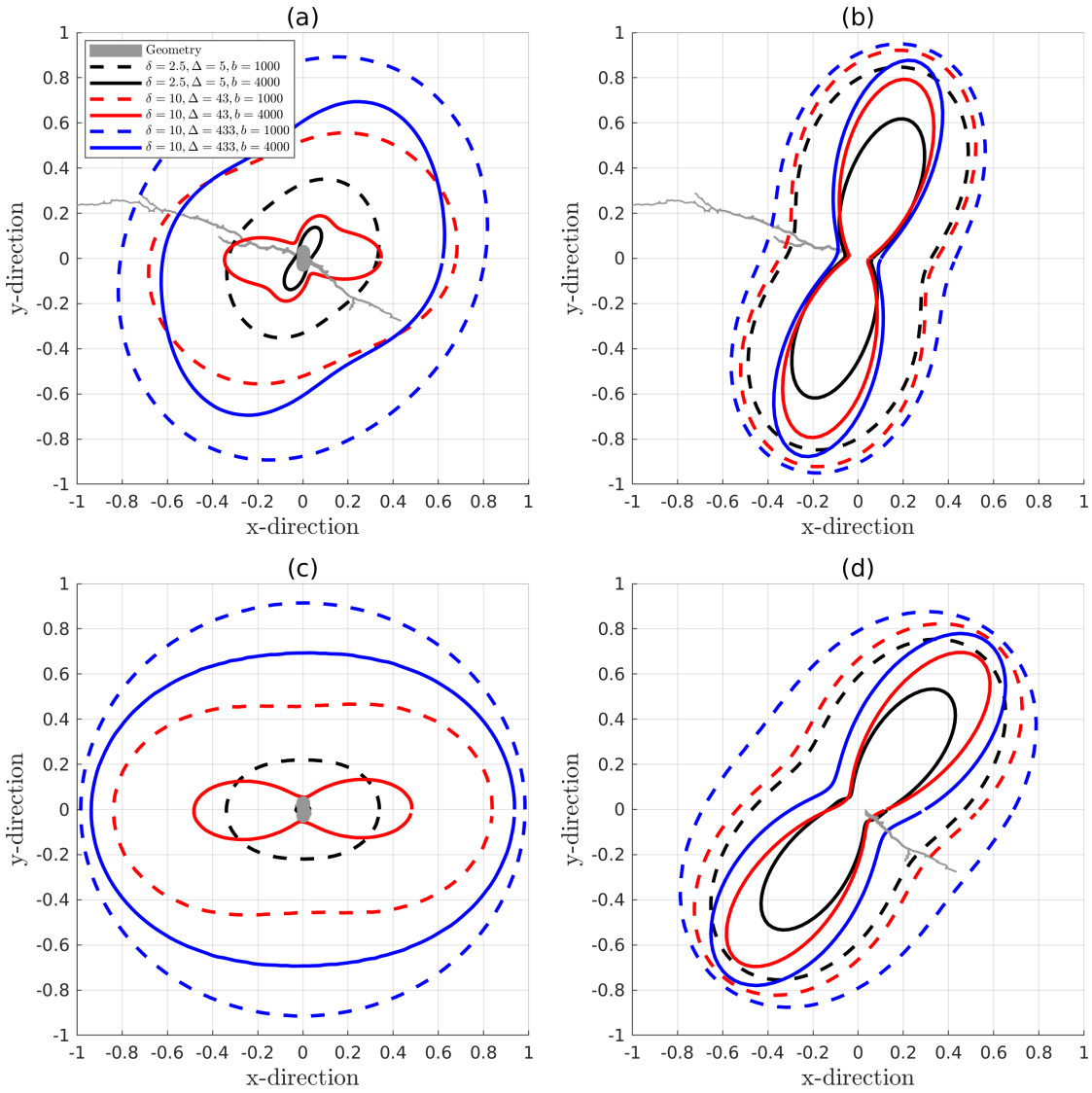


Figure 3.7: The signal attenuation in 180 directions lying on the $x - y$ plane, uniformly distributed on the unit circle. The distance from each data point to the origin represents the magnitude of the signal attenuation. The simulation parameters are $rtol = 10^{-3}, atol = 10^{-5}, H = 0.5\ \mu m^3$. The diffusion coefficient is $2 \times 10^{-3}\ \mu m^2/\mu s$. **(a)** the whole neuron (19425 vertices). **(b)** the dendrite1 (10825 vertices). **(c)** the soma (5842 vertices). **(d)** the dendrite2 (6444 vertices).

Here we compute the volume-weighted composite signal of the 3 cellular parts

$$E_{composite} = \frac{V_{soma}E_{soma} + V_{dendrite1}E_{dendrite1} + V_{dendrite2}E_{dendrite2}}{V_{neuron}} \tag{3.45}$$

and compare it to the signal attenuation of the whole neuron in the different gradient directions. In fig. 3.8, we see that the signal difference between the two is larger at longer diffusion times and higher b-values. The error also presents a gradient-direction dependence. According to fig. 3.7(a) and fig. 3.8, we can see that the error is larger in the direction parallel to the longitudinal axis of the neuron than in the direction perpendicular to the longitudinal axis. It is apparent that the exchange effect depends not only on diffusion times but also on the bvalues. The exchange effect is not negligible, especially for long diffusion times and high bvalues.

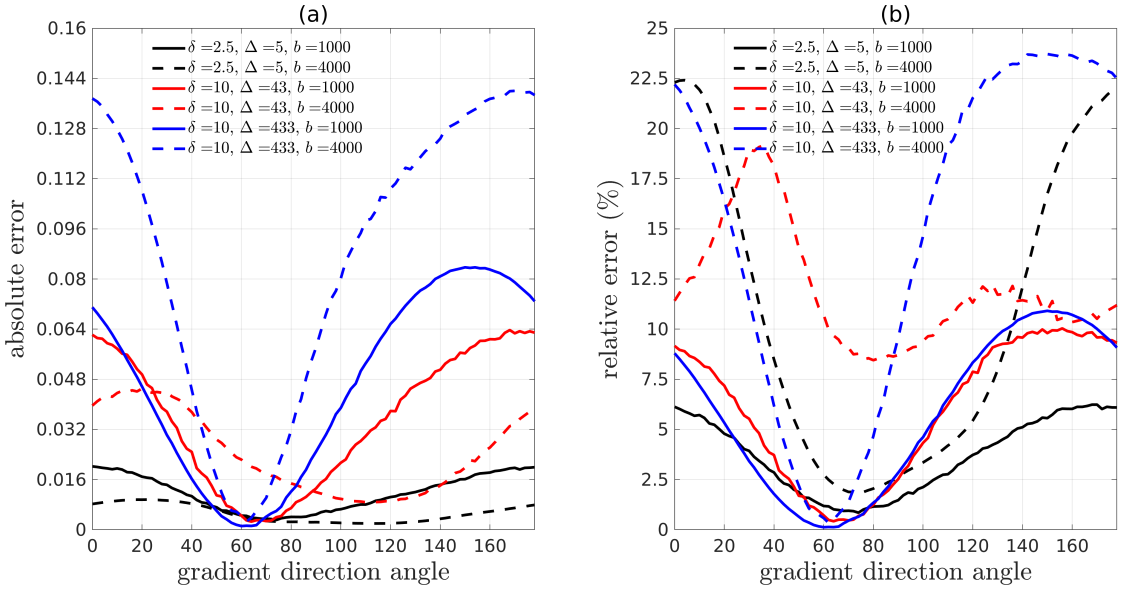


Figure 3.8: **(a)** The absolute error between volume-weighted composite signal and whole neuron signal. **(b)** The relative error between volume-weighted composite signal and whole neuron signal. Ninety gradient directions uniformly placed on the unit semi-circle in the $x-y$ plane were simulated. The gradient direction angle is given with respect to the x-axis. The position of the neuron can be seen in fig. 3.7(a).

3.4.3 Power-law scaling

In the work of Veraart et al. [85], it was shown that signal attenuations of tubular structures such as axons exhibit a certain high b-value behavior, namely, the direction-averaged signal, $\overline{E}$, is linear in $\frac{1}{\sqrt{b}}$ at high b-values:

$$\overline{E} \equiv \int_{\|\boldsymbol{u_g}\|=1} E d\boldsymbol{u_g} \sim c_0 + c_1 \frac{1}{\sqrt{b}}, \tag{3.46}$$

with c_0 the y-intercept and c_1 the slope of the linear function. The linear relationship eq. (3.46) is often referred to as the power-law scaling of direction-averaged signals. Because the dendrites of neurons also have a tubular structure, we test whether the direction-averaged signal $\overline{E}$ of dendrite branches also exhibits the power-law scaling. We computed $\overline{E}$ for the whole neuron as well as its two dendrite branches, averaging over 120 gradient directions uniformly distributed in the unit sphere. The results are shown in fig. 3.9. We see clearly the linear relationship between $\overline{E}$ and $\frac{1}{\sqrt{b}}$ in the dendrite branches for b-values in the range $2500\ \mu s/\mu m^2 \leq b \leq 20000\ \mu s/\mu m^2$. In contrast, in the whole neuron, due to the presence of the soma, such a linear relationship is not exhibited. By simulating for both $D_0 = 2 \times 10^{-3}\ \mu m^2/\mu s$ and $D_0 = 1 \times 10^{-3}\ \mu m^2/\mu s$ we see that the fitted slope c_1 is close to $\frac{1}{\sqrt{D_0}}$.

3.4.4 Markers of the soma size

As we have shown in fig. 3.9, the linear relationship between $\overline{E}$ and $\frac{1}{\sqrt{b}}$, i.e., the power-law scaling of direction-averaged signals [85], doesn't hold due to the presence of the soma and the exchange effects between the soma and the dendrites. The breakdown of the power-law scaling is also observed in [86, 88, 99]. By leveraging the collection of the realistic neuron meshes in Neuron Module, we statistically show that the deviation from the power-law scaling allows us to define several markers for revealing the soma size.

To do this, we conducted the following simulations that are slightly different from the constant (δ,Δ) experiments in [85, 86, 88] and shown in fig. 3.9. The signals are numerically computed using numerical MF. In the following, we held the gradient intensity constant, $g = 37\ mT/m$, and varied δ to obtain a wide range of b-values, all the while choosing $\Delta = \delta$ (PGSE sequence). The simulations were conducted in 64 gradient directions, and the signals were averaged over these directions. This was performed for the full set of 65 neuron meshes in the Neuron Module.

In fig. 3.10, we show an example of the simulated signal curve and the power-law approximation for the neuron *03a_spindle2aFI*. From the direction-averaged simulated signals, we find the inflection point (blue dot) of the signal curve (black

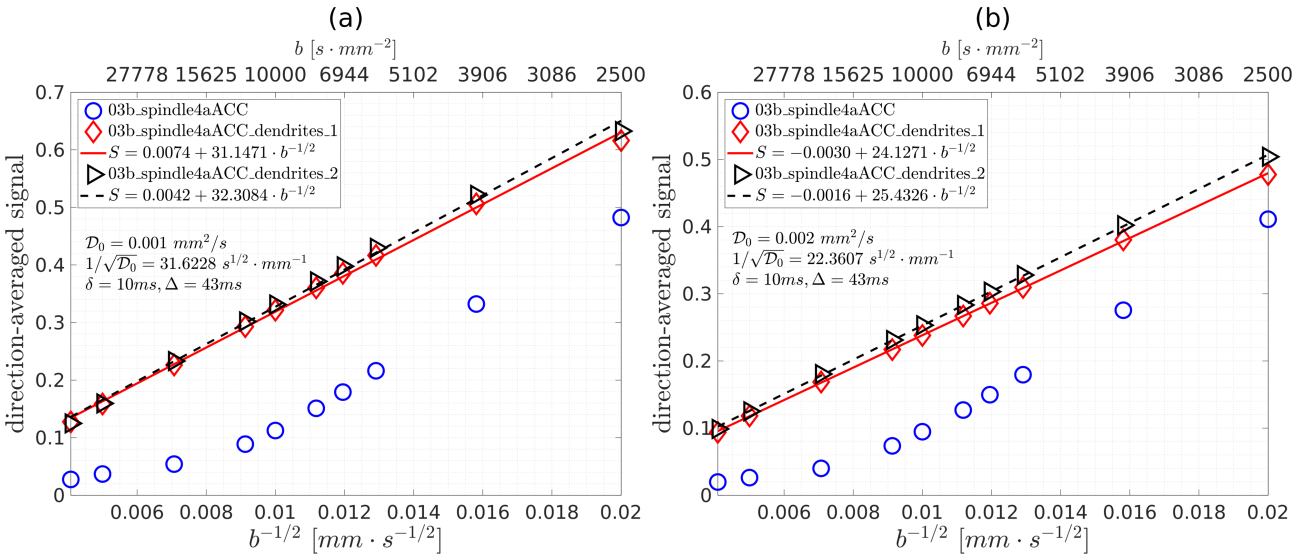


Figure 3.9: The direction-averaged signal for the neuron *03b_spindle4aACC*. The $\overline{E}$ is averaged over 120 diffusion directions, uniformly distributed in the unit sphere. The simulation parameters are $rtol = 10^{-3}, atol = 10^{-5}, Htetgen = 0.5\ \mu m^3$. The diffusion-encoding sequence is PGSE ($\delta/\Delta = 10/43\ ms$). The b-values are $b = \{60000, 40000, 20000, 12000, 10000, 8000, 7000, 6000, 4000, 2500\}\ \mu s/\mu m^2$. **(a)** $D_0 = 1 \times 10^{-3}\ \mu m^2/\mu s$. **(b)** $D_0 = 2 \times 10^{-3}\ \mu m^2/\mu s$.

curve). We fit the power-law approximation (straight blue dashed line) around the inflection point. The power-law region is the range where the relative error between the simulated signal curve and the power-law fit is less than 2% (width of the yellow region), and the approximation error is estimated by the area between the signal curve and the power-law fit to the left of the inflection point (the green area).

In order to characterize the influence of soma on the power-law approximation, we chose the following six candidate markers:

- x_0: the x-coordinate of the inflection point;
- y_0: the y-coordinate of the inflection point;
- c_0: the y-intercept of the power-law fit;
- c_1: the slope of the power-law fit;
- $\mathcal{E}$: the power-law approximation error;
- w: the width of the power-law region.

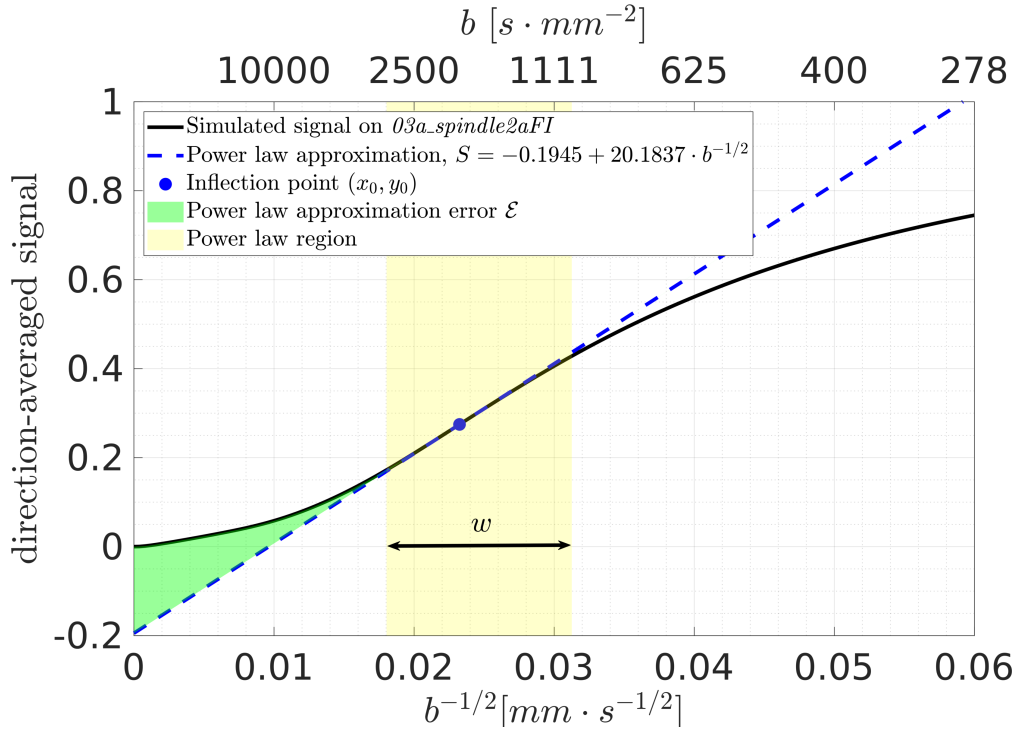


Figure 3.10: The direction-averaged signal curve for the neuron *03a_spindle2aFI*. The signals are computed using the numerical MF within the SpinDoctor Toolbox. The $\overline{E}$ was averaged over 64 diffusion directions, uniformly distributed in the unit sphere. The b-values are greater than 278 $\mu s/\mu m^2$ and the diffusivity is $D_0 = 2 \times 10^{-3}\ \mu m^2/\mu s$. The gradient intensity is constant, $g = 37\ mT/m$, and δ was varied to obtain a wide range of b-values, all the while choosing $\Delta = \delta$ (PGSE sequences). The blue dot indicates the inflection point of the simulated signal curve. The power-law fit is performed around the inflection point. The power-law region is the width of the range where the relative error between the simulated signal and the power-law approximation is less than 2%. The area between the simulated curve and the power-law fit to the left of the inflection point represents the approximation error of the power-law fit.

A statistical study of the above 6 candidate markers on the collection of the 65 neurons in the Neuron Module was performed. Since the undersampling when $\frac{1}{\sqrt{b}}$ approaches 0 could produce a significant numerical error, we only kept the neurons whose x_0 are greater than $0.016\ mm \cdot s^{-1/2}$. In total, 28 spindle neurons and 21 pyramidal neurons were retained.

We first plot the candidate markers with respect to the soma volume v_{soma} in fig. 3.11. Each data point in the figure corresponds to a neuron (for a total of 49). It can be seen that x_0, c_0, c_1, $\mathcal{E}$, and w exhibit an exponential relationship with the soma volume. The fitted equations allow us to infer the soma volume

by measuring the markers. We also see that y_0 is not a biomarker for the soma volume. Similarly, we show the scatter plot of the candidate markers with respect to the soma volume fraction f_{soma} in fig. 3.12. In this case, the x_0, c_1, and w are not markers of the soma volume fraction. The candidate markers y_0, c_0, and $\mathcal{E}$ seem capable of indicating the lower bound for the soma volume fraction.

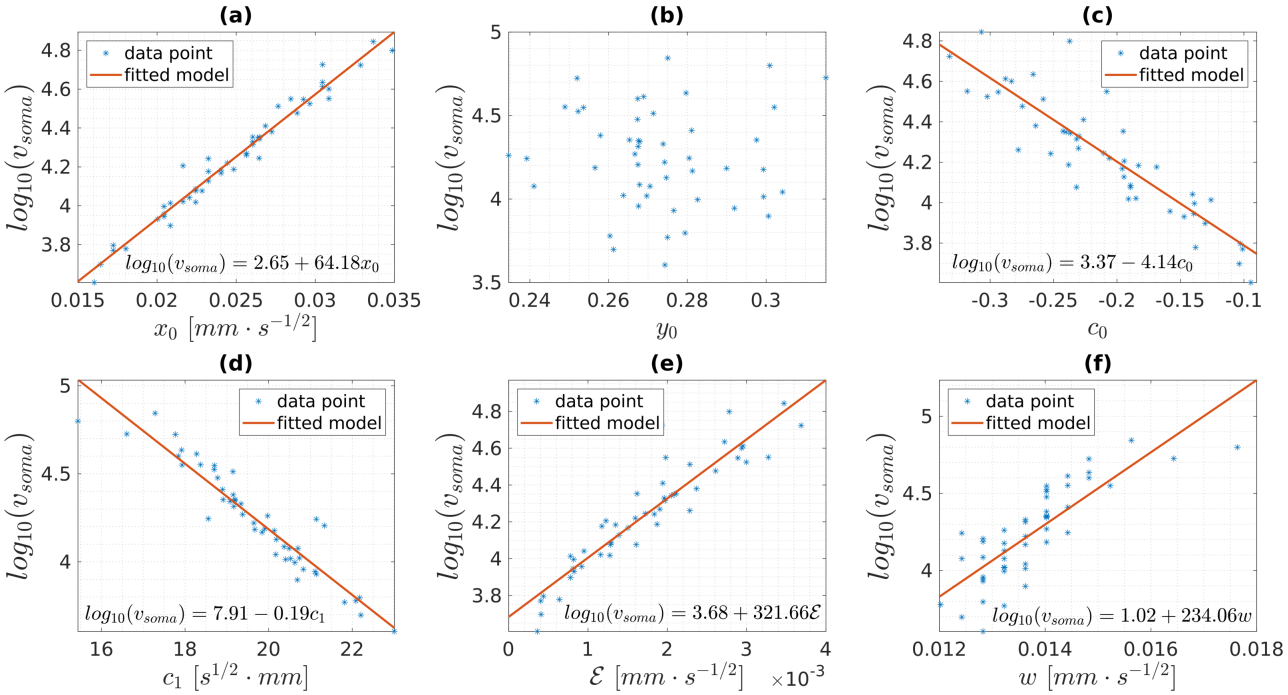


Figure 3.11: The scatter plots of the logarithm of soma volume with respect to the six markers. Each blue dot represents the data from one of the 49 neurons (28 spindle neurons and 21 pyramidal neurons) retained for this study. **(a)** the x-coordinate of the inflection point x_0. **(b)** the y-coordinate of the inflection point y_0. **(c)** the y-intercept of the power-law fit c_0. **(d)** the slope of the power-law fit c_1. **(e)** the power-law approximation error $\mathcal{E}$. **(f)** the width of the power-law region w.

3.5 Summary

This chapter concerns three subjects: (1) presenting the Bloch-Torrey equation and the numerical methods to solve it; (2) optimizing the numerical matrix formalism; (3) demonstrating the benefits of diffusion MRI simulation of realistic neurons.

First, we introduced a general form of the Bloch-Torrey equation. By assuming that compartment interfaces are impermeable and the transverse relaxation

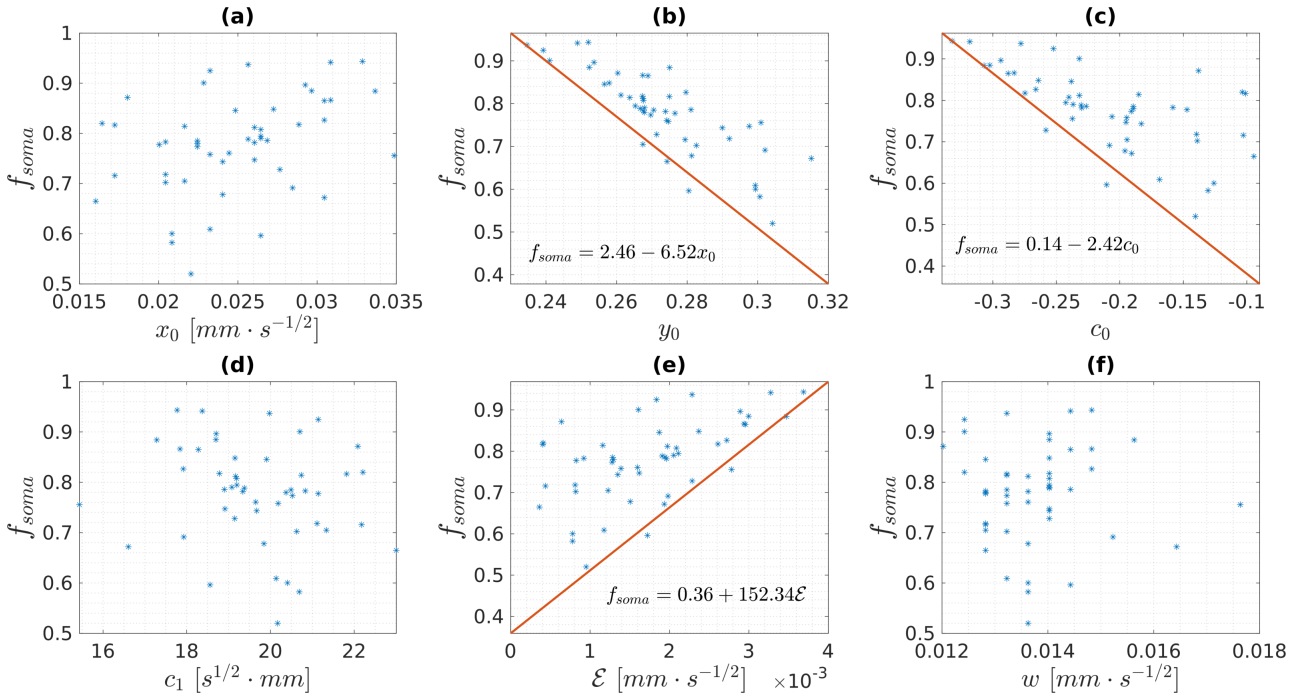


Figure 3.12: The scatter plots of the logarithm of soma volume fraction with respect to the six markers.

is homogeneous, we obtained a simplified form of the BT equation concerning three main attenuation mechanisms: diffusion, gradient dephasing, and boundary restriction. Then, we presented two numerical methods, i.e., finite element method and numerical matrix formalism, for solving the BT equation. The numerical MF, combining the advantages of classical MF and FEM, is of great numerical interest.

Second, we optimized the computational efficiency of the numerical MF by adopting an appropriate eigendecomposition algorithm and leveraging GPU computation. The numerical MF after optimization is ten times more efficient than the previous version (see fig. 3.4). The optimized method is then compared with FEM and a GPU-accelerated Monte-Carlo method. The efficiency advantage of the optimized numerical MF is significant. With comparable accuracy, the numerical MF is **20** times faster than FEM and **65** times faster than the Monte-Carlo method (see table 3.2). Furthermore, the optimized numerical MF can compute hundreds of dMRI signals from a neuron mesh in a few minutes, making large-scale dMRI simulations practical. The optimization of the numerical matrix formalism is the second contribution of this thesis.

Finally, we performed dMRI simulations on the realistic neuron meshes constructed in chapter 2. Using segmented neuron meshes in Neuron Module, we

showed that the water exchange effect is not negligible, especially at long diffusion times or high bvalues. In addition, we demonstrated that the power-law scaling holds despite the structural deviation of the dendrites from cylinders. However, the power-law scaling pattern of the intracellular signals is modified by the presence of the soma. Leveraging the deviation of intracellular signals from the power-law scaling, we defined six measurable markers that can be statistically related to the soma size. The results of these numerical experiments confirm the possibility of adopting a simulation-driven approach for brain microstructure imaging.

The ultimate goal of this thesis is to facilitate simulation-driven brain microstructure estimation. This chapter contributed to this goal by providing ultrafast numerical matrix formalism. Besides, the statistical analysis conducted in section 3.4.4 suggests the existence of underlying mappings between dMRI signals and the microstructure properties of interest. This inspires us to employ machine learning techniques to approximate the underlying mappings. We present the simulation-based brain microstructure imaging using artificial neural networks in chapter 5.

As a spectral method, the numerical matrix formalism has demonstrated its theoretical and numerical advantages in this chapter. The Laplacian eigenbases adopted in numerical MF are geometry-specific. We wonder if it is possible to decompose the diffusion MRI signal on a Fourier-type basis. Contrary to the Laplacian eigenbasis, the Fourier basis functions do not depend on the geometrical configuration. This independence could allow for comparing various geometries and provide a new spectral perspective. The next chapter presents a new numerical method for solving the BT equation, which relies on potential theory and the decomposition of dMRI signals on a Fourier-type basis.

Chapter 4

Solving Bloch-Torrey Equation with Potential Theory

Existing simulation frameworks use the finite element methods or the Matrix Formalism method to solve the BT equation. As the third contribution of this thesis, we proposed a new method based on the efficient evaluation of layer potentials. In this chapter, the mathematical framework and the numerical implementation of the new method are described. We demonstrate the convergence of our method via numerical experiments and analyze the errors linked to various model and simulation parameters. Since our method provides a Fourier-type representation of the diffusion MRI signal, it can potentially facilitate new physical and biological signal interpretations in the future. This chapter is mainly based on the work in [201].

4.1 Introduction

We propose a new method based on potential theory from classical mathematics that provides a Fourier-type representation of the diffusion MRI signal. The main challenge of this method involves the fundamental solution of the diffusion equation, also known as the heat kernel, which has a singularity in time. In theory, infinite Fourier modes are required to represent the heat kernel due to the singularity, while only finite Fourier modes are accessible for practical computation. This practical limitation may lead to the Gibbs phenomenon that could degrade the approximation accuracy [202]. In order to overcome this challenge, we follow the path of several previous works [203–206] focusing on the evaluation of heat potentials. In particular, in [203], the authors proposed several fundamental ideas, such as (1) splitting the heat potential into a local in time part and a history part in order to overcome the singularity of the heat kernel; (2) ap-

proximating the local in time part by asymptotics; (3) leveraging the exponential decay of the history part to represent it using a few Fourier modes. These ideas are crucial to the Fourier-type representation of the diffusion MRI signal derived in this chapter.

Despite the intrinsic similarity between thermal conduction and diffusion process, in the literature, there have not been previous works about the representation of the diffusion MRI signal via potential theory, and certainly not by using a Fourier basis for layer potentials. As the first work addressing this subject, we restrict ourselves to the 2D setting with impermeable interfaces. We also restrict ourselves to simplified conditions on the diffusion-encoding gradient. Specifically, we derive our method under the narrow pulse assumption, where the diffusion-encoding pulse duration is very short compared to the delay between the pulses. These two assumptions allow us to apply the theory developed for the diffusion kernel to the diffusion MRI application.

We call our method the Fourier Potential Method (FPM). The main steps of our method are:

1. transforming the BT equation to the diffusion equation using the narrow pulse assumption on the diffusion-encoding sequence;
2. formulating the solution of the diffusion equation using the single layer potential;
3. approximating the singular part of the single layer potential using an asymptotic expansion and solving the integral equation;
4. storing the non-singular part of the single layer potential using the Fourier coefficients, leveraging the fast decay in the Fourier spectrum;
5. computing the diffusion MRI signal using the above representation.

The chapter is organized as follows. Section 4.2 introduces the mathematical framework of FPM. Section 4.3 describes the Fourier Potential Method and error analysis. Section 4.4 contains numerical results, including convergence in the various simulation parameters. Section 4.5 contains conclusion and future perspectives. In the appendix, chapter C shows a flowchart and a streamlined description of the numerical implementation.

4.2 Mathematical framework

We aim to simulate the diffusion MRI signal due to spins inside biological cells and assume that the spin exchange across cell membranes is negligible under

the simulation conditions. The impermeability assumption allows us to focus on solving the BT equation on a connected domain. Let Ω be the 2D computational domain, and let $\Gamma = \partial\Omega$ be the domain boundary.

4.2.1 Bloch-Torrey equation

We recall the simplified form of the BT equation:

$$\frac{\partial}{\partial t}\varphi(\boldsymbol{x},t) = -\imath\gamma g f_1(t)\boldsymbol{u}_g \cdot \boldsymbol{x}\,\varphi(\boldsymbol{x},t) + D_0\nabla\cdot\nabla\varphi(\boldsymbol{x},t), \qquad \boldsymbol{x}\in\Omega, \tag{4.1}$$

$$D_0\nabla\varphi(\boldsymbol{x},t)\cdot\boldsymbol{n} = 0, \qquad \boldsymbol{x}\in\partial\Omega, \tag{4.2}$$

$$\varphi(\boldsymbol{x},0) = \rho, \qquad \boldsymbol{x}\in\Omega, \tag{4.3}$$

where D_0 is the intrinsic diffusion coefficient in the domain Ω and $\rho = 1$ the initial magnetization.

We adopt the PGSE sequences [18] in this chapter. The PGSE time profile f_1 is shown in fig. 3.2(a). If the rectangular pulses are narrow, i.e., $\delta \ll \Delta$, the BT equation can be transformed into the diffusion equation. This assumption is called the narrow pulse approximation [18].

The diffusion MRI signal is measured at echo time TE for PGSE. The signal attenuation E is the integral of $\varphi(\boldsymbol{x}, TE)$ divided by the area of Ω:

$$E = \frac{1}{|\Omega|}\int_{\boldsymbol{x}\in\Omega}\varphi(\boldsymbol{x},TE)\,d\boldsymbol{x}. \tag{4.4}$$

4.2.2 Narrow pulse approximation

We restrict ourselves to simplified conditions on the diffusion-encoding gradient. Specifically, we derive our method under the narrow pulse assumption, where the pulse duration is very short compared to the inter-pulse duration [18], i.e., $\delta \ll \Delta$. This assumption enables us to ignore molecular diffusion during the pulses of PGSE.

Let us consider spins initially located at $\boldsymbol{x}$. After the first pulse of PGSE, the complex phase of these spins is $e^{-\imath\delta\gamma\boldsymbol{g}\cdot\boldsymbol{x}}$. This means the complex magnetization at $t = \delta$ due to a uniform initial distribution ρ can be written as:

$$\varphi(\boldsymbol{x},\delta) \approx \rho\, e^{-\imath\delta\gamma\boldsymbol{g}\cdot\boldsymbol{x}}, \quad \boldsymbol{x}\in\Omega. \tag{4.5}$$

Because the magnetic field gradient is turned off after the first pulse, i.e., $\boldsymbol{g} = \boldsymbol{0}$, the magnetization between pulses satisfies the diffusion equation:

$$\frac{\partial}{\partial t}\varphi(\boldsymbol{x},t) = D_0\nabla\cdot\nabla\varphi(\boldsymbol{x},t), \quad \boldsymbol{x}\in\Omega, t\in[\delta,\Delta], \tag{4.6}$$

subject to the zero Neumann boundary condition:

$$D_0 \nabla \varphi(\boldsymbol{x}_0, t) \cdot \boldsymbol{n} = 0, \quad \boldsymbol{x}_0 \in \partial\Omega, t \in [\delta, \Delta], \tag{4.7}$$

where $\boldsymbol{n}$ is the unit outward pointing normal vector at $\boldsymbol{x}_0$, and initial condition at $t = \delta$ is:

$$\varphi(\boldsymbol{x}, \delta) = \rho\, e^{-\imath\delta\gamma \boldsymbol{g}\cdot\boldsymbol{x}}, \quad \boldsymbol{x} \in \Omega. \tag{4.8}$$

During the second pulse, at the point $\boldsymbol{x}$, the additional accumulated complex phase is $e^{\imath\delta\gamma\boldsymbol{g}\cdot\boldsymbol{x}}$, so the magnetization at the position $\boldsymbol{x}$ and time TE is:

$$\varphi(\boldsymbol{x}, TE) \approx \varphi(\boldsymbol{x}, \Delta) e^{\imath\delta\gamma\boldsymbol{g}\cdot\boldsymbol{x}}, \quad \boldsymbol{x} \in \Omega. \tag{4.9}$$

We emphasize again that we used the assumption $\delta \ll \Delta$. The echo-time TE is usually some time after the end of the second pulse (i.e., $TE \geq \Delta + \delta$).

The signal attenuation E is proportional to the total magnetization measured at the echo time:

$$E = \frac{1}{|\Omega|} \int_{\boldsymbol{x}\in\Omega} \varphi(\boldsymbol{x}, \Delta) e^{\imath\delta\gamma\boldsymbol{g}\cdot\boldsymbol{x}}\, d\boldsymbol{x}. \tag{4.10}$$

4.3 Method

Our method aims to solve eqs. (4.6) to (4.8) based on potential theory, which, in the meantime, provides a Fourier-type representation of the diffusion MRI signal. We derive our new method below. In the appendix, chapter C shows a flowchart and a streamlined description of the numerical implementation of FPM.

4.3.1 Solution of the diffusion equation and the diffusion MRI signal

Before we solve the diffusion equation using potential theory, we transform the initial and boundary conditions. We transform the diffusion equation in eqs. (4.6) to (4.8) such that it is subject to zero initial conditions and complex-valued non-zero Neumann boundary conditions. Define

$$\omega(\boldsymbol{x}, t) \equiv \varphi(\boldsymbol{x}, t + \delta) - \rho e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 t} e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{x}}, \quad \boldsymbol{x} \in \Omega, t \in [0, \Delta - \delta], \tag{4.11}$$

where $\mathbf{q} = \delta\gamma\boldsymbol{g}/2\pi$. We will work on the quantity $\omega(\boldsymbol{x}, t)$ defined in eq. (4.11), which satisfies the diffusion equation:

$$\frac{\partial}{\partial t}\omega(\boldsymbol{x}, t) = D_0 \nabla \cdot \nabla \omega(\boldsymbol{x}, t), \quad \boldsymbol{x} \in \Omega, t \in [0, \Delta - \delta], \tag{4.12}$$

subject to non-homogeneous Neumann boundary conditions:

$$D_0 \nabla \omega(\boldsymbol{x}_0, t) \cdot \boldsymbol{n} = D_0 \mathcal{N}(\boldsymbol{x}_0, t, \mathbf{q}) \quad \boldsymbol{x}_0 \in \partial\Omega, t \in [0, \Delta - \delta], \tag{4.13}$$

and zero initial conditions:

$$\omega(\boldsymbol{x}, 0) = 0, \quad \boldsymbol{x} \in \Omega. \tag{4.14}$$

The Neumann forcing term is complex-valued, periodic in space in the direction $\mathbf{q}$, and decays exponentially in time:

$$\mathcal{N}(\boldsymbol{x}_0, t, \mathbf{q}) \equiv 2\pi\rho\, \mathbf{q} \cdot \boldsymbol{n} \left(\imath e^{-2\pi\imath \mathbf{q}\cdot\boldsymbol{x}_0}\right) e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 t}. \tag{4.15}$$

The signal attenuation E can be reformulated in terms of ω:

$$E = \rho e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 (\Delta - \delta)} + \frac{1}{|\Omega|} \int_{\boldsymbol{x}\in\Omega} \omega(\boldsymbol{x}, \Delta - \delta) e^{2\pi\imath \mathbf{q}\cdot\boldsymbol{x}} d\boldsymbol{x}. \tag{4.16}$$

In the above, the first term is explicit, and the second term needs to be computed numerically. We define a time-dependent integral whose value at $t = \Delta - \delta$ gives the second term:

$$\overline{\omega}(\mathbf{q}, t) \equiv \int_{\boldsymbol{x}\in\Omega} \omega(\boldsymbol{x}, t) e^{2\pi\imath \mathbf{q}\cdot\boldsymbol{x}} d\boldsymbol{x}, t \in [0, \Delta - \delta]. \tag{4.17}$$

The function $\overline{\omega}$ can be expanded by the Green's second identity:

$$\begin{aligned} \overline{\omega}(\mathbf{q}, t) &= \frac{-1}{4\pi^2 D_0 \|\mathbf{q}\|^2} \left(\int_\Omega D_0 \nabla \cdot \nabla \omega(\boldsymbol{x}, t) e^{2\pi\imath \mathbf{q}\cdot\boldsymbol{x}} d\boldsymbol{x} + B \right), \\ B &= \int_{\partial\Omega} 2\pi\imath D_0 \mathbf{q} \cdot \boldsymbol{n} \omega(\boldsymbol{x}, t) e^{2\pi\imath \mathbf{q}\cdot\boldsymbol{x}} ds_{\boldsymbol{x}} - \int_{\partial\Omega} D_0 \nabla \omega(\boldsymbol{x}, t) \cdot \boldsymbol{n} e^{2\pi\imath \mathbf{q}\cdot\boldsymbol{x}} ds_{\boldsymbol{x}}. \end{aligned} \tag{4.18}$$

Using the diffusion equation and the nonhomogeneous Neumann boundary conditions, we get an ordinary differential equation for $\overline{\omega}$:

$$\frac{d}{dt}\overline{\omega}(\mathbf{q}, t) = -4\pi^2 D_0 \|\mathbf{q}\|^2 \overline{\omega}(\mathbf{q}, t) - 2\pi\imath D_0 \int_{\partial\Omega} \mathbf{q} \cdot \boldsymbol{n} \omega(\boldsymbol{x}, t) e^{2\pi\imath \mathbf{q}\cdot\boldsymbol{x}} ds_{\boldsymbol{x}}, \tag{4.19}$$

which has an analytical solution:

$$\begin{aligned} \overline{\omega}(\mathbf{q}, t) &= -D_0 \int_{\partial\Omega} \int_0^t 2\pi\imath \mathbf{q} \cdot \boldsymbol{n} e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 (t-\tau)} \omega(\boldsymbol{x}, \tau) e^{2\pi\imath \mathbf{q}\cdot\boldsymbol{x}} d\tau ds_{\boldsymbol{x}}, \\ &= D_0 \rho^{-1} \int_{\partial\Omega} \int_0^t \mathcal{N}^*(\boldsymbol{x}, t - \tau, \mathbf{q}) \omega(\boldsymbol{x}, \tau) d\tau ds_{\boldsymbol{x}}. \end{aligned} \tag{4.20}$$

The asterisk symbol $*$ denotes the complex conjugation. It can be proved that eq. (4.20) satisfies a recursive relationship in time:

$$\overline{\omega}(\mathbf{q},t) = e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t}\overline{\omega}(\mathbf{q},t-\Delta t) + D_0\rho^{-1}\int_{\partial\Omega}\int_{t-\Delta t}^{t}\mathcal{N}^*(\boldsymbol{x},t-\tau,\mathbf{q})\omega(\boldsymbol{x},\tau)d\tau ds_{\boldsymbol{x}}. \tag{4.21}$$

Equation (4.17) and eq. (4.20) are mathematically equivalent for evaluating the diffusion MRI signal (at $t=\Delta-\delta$). It can be seen that, while eq. (4.17) requires the value of ω on the entire domain Ω, eq. (4.20) only needs the value of ω on the boundary, which is more computationally efficient. The recursion in time above also increases computational efficiency. We will use the method of layer potentials to get the boundary values of ω in the next section.

4.3.2 The single layer potential representation

The PDE in eqs. (4.12) to (4.14) has Neumann boundary conditions, zero initial conditions, and zero forcing term, allowing us to represent the solution $\omega(\boldsymbol{x},t)$ as a single layer potential, with a density function μ defined on $\partial\Omega$ [206]. In other words, $\omega(\boldsymbol{x},t) = S[\mu](\boldsymbol{x},t)$. The definition of the single layer potential is

$$\omega(\boldsymbol{x},t) = S[\mu](\mathbf{x},t) \equiv \int_0^t\int_{\partial\Omega} D_0 G(\mathbf{x}-\boldsymbol{y},t-\tau)\mu(\boldsymbol{y},\tau)ds_{\boldsymbol{y}}d\tau, \tag{4.22}$$

where $G(\mathbf{x},t)$ is the fundamental solution of the 2D diffusion equation in a box $[-L_1/2,L_1/2]\times[-L_2/2,L_2/2]$, with periodic boundary conditions. The fundamental solution $G(\mathbf{x},t)$ has two equivalent representations [203]:

$$G_{Gauss}(\mathbf{x},t) = (4\pi D_0 t)^{-1}\sum_{\mathbf{z}\in\mathbb{Z}^2} e^{-\frac{\|\mathbf{x}-\mathbf{z}\odot\mathbf{L}\|^2}{4D_0 t}}, \tag{4.23}$$

$$G_{Fourier}(\mathbf{x},t) = \frac{1}{L_1 L_2}\sum_{\substack{\boldsymbol{\nu}=\mathbf{z}\oslash\mathbf{L}\\ \mathbf{z}\in\mathbb{Z}^2}} e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2 t}e^{2\pi\imath\boldsymbol{\nu}\cdot\mathbf{x}}, \tag{4.24}$$

where $\odot$ and $\oslash$ are Hadamard product and Hadamard division, respectively, and $\mathbf{L} = [L_1,L_2]^T$. For the convenience of notation, in the following, we set $L_1 = L_2 = L$ and note by $\Delta\nu = \frac{1}{L}$. In this way, we rewrite eq. (4.24) as

$$G_{Fourier}(\mathbf{x},t) = \sum_{\substack{\boldsymbol{\nu}=\mathbf{z}\oslash\mathbf{L}\\ \mathbf{z}\in\mathbb{Z}^2}} e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2 t}e^{2\pi\imath\boldsymbol{\nu}\cdot\mathbf{x}}\Delta\nu^2 \tag{4.25}$$

in order to recall its relationship with the Fourier transform. The imposition of periodic boundary conditions on the faces of the box allows us to use the discrete Fourier series.

The density function μ is chosen to be a causal function and is determined by imposing the Neumann boundary conditions on the geometry boundary $\partial\Omega$ [205]:

$$\lim_{\boldsymbol{x}\to\boldsymbol{x}_0\in\partial\Omega} \nabla S[\mu](\boldsymbol{x},t)\cdot\boldsymbol{n} = \mathcal{N}(\boldsymbol{x}_0,t,\mathbf{q}), \quad \boldsymbol{x}_0\in\partial\Omega, t\in[0,\Delta-\delta]. \tag{4.26}$$

Using the jump property of the trace of the double layer potential, the integral equation to be solved for μ is then the following:

$$\frac{1}{2}\mu(\boldsymbol{x}_0,t) + K[\mu](\boldsymbol{x}_0,t) = \mathcal{N}(\boldsymbol{x}_0,t,\mathbf{q}), \quad \boldsymbol{x}_0\in\partial\Omega, t\in[0,\Delta-\delta], \tag{4.27}$$

with

$$K[\mu](\boldsymbol{x}_0,t) \equiv \int_0^t\int_{\partial\Omega} D_0\frac{\partial G}{\partial \boldsymbol{n}_{\boldsymbol{x}_0}}(\boldsymbol{x}_0-\boldsymbol{y},t-\tau)\mu(\boldsymbol{y},\tau)ds_{\boldsymbol{y}}d\tau \tag{4.28}$$

being the principal value integral on the boundary. Solving the integral equation eq. (4.27) for μ plays a pivotal role in our method. We present the detailed steps in the next sections.

4.3.3 Splitting the single layer potential into local and history parts

The single layer potential $S[\mu]$ is split into a history part, $S_{long}[\mu]$, and a local in time part, $S_{short}[\mu]$. Since the local in time part $S_{short}[\mu]$ contains the singularity of the fundamental solution G, we approximate it by asymptotic formulas. The asymptotic trace formulas are only accurate in an interval near the singularity, so we limit their use to the interval $[t-\eta,t]$, with η being a small quantity to be determined later. In other words,

$$S[\mu](\mathbf{x},t) = S_{short}[\mu](\mathbf{x},t) + S_{long}[\mu](\mathbf{x},t), \tag{4.29}$$

with

$$S_{short}[\mu](\mathbf{x},t) \equiv \int_{t-\eta}^t\int_{\partial\Omega} D_0 G_{Gauss}(\mathbf{x}-\boldsymbol{y},t-\tau)\mu(\boldsymbol{y},\tau)ds_{\boldsymbol{y}}d\tau, \tag{4.30}$$

$$S_{long}[\mu](\mathbf{x},t) \equiv \int_0^{t-\eta}\int_{\partial\Omega} D_0 G_{Fourier}(\mathbf{x}-\boldsymbol{y},t-\tau)\mu(\boldsymbol{y},\tau)ds_{\boldsymbol{y}}d\tau. \tag{4.31}$$

Similarly, we decompose $K[\mu]$ into 2 parts:

$$K[\mu](\boldsymbol{x}_0,t) = K_{short}[\mu](\boldsymbol{x}_0,t) + K_{long}[\mu](\boldsymbol{x}_0,t), \tag{4.32}$$

with

$$K_{short}[\mu](\boldsymbol{x}_0,t) \equiv \int_{t-\eta}^{t}\int_{\partial\Omega} D_0 \frac{\partial G_{Gauss}}{\partial \boldsymbol{n}_{\boldsymbol{x}_0}}(\boldsymbol{x}_0-\boldsymbol{y},t-\tau)\mu(\boldsymbol{y},\tau)ds_{\boldsymbol{y}}d\tau, \tag{4.33}$$

$$K_{long}[\mu](\boldsymbol{x}_0,t) \equiv \int_{0}^{t-\eta}\int_{\partial\Omega} D_0 \frac{\partial G_{Fourier}}{\partial \boldsymbol{n}_{\boldsymbol{x}_0}}(\boldsymbol{x}_0-\boldsymbol{y},t-\tau)\mu(\boldsymbol{y},\tau)ds_{\boldsymbol{y}}d\tau. \tag{4.34}$$

Next, we compute or approximate the above history and local parts.

Asymptotic trace formulas for the local part

Based on the expressions derived by Greengard and Strain [203], the asymptotic trace formulas in two dimensions for the local parts, when $t > \eta$, are:

$$S_{short}[\mu](\boldsymbol{x}_0,t) = \sqrt{\frac{D_0\eta}{\pi}}\mu(\boldsymbol{x}_0,t) + O(\eta^{3/2}), t > \eta \tag{4.35}$$

and

$$K_{short}[\mu](\boldsymbol{x}_0,t) = -\frac{\sqrt{D_0\eta}}{2\sqrt{\pi}}\xi(\boldsymbol{x}_0)\mu(\boldsymbol{x}_0,t) + O(\eta^{3/2}), t > \eta, \tag{4.36}$$

where $\xi(\boldsymbol{x}_0)$ is the curvature at the point $\boldsymbol{x}_0 \in \partial\Omega$. The boundary $\partial\Omega$, which models the cell membrane, is a closed 2D plane curve. We assume it is twice differentiable. Let $\psi(\alpha) = (x(\alpha), y(\alpha))$ be a parametric representation of $\partial\Omega$. We choose a general parameter α such that $\psi(\alpha)$ is oriented counterclockwise. The curvature at the point $\boldsymbol{x}_0 = \psi(\alpha_0)$ is defined as

$$\xi(\boldsymbol{x}_0) = \left.\frac{x'y'' - y'x''}{(x'^2+y'^2)^{3/2}}\right|_{\alpha=\alpha_0}, \tag{4.37}$$

where primes refer to derivatives with respect to α.

We also need to initialize values for $t \leq \eta$. It has been derived in [206] that the expressions are:

$$S_{short}[\mu](\boldsymbol{x}_0,t) = \sqrt{\frac{D_0 t}{\pi}}\mu(\boldsymbol{x}_0,t) + O(t^{3/2}), t \leq \eta, \tag{4.38}$$

and

$$K_{short}[\mu](\boldsymbol{x}_0,t) = -\frac{\sqrt{D_0 t}}{2\sqrt{\pi}}\xi(\boldsymbol{x}_0)\mu(\boldsymbol{x}_0,t) + O(t^{3/2}), t \leq \eta. \tag{4.39}$$

Fourier representation of history part

For the smooth part of the single layer potential, a Fourier representation for the Dirichlet trace is proposed in [203]:

$$S_{long}[\mu](\boldsymbol{x}_0,t) = D_0 \sum_{\substack{\boldsymbol{\nu}=\mathbf{z}\oslash\mathbf{L}\\ \mathbf{z}\in\mathbb{Z}^2}} \hat{f}(\boldsymbol{\nu},t) e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0}\Delta\nu^2, \tag{4.40}$$

and the Neumann trace is

$$K_{long}[\mu](\boldsymbol{x}_0,t) = D_0 \sum_{\substack{\boldsymbol{\nu}=\mathbf{z}\oslash\mathbf{L}\\ \mathbf{z}\in\mathbb{Z}^2}} 2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{n}\hat{f}(\boldsymbol{\nu},t) e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0}\Delta\nu^2, \tag{4.41}$$

where the Fourier coefficients are

$$\hat{f}(\boldsymbol{\nu},t) = \int_0^{t-\eta}\int_{\partial\Omega} e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2(t-\tau)}\mu(\boldsymbol{y},\tau)e^{-2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{y}}ds_{\boldsymbol{y}}d\tau. \tag{4.42}$$

To avoid history-dependent time integration, we use the following recurrence formula for the Fourier coefficients

$$\hat{f}(\boldsymbol{\nu},t) = e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t}\hat{f}(\boldsymbol{\nu},t-\Delta t) + \int_{t-\eta-\Delta t}^{t-\eta}\int_{\partial\Omega} e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2(t-\tau)}\mu(\boldsymbol{y},\tau)e^{-2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{y}}ds_{\boldsymbol{y}}d\tau, \tag{4.43}$$

so only local-in-time integrals are computed at each time step.

The above formulas hold when $t > \eta$. For $t \le \eta$, we initialize $S_{long}[\mu]$, $K_{long}[\mu]$, and $\hat{f}$ to be 0.

4.3.4 Computation of the single layer density

Based on the decomposition of the single layer potential and the approximation of the history and the local parts detailed previously, we can compute the density function μ.

For $t \le \eta$, substituting eq. (4.39) into eq. (4.27) and solving the integral equation, we can get the approximation to the density

$$\mu(\boldsymbol{x}_0,t) = \frac{2\mathcal{N}(\boldsymbol{x}_0,t,\mathbf{q})}{1-\sqrt{\frac{D_0 t}{\pi}}\xi(\boldsymbol{x}_0)} + O(t^{3/2}), \quad \boldsymbol{x}_0\in\partial\Omega, \quad t\le\eta. \tag{4.44}$$

For $t \in (\eta, \Delta-\delta]$, the integral equation eq. (4.27) can be rewritten as

$$\frac{1}{2}\mu(\boldsymbol{x}_0,t) + K_{short}[\mu](\boldsymbol{x}_0,t) = \beta(\boldsymbol{x}_0,t), \tag{4.45}$$

where the right-hand side is

$$\beta(\boldsymbol{x}_0, t) \equiv -K_{long}[\mu](\boldsymbol{x}_0, t) + \mathcal{N}(\boldsymbol{x}_0, t, \mathbf{q}). \tag{4.46}$$

We write the solution of the above integral equation as

$$\mu(\boldsymbol{x}_0, t) = 2\left(I + 2K_{short}\right)^{-1}[\beta]\left(\boldsymbol{x}_0, t\right), \quad \boldsymbol{x}_0 \in \partial\Omega, t \in (\eta, \Delta - \delta], \tag{4.47}$$

and expand the operator $(I + 2K_{short})^{-1}$ (corresponding to K_{short} being a contraction) as

$$\mu(\boldsymbol{x}_0, t) = 2\left(I - 2K_{short} + 4K_{short}^2 + \cdots + (-2)^n K_{short}^n + \ldots\right)[\beta]\left(\boldsymbol{x}_0, t\right). \tag{4.48}$$

We approximate $K_{short}^n[\beta]$ using eq. (4.36) and we get

$$K_{short}^n[\beta](\boldsymbol{x}_0, t) = \frac{1}{(-2)^n}\left(\frac{D_0\eta}{\pi}\right)^{n/2}\xi^n(\boldsymbol{x}_0)\beta(\boldsymbol{x}_0, t) + O(\eta^{3/2}). \tag{4.49}$$

Then, we keep all terms of the operator expansion to obtain

$$\begin{aligned}
\mu(\boldsymbol{x}_0, t) &= 2\left(\beta(\boldsymbol{x}_0, t) - 2K_{short}[\beta](\boldsymbol{x}_0, t) + 4K_{short}^2[\beta](\boldsymbol{x}_0, t) + \ldots\right) \\
&= 2\beta(\boldsymbol{x}_0, t)\left(1 + (\frac{D_0\eta}{\pi})^{\frac{1}{2}}\xi(\boldsymbol{x}_0) + \frac{D_0\eta}{\pi}\xi^2(\boldsymbol{x}_0) + \ldots\right) + O(\eta^{3/2}) \\
&= 2\beta(\boldsymbol{x}_0, t)/\left(1 - \sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{x}_0)\right) + O(\eta^{3/2}).
\end{aligned} \tag{4.50}$$

4.3.5 Computation of the single layer potential

Once the density μ is obtained, we compute the single layer potential $S[\mu]$ in the following way.

When $t \leq \eta$, the expression for the single layer potential is

$$\begin{aligned}
S[\mu](\boldsymbol{x}_0, t) &= S_{short}[\mu](\boldsymbol{x}_0, t) \\
&= \sqrt{\frac{D_0 t}{\pi}}\frac{4\pi\imath\rho\mathbf{q}\cdot\boldsymbol{n}e^{-4\pi^2 D_0\|\mathbf{q}\|^2 t}e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{x}_0}}{1 - \sqrt{\frac{D_0 t}{\pi}}\xi(\boldsymbol{x}_0)} + O(t^{3/2}), \quad \boldsymbol{x}_0 \in \partial\Omega, t \in [0, \eta].
\end{aligned} \tag{4.51}$$

For $t \in (\eta,\ \Delta - \delta]$, the single layer potential has both a local part and a history part. The local part is

$$S_{short}[\mu](\boldsymbol{x}_0, t) = \sqrt{\frac{D_0\eta}{\pi}}\mu(\boldsymbol{x}_0, t) + O(\eta^{3/2}), \quad \boldsymbol{x}_0 \in \partial\Omega, t \in (\eta,\ \Delta - \delta]. \tag{4.52}$$

As for the history part $S_{long}[\mu]$, it can be approximated by the truncated Fourier series:

$$S_{long}[\mu](\boldsymbol{x}_0,t) = D_0 \sum_{\boldsymbol{\nu}=-\boldsymbol{\nu}_{max}}^{\boldsymbol{\nu}_{max}} \hat{f}(\boldsymbol{\nu},t) e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0}\Delta\nu^2 + \mathcal{E}(\nu_{max}), \quad \boldsymbol{x}_0 \in \partial\Omega, t \in (\eta,\ \Delta-\delta]. \tag{4.53}$$

In the above, we denote the error term due to truncating the infinite Fourier series up to ν_{max} by $\mathcal{E}(\nu_{max})$. We do not have an analytical expression for $\mathcal{E}(\nu_{max})$, but we will show later in the numerical results that it decays exponentially in ν_{max}.

The addition of $S_{short}[\mu]$ and $S_{long}[\mu]$ gives the single layer potential $S[\mu]$ which is the solution of eq. (4.12) on the boundary:

$$S[\mu](\boldsymbol{x}_0,t) = S_{short}[\mu](\boldsymbol{x}_0,t) + S_{long}[\mu](\boldsymbol{x}_0,t), \quad \boldsymbol{x}_0 \in \partial\Omega, t \in [0,\ \Delta-\delta]. \tag{4.54}$$

At the current iteration step, the Fourier coefficients $\hat{f}$ that are still unknown will be computed using the density function μ from the previous iterations, as explained in the following.

Computation of the Fourier coefficients of the history part

For $t \leq \eta$, $\hat{f}$ is set to zero, as well as $K_{long}[\mu]$. For $t \in (\eta,\ 2\eta]$, $\hat{f}$ are computed using the density μ from the previous iterations:

$$\begin{aligned}\hat{f}(\boldsymbol{\nu},t) =& e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t}\hat{f}(\boldsymbol{\nu},t-\Delta t) + \\ & \underbrace{\int_{\partial\Omega}\int_{t-\eta-\Delta t}^{t-\eta} e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2(t-\tau)} e^{-2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{y}}\mu(\boldsymbol{y},\tau)d\tau ds_{\boldsymbol{y}}}_{\hat{f}_{temp1}(\boldsymbol{\nu},t)}, \quad \boldsymbol{\nu} \in [-\nu_{max},\nu_{max}]^2,\end{aligned} \tag{4.55}$$

with

$$\hat{f}_{temp1}(\boldsymbol{\nu},t) = \int_{\partial\Omega} 4\pi\imath\mathbf{q}\cdot\boldsymbol{n} e^{-2\pi\imath(\mathbf{q}+\boldsymbol{\nu})\cdot\boldsymbol{y}} \underbrace{\int_{t-\eta-\Delta t}^{t-\eta} \frac{e^{-4\pi^2 D_0\left[\|\boldsymbol{\nu}\|^2(t-\tau)+\|\mathbf{q}\|^2\tau\right]}}{1-\sqrt{\frac{D_0\tau}{\pi}}\xi(\boldsymbol{y})} d\tau}_{p} ds_{\boldsymbol{y}}. \tag{4.56}$$

We apply the trapezoidal rule to the time integration p to obtain

$$p = \begin{cases} -\frac{2\pi e^{-4\pi^2 D_0\|\mathbf{q}\|^2 t}}{D_0\xi^2(\boldsymbol{y})}\left[\xi(\boldsymbol{y})\sqrt{\frac{D_0}{\pi}}\left(\sqrt{t-\eta}-\sqrt{t-\eta-\Delta t}\right)+\right. \\ \qquad\qquad \left.\ln\left(\frac{1-\xi(\boldsymbol{y})\sqrt{\frac{D_0}{\pi}(t-\eta)}}{1-\xi(\boldsymbol{y})\sqrt{\frac{D_0}{\pi}(t-\eta-\Delta t)}}\right)\right], \|\boldsymbol{\nu}\| = \|\mathbf{q}\|; \\ e^{-4\pi^2 D_0\left[\|\mathbf{q}\|^2(t-\eta)+\|\boldsymbol{\nu}\|^2\eta\right]}\left[\frac{1+e^{4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t}(4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t-1)}{\Delta t(4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2))^2\left(1-\xi(\boldsymbol{y})\sqrt{\frac{D_0}{\pi}(t-\eta-\Delta t)}\right)}+\right. \\ \left.\frac{e^{4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t}-4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t-1}{\Delta t(4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2))^2\left(1-\xi(\boldsymbol{y})\sqrt{\frac{D_0}{\pi}(t-\eta)}\right)}\right], \|\boldsymbol{\nu}\| \neq \|\mathbf{q}\|. \end{cases} \tag{4.57}$$

Once we compute the time integration p, the integration over the boundary $\partial\Omega$ can be approximated by discretization in arc length.

Remaining on $t \in (\eta,\ 2\eta]$, next we compute the long time part $K_{long}[\mu]$ via the Fourier series

$$K_{long}[\mu](\boldsymbol{x}_0, t) = D_0 \sum_{\boldsymbol{\nu}=-\boldsymbol{\nu}_{max}}^{\boldsymbol{\nu}_{max}} 2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{n}\hat{f}(\boldsymbol{\nu}, t)e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0}\Delta\nu^2 + \mathcal{E}(\nu_{max}). \tag{4.58}$$

With a slight abuse of notation, we use the same notation $\mathcal{E}(\nu_{max})$ as in eq. (4.53) for the error due to truncating the Fourier series at ν_{max}.

Finally, the density function μ for (the current time) $t \in (\eta,\ 2\eta]$ is computed as:

$$\mu(\boldsymbol{x}_0, t) = \frac{2\left[\mathcal{N}(\boldsymbol{x}_0, t) - K_{long}[\mu](\boldsymbol{x}_0, t)\right]}{1-\sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{x}_0)} + O(\eta^{3/2}). \tag{4.59}$$

On the rest of the time interval, $t \in (2\eta,\ \Delta-\delta]$, $\hat{f}$ still uses the density μ from previous iterations, but the formulas are different:

$$\hat{f}(\boldsymbol{\nu}, t) = e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t}\hat{f}(\boldsymbol{\nu}, t-\Delta t) + \underbrace{\int_{\partial\Omega}\int_{t-\eta-\Delta t}^{t-\eta} e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2(t-\tau)}e^{-2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{y}}\mu(\boldsymbol{y},\tau)d\tau ds_{\boldsymbol{y}}}_{\hat{f}_{temp2}(\boldsymbol{\nu},t)}, \quad \boldsymbol{\nu}\in[-\nu_{max},\nu_{max}]^2. \tag{4.60}$$

In the above, the Fourier coefficients $\hat{f}(\boldsymbol{\nu}, t-\Delta t)$ at the previous time step are

known, and the expression of $\mu(\boldsymbol{x}_0,\tau)$ for $\tau\in(\eta,\Delta-\delta-\eta]$ is

$$\mu(\boldsymbol{x}_0,\tau)=2\left(1-\sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{x}_0)\right)^{-1}\left[\mathcal{N}(\boldsymbol{x}_0,\tau)-K_{long}[\mu](\boldsymbol{x}_0,\tau)\right],\quad \boldsymbol{x}_0\in\partial\Omega. \tag{4.61}$$

The integration on the right-hand side of eq. (4.60) is noted as $\hat{f}_{temp2}(\boldsymbol{\nu},t)$ in which we substitute the expression of μ above. We split $\hat{f}_{temp2}$ into two parts and gather the terms that are independent of time

$$\begin{aligned}\hat{f}_{temp2}(\boldsymbol{\nu},t)=\int_{\partial\Omega}2\left(1-\sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{y})\right)^{-1}e^{-2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{y}}\times\\ \big(2\pi\imath\mathbf{q}\cdot\boldsymbol{n}e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{y}}\underbrace{\int_{t-\eta-\Delta t}^{t-\eta}e^{-4\pi^2D_0(\|\mathbf{q}\|^2\tau+\|\boldsymbol{\nu}\|^2(t-\tau))}d\tau}_{h_1}-\\ \underbrace{\int_{t-\eta-\Delta t}^{t-\eta}K_{long}[\mu](\boldsymbol{y},\tau)e^{-4\pi^2D_0\|\boldsymbol{\nu}\|^2(t-\tau)}d\tau}_{h_2}\big)ds_{\boldsymbol{y}}.\end{aligned} \tag{4.62}$$

The time integration h_1 in the first part has an analytical expression

$$h_1=\begin{cases}\Delta t\cdot e^{-4\pi^2D_0\|\boldsymbol{\nu}\|^2t} & \|\mathbf{q}\|=\|\boldsymbol{\nu}\|\\ e^{-4\pi^2D_0\left[\|\mathbf{q}\|^2(t-\eta)+\|\boldsymbol{\nu}\|^2\eta\right]}\dfrac{e^{4\pi^2D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t}-1}{4\pi^2D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)} & \|\mathbf{q}\|\neq\|\boldsymbol{\nu}\|\end{cases} \tag{4.63}$$

The time integration h_2 in the second part has to be calculated numerically. We apply the trapezoidal rule to $K_{long}[\mu](\boldsymbol{y},\tau)$ and we get

$$h_2=\begin{cases}\frac{\Delta t}{2}\left[K_{long}[\mu](\boldsymbol{y},t-\eta)+K_{long}[\mu](\boldsymbol{y},t-\eta-\Delta t)\right] & \|\boldsymbol{\nu}\|=0\\ \Big[\frac{1-e^{-4\pi^2D_0\|\boldsymbol{\nu}\|^2\Delta t}(4\pi^2D_0\|\boldsymbol{\nu}\|^2\Delta t+1)}{(4\pi^2D_0\|\boldsymbol{\nu}\|^2)^2\Delta t}K_{long}[\mu](\boldsymbol{y},t-\eta-\Delta t)+\\ \frac{e^{-4\pi^2D_0\|\boldsymbol{\nu}\|^2\Delta t}+4\pi^2D_0\|\boldsymbol{\nu}\|^2\Delta t-1}{(4\pi^2D_0\|\boldsymbol{\nu}\|^2)^2\Delta t}K_{long}[\mu](\boldsymbol{y},t-\eta)\Big]e^{-4\pi^2D_0\|\boldsymbol{\nu}\|^2\eta} & \|\boldsymbol{\nu}\|\neq0\end{cases} \tag{4.64}$$

The values of $K_{long}[\mu]$ at time $t-\eta-\Delta t$ and $t-\eta$ have been computed in previous steps, thus the expressions for h_1 and h_2 can be computed in the current time step. Then we discretize in the arc length over the boundary to obtain $\hat{f}_{temp2}$ as well as $\hat{f}$.

Staying on $t\in(2\eta,\ \Delta-\delta]$, it is straightforward to recover the long time part $K_{long}[\mu]$ at time t by applying the inverse discrete Fourier transform

$$K_{long}[\mu](\boldsymbol{x}_0,t)=D_0\sum_{\boldsymbol{\nu}=-\boldsymbol{\nu}_{max}}^{\boldsymbol{\nu}_{max}}2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{n}\hat{f}(\boldsymbol{\nu},t)e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0}\Delta\nu^2+\mathcal{E}(\nu_{max}). \tag{4.65}$$

Again, with a slight abuse of notation, we use the same notation $\mathcal{E}(\nu_{max})$ as in eq. (4.53) for the error due to truncating the Fourier series at ν_{max}. Finally, the density function μ at the current time t is

$$\mu(\boldsymbol{x}_0, t) = \frac{2\left[\mathcal{N}(\boldsymbol{x}_0, t) - K_{long}[\mu](\boldsymbol{x}_0, t)\right]}{1 - \sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{x}_0)} + O(\eta^{3/2}), \tag{4.66}$$

which will be used for future iterations.

4.3.6 Computation of the diffusion MRI signal

After obtaining the single layer potential, the following procedure produces the diffusion MRI signal attenuation.

The signal attenuation E has the representation

$$E = \rho e^{-4\pi^2 D_0 \|\mathbf{q}\|^2(\Delta-\delta)} + \frac{1}{|\Omega|}\overline{\omega}(\mathbf{q}, \Delta - \delta). \tag{4.67}$$

The quantity $\overline{\omega}$ will be computed using the recursive relationship below (rewritten from eq. (4.21)):

$$\begin{aligned}\overline{\omega}(\mathbf{q}, t) &= e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t}\overline{\omega}(\mathbf{q}, t - \Delta t) \\ &\quad - D_0 \int_{\partial\Omega} 2\pi\imath\mathbf{q}\cdot\boldsymbol{n} e^{2\pi\imath\mathbf{q}\cdot\boldsymbol{y}} \overbrace{\int_{t-\Delta t}^{t} e^{-4\pi^2 D_0\|\mathbf{q}\|^2(t-\tau)}\omega(\boldsymbol{y}, \tau)d\tau}^{u}\, ds_{\boldsymbol{y}}.\end{aligned} \tag{4.68}$$

By applying the trapezoidal rule to the time integration u, we then get the expression

$$u = \begin{cases} \frac{\Delta t}{2}\left[\omega(\boldsymbol{y}, t - \Delta t) + \omega(\boldsymbol{y}, t)\right] & \|\mathbf{q}\| = 0 \\ \frac{1 - e^{-4\pi^2 D_0\|\mathbf{q}\|^2\Delta t}(4\pi^2 D_0\|\mathbf{q}\|^2\Delta t + 1)}{(4\pi^2 D_0\|\mathbf{q}\|^2)^2\Delta t}\omega(\boldsymbol{y}, t - \Delta t) & \\ + \frac{e^{-4\pi^2 D_0\|\mathbf{q}\|^2\Delta t} + 4\pi^2 D_0\|\mathbf{q}\|^2\Delta t - 1}{(4\pi^2 D_0\|\mathbf{q}\|^2)^2\Delta t}\omega(\boldsymbol{y}, t) & \|\mathbf{q}\| \neq 0 \end{cases} \tag{4.69}$$

The variable ω is the single layer potential $S[\mu]$

$$\omega(\boldsymbol{x}_0, t) = S[\mu](\boldsymbol{x}_0, t) = S_{short}[\mu](\boldsymbol{x}_0, t) + S_{long}[\mu](\boldsymbol{x}_0, t). \tag{4.70}$$

The short time part has an asymptotic expression

$$S_{short}[\mu](\boldsymbol{x}_0, t) = 2\sqrt{\frac{D_0\eta}{\pi}}\frac{\mathcal{N}(\boldsymbol{x}_0, t) - K_{long}[\mu](\boldsymbol{x}_0, t)}{1 - \sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{x}_0)} + O(\eta^{3/2}) \tag{4.71}$$

with

$$\mathcal{N}(\boldsymbol{x}_0, t) = 2\pi\imath\rho\mathbf{q}\cdot\boldsymbol{n}e^{-4\pi^2 D_0\|\mathbf{q}\|^2 t}e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{x}_0}, \tag{4.72}$$

and

$$K_{long}[\mu](\boldsymbol{x}_0, t) = D_0 \sum_{\boldsymbol{\nu}=-\boldsymbol{\nu}_{max}}^{\boldsymbol{\nu}_{max}} 2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{n}\hat{f}(\boldsymbol{\nu}, t)e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0}\Delta\nu^2 + \mathcal{E}(\nu_{max}). \tag{4.73}$$

The long time part is approximated by a Fourier series

$$S_{long}[\mu](\boldsymbol{x}_0, t) = D_0 \sum_{\boldsymbol{\nu}=-\boldsymbol{\nu}_{max}}^{\boldsymbol{\nu}_{max}} \hat{f}(\boldsymbol{\nu}, t)e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0}\Delta\nu^2 + \mathcal{E}(\nu_{max}). \tag{4.74}$$

Finally, a boundary discretization allows the numerical computation of $\overline{\omega}$.

4.4 Numerical results

In this section, we study the convergence of the Fourier Potential method. The matrix formalism method [47, 48] is capable of computing analytical signals for simple geometries, such as circles and spheres, using analytical expressions for the Laplace eigendecomposition, so we use the MF signals as the reference signals. We note the diffusion MRI signal simulated by our method as s and the analytical signal given by the MF as s_{ref}.

The geometry on which we will conduct the convergence study is a circle of radius r ($\phi = 2r, \xi = 1/r$), where ϕ is the size of the geometry and ξ is the curvature. The default values for the physical parameters are below:

- $r = \{1, 2, 4\}\ \mu m$, $\xi = \{1, 0.5, 0.25\}\ \mu m^{-1}$
- $\mathcal{D}_0 = 2 \times 10^{-3}\ \mu m^2/\mu s$
- $\delta = 10^{-3}\ \mu s$, $\Delta = 5,000\ \mu s$
- $u_{\mathbf{g}} = [1, 0]^T$
- $b = \{1000, 4000\}\ \mu s/\mu m^2$, or equivalently $\{1, 4\}\ ms/\mu m^2$

We will study the dependence of the relative error (not in percent), defined by

$$\varepsilon = \left|\frac{s - s_{ref}}{s_{ref}}\right|, \tag{4.75}$$

on the discretization parameters: spatial step Δx, the time step Δt, the maximum frequency ν_{max}, the spectral step $\Delta\nu$, and the single layer local time interval η. As the convergence studies for various algorithm parameters are conducted, the default values for the fixed parameters are listed below:

- $\eta = 1\ \mu s$
- $\nu_{max} = 10\ \mu m^{-1}$, $\Delta\nu = 0.05\ \mu m^{-1}$
- $\Delta x = 0.005\ \mu m$, $\Delta t = 0.5\ \mu s$

4.4.1 The narrow pulse assumption error

One important point to discuss here, before showing the convergence studies, is the choice of the duration of the diffusion-encoding gradient pulse, δ. We need that $\delta \ll \Delta$ to satisfy the narrow pulse assumption. In fig. 4.1, we show the error due to the narrow pulse assumption for a range of δ values. At $b = 1000\ \mu s/\mu m^2$, the narrow pulse approximation error is around 10^{-2} at $\delta = 10^2\ \mu s$ for all three circle radii.

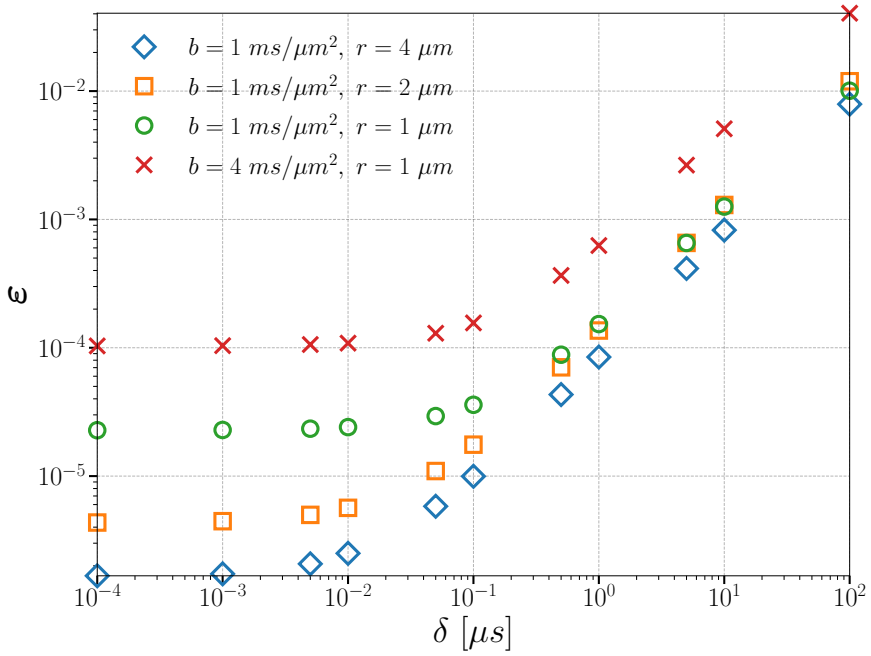


Figure 4.1: Influence of δ on the relative error. All discretization parameters are set to be the default. The sampled δ's are {0.0001, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10, 100} μs (from left to right).

The validity of the narrow pulse assumption also depends on the separation between the two pulses Δ. The narrow pulse assumption requires a small ratio of δ and Δ. In fig. 4.2, we show the influence of this ratio on the relative error. For large Δ such as $20\ ms$, the relative error is less than 5% with δ being $2\ ms$.

Despite the fact that a relative error of a few percent is perfectly acceptable for diffusion MRI applications, for the sake of the numerical convergence study

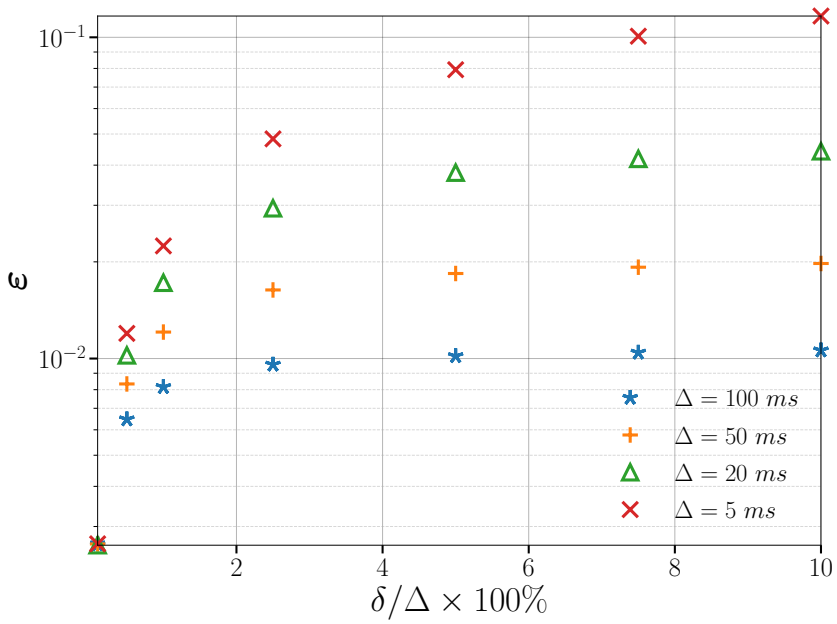


Figure 4.2: Influence of δ/Δ on the relative error. All discretization parameters are set to be the default. The geometry is a circle of radius $1\ \mu m$ and the b-value is $4\ ms/\mu m^2$. The sampled ratios δ/Δ are {0.1%, 0.5%, 1%, 2.5%, 5%, 7.5%, 10%} (from left to right).

that follows, we have chosen much lower thresholds for the narrow pulse approximation error and picked an exceedingly small value of $\delta = 10^{-3}\ \mu s$, which is not achievable with current MRI scanners. This choice is because we wanted the error from the narrow pulse assumption to be significantly smaller than the discretization errors of the numerical method as we refined the method parameters. In this way, the plateauing of the errors towards the narrow pulse approximation error occurs later in the refinement process so that we can verify if the error behavior follows the error analysis presented in section 4.3. We note that at our choice of $\delta = 10^{-3}\ \mu s$, the narrow pulse approximation errors shown in fig. 4.1 range from 10^{-6} ($b = 1000\ \mu s/\mu m^2$, $r = 4\ \mu m$) to 10^{-4} (higher b-values). These values will form the "floor" values for our convergence curves, to be shown next.

4.4.2 Duration of the local in time part of the single layer potential, η

First, we study the duration η of the local in time part of the single layer potential. The error term $O(\eta^{3/2})$ originates from the asymptotic trace formulas eq. (4.35) and eq. (4.36). In fig. 4.3 the curves show a clear convergence order

of $3/2$ in η. In addition, the circle radius (curvature) and the b-value affect the errors: the errors are bigger for larger b-value and higher curvature.

We observe that minimum errors occur at $\eta = 1\ \mu s$. The values of the minimum errors coincide with the size of the narrow pulse approximation errors shown in fig. 4.1. At the smaller value of $\eta = 0.5\ \mu s$, the errors increased. The reason is that there is a tradeoff between two sources of error, one linked to $O(\eta^{3/2})$ and one to $\mathcal{E}(\nu_{max})$. With a smaller η, the long time part $S_{long}[\mu]$ suffers more from the singularity of the heat kernel, thereby increasing the error $\mathcal{E}(\nu_{max})$. After we decrease η beyond a certain point, $\mathcal{E}(\nu_{max})$ becomes the bottleneck for the accuracy, which will be studied next.

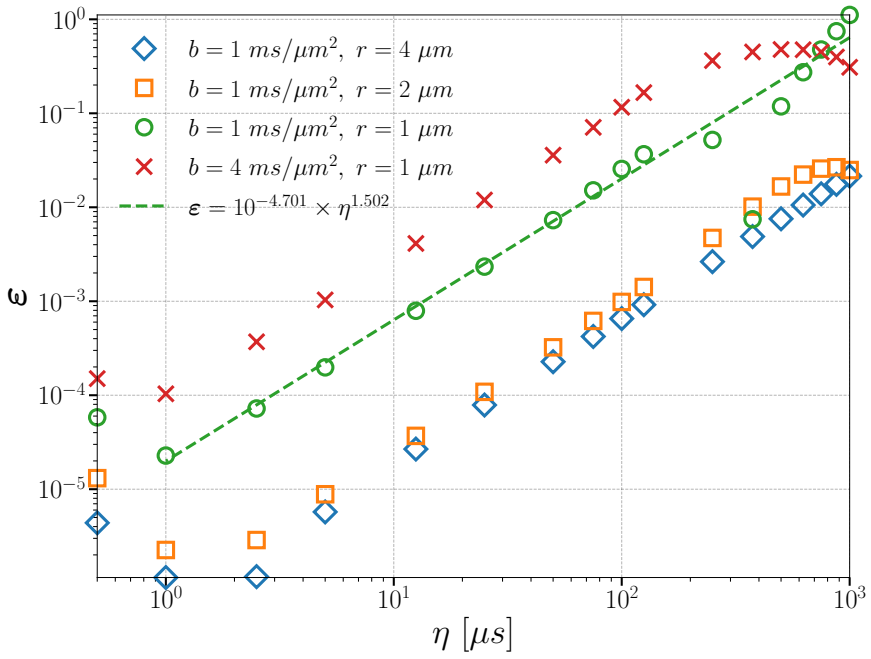


Figure 4.3: Convergence curves regarding η. All discretization parameters except for η are set to be the default. The sampled η's are {0.5, 1, 2.5, 5, 12.5, 25, 50, 75, 100, 125, 250, 375, 500, 625, 750, 875, 1000} μs (from left to right). The slopes of the curves are around $3/2$.

4.4.3 Maximum frequency

The main feature of our method is that the history part of the single layer potential $S_{long}[\mu]$ has a spectral representation. The spectrum of the fundamental solution G decreases exponentially with respect to the frequency $\boldsymbol{\nu}$. As a result, the Fourier coefficients $\hat{f}$ are also subject to the exponential decay:

$$\hat{f}(\boldsymbol{\nu}, t) = O(e^{-4\pi^2 D_0 \eta \|\boldsymbol{\nu}\|^2}). \tag{4.76}$$

In order to numerically compute the spectrum of $S_{long}[\mu]$, we truncated it at ν_{max} and omitted all higher frequency components. The truncation gives rise to an error caused by the omitted Fourier modes, which we have denoted as $\mathcal{E}(\nu_{max})$. Even though we do not have an analytical expression for $\mathcal{E}(\nu_{max})$, considering the exponential decay of the Fourier coefficients, we could expect a rapid decrease in the truncation error.

We present the convergence curves in fig. 4.4. We note that the x-axis is linear, and the y-axis is logarithmic. Empirically, we observe that the error can be fitted by $c_1 e^{-c_2 \nu_{max}}$, where c_1 is a constant, and $c_2 = 1.17$. As expected, for the largest ν_{max}, ($\nu_{max} > 9\ \mu m^{-1}$), the curves approach the errors due to the narrow pulse approximation.

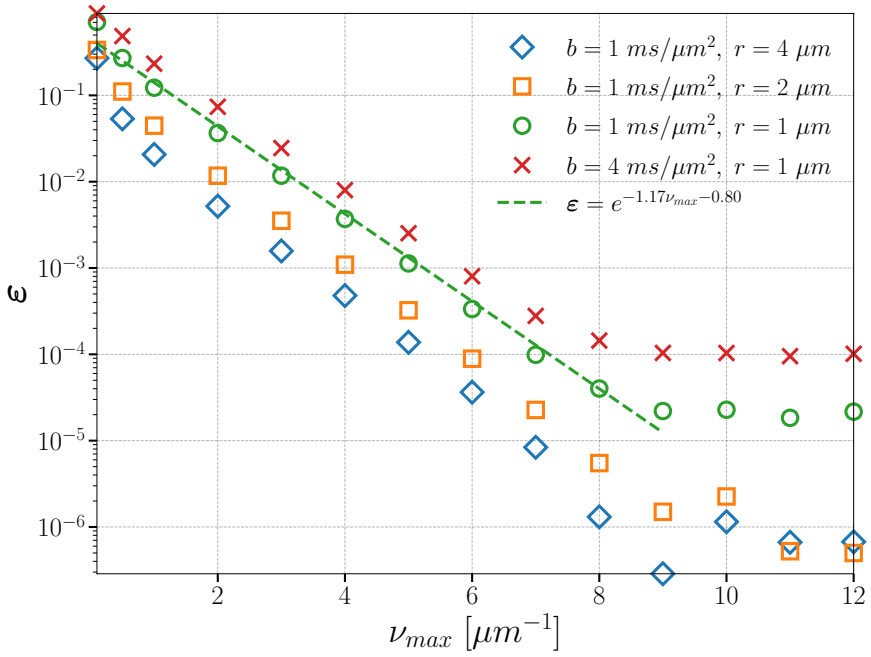


Figure 4.4: Convergence curves regarding ν_{max}. All discretization parameters except for ν_{max} are set to be the default. The sampled ν_{max}'s are {0.1, 0.5, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12} μm^{-1} (from left to right).

4.4.4 Spatial discretization

Our method contains several boundary integrations. The geometries we used are circles, and we chose to have a piecewise linear approximation of $\partial\Omega$. This means the discretized geometries are regular polygons. Let us call the discretized segment length of the boundary Δx. On the other hand, the reference solution (MF) computes the Laplace eigenfunctions of exact circles.

Figure 4.5 illustrates the convergence curves in Δx. At the larger range of Δx, we observe exponential convergence in $\frac{1}{\Delta x}$, due to the exponential convergence of the trapezoidal rule for periodic functions (the integrand over a closed boundary being a periodic function). At the smaller range of Δx, we observe the plateauing towards the narrow pulse approximation errors.

In the middle range of Δx, we observe a convergence of $O(\Delta x^2)$, due to the approximation of the exact circle geometry by regular polygons. To better visualize the convergence pattern, we plot the approximation error for the area of an exact circle by regular polygons. The area error e is defined as the normalized difference between the circle area and the area of a regular n-sided inscribed polygon $\mathcal{A}_n$

$$e = \frac{\pi r^2 - \mathcal{A}_n}{\pi r^2} \sim \frac{\Delta x^2}{6r^2} = \frac{(\xi \Delta x)^2}{6}. \tag{4.77}$$

This explains the convergence order of Δx^2. Moreover, eq. (4.77) also indicates the influence of curvature. High curvature geometries endure greater area errors, thus, larger simulation errors.

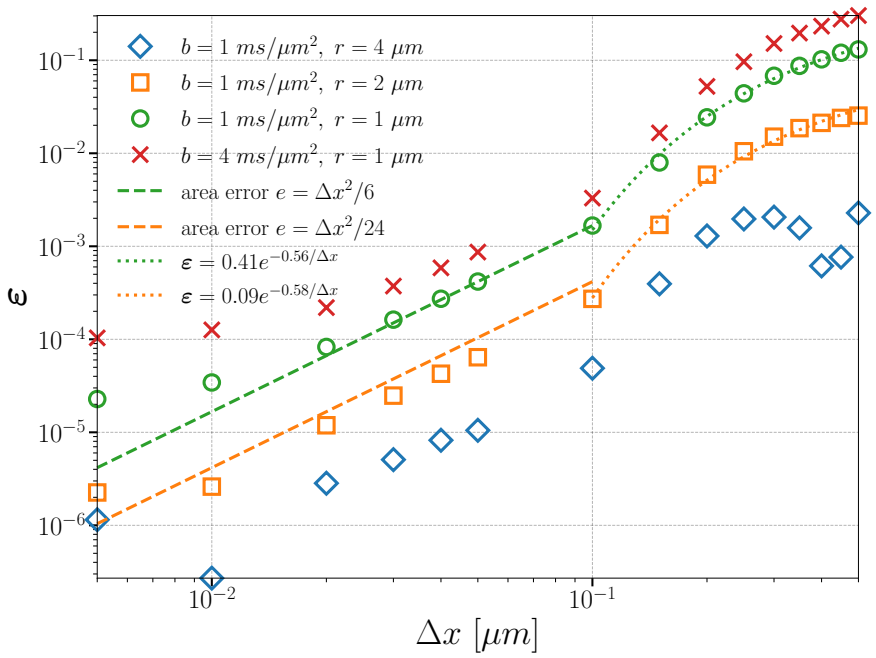


Figure 4.5: Convergence curves regarding Δx. All discretization parameters except for Δx are set to be the default. The sampled Δx's are {0.005, 0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5} μm (from left to right).

4.4.5 Temporal discretization

Let the time step be Δt. We apply the trapezoidal rule to every time integration in our implementation, for instance, eqs. (4.55), (4.64) and (4.69). Theoretically, the trapezoidal integration error is $O(\Delta t^2)$ [207]. However, the local-in-time region size, η, which is an integer multiple of Δt, contributes an error from the asymptotic formula, as shown in fig. 4.3. This asymptotic formula error numerically dominates the $O(\Delta t^2)$ error from the trapezoidal integration. Thus, we do not show a plot of the trapezoidal rule convergence.

4.4.6 Spectral discretization

The spectral resolution $\Delta\nu$ is closely related to the size of the periodic box enclosing the geometry. Let the side length of the box be L. According to the Nyquist–Shannon sampling theorem, we should have

$$\frac{1}{\Delta\nu} = L. \tag{4.78}$$

The inverse relationship manifests itself in eqs. (4.23) and (4.24) as well. A necessary restriction on the box is that it must contain the entire domain Ω, in our case, the domain being a circle, we get

$$\frac{1}{\Delta\nu} = L \geq \phi, \tag{4.79}$$

where we defined ϕ as twice the radius. In fig. 4.6, it is shown that the relative errors are greater than 100% when the box is smaller than the domain ($\frac{1}{\Delta\nu} < \phi$). As soon as the box contains the geometry, the errors reduce to the plateau values of the narrow approximation errors. Clearly, all simulations must satisfy the spectral discretization condition eq. (4.79).

4.4.7 Influence of q-vector

Now we study the influence of the b-value/q-vector on the relative errors. We fix the diffusion time δ and Δ, so the b-value is equivalent to the square of the magnitude of the q-vector. We chose to plot the relative error versus the magnitude of the q-vector because we explicitly formulated our method using q-vectors rather than b-values. The results are given in fig. 4.7. We note that the x-axis is logarithmic, and the y-axis is linear. The experiment results show that once the magnitude of the q-vector is large enough, the error increases logarithmically with the norm of q-vectors. For small q-vectors, the errors are within the range of the error floor ($10^{-4} - 10^{-6}$) imposed by the narrow pulse approximation. For larger q, the logarithmic dependence of the error on $\|\mathbf{q}\|$ requires further study to explain. We do not at this time have an explication for it.

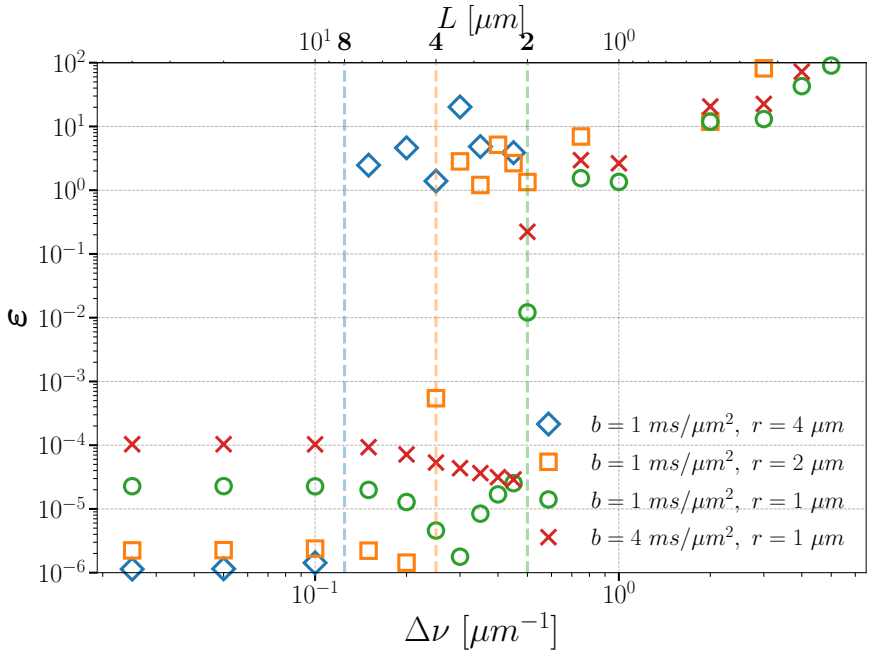


Figure 4.6: Convergence curves regarding $\Delta\nu$. Relative errors which are greater than 100 (2dB) are omitted for a better visualization. All discretization parameters except for $\Delta\nu$ are set to be the default. The sampled $\Delta\nu$'s are {0.025, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.75, 1, 2, 3, 4, 5} μm^{-1} (from left to right).

4.4.8 Extension to complex geometries

In the previous sections, we used circles to study the convergence of our method. The Matrix Formalism method can compute the analytical solution on circles, which allows us to show the convergence behavior of the Fourier Potential Method.

Our method can simulate diffusion MRI signals on more complex 2D geometries. Here we present FPM simulation results on two realistic axons. The microscopy image (fig. 4.8) and the axon sections are obtained using the AxonDeepSeg segmentation framework [208]. With these irregular shapes, analytical solutions are not accessible, so we computed the reference signals by finite element simulations using the SpinDoctor toolbox [44]. We show, in fig. 4.9, the dMRI signals in 40 directions as well as the relative errors. Our method agrees with the finite element reference signals. For the middle b-value ($4000\ \mu s/\mu m^2$), the relative error is less than 5%. One should note that the magnetization of the two adjacent axons is computed simultaneously by sharing the same Fourier basis. This feature is different from the matrix formalism method, which requires geometry-dependent bases.

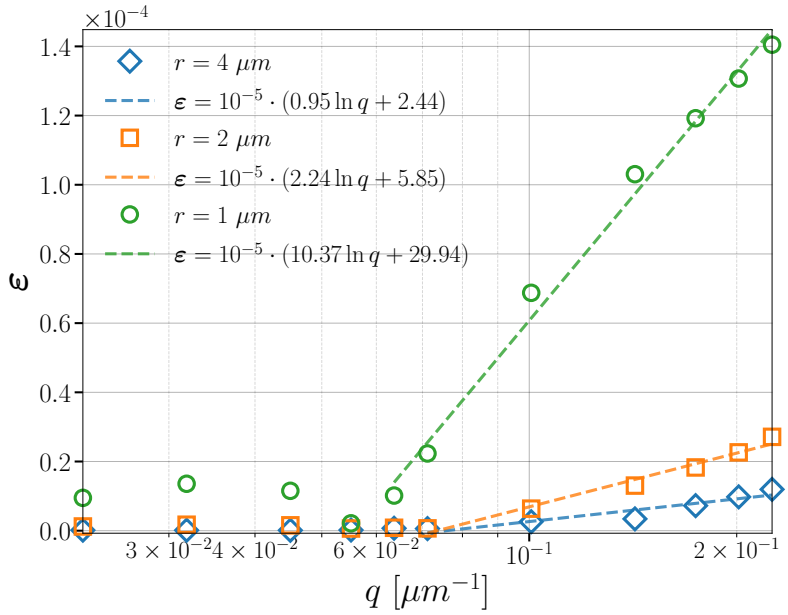


Figure 4.7: Influence of q on the relative error. All discretization parameters are set to be the default. The sampled q's are {0.0225, 0.0318, 0.0450, 0.0551, 0.0637, 0.0712, 0.1007, 0.1424, 0.1743, 0.2013, 0.2251} μm^{-1} and the corresponding b-values are {100, 200, 400, 600, 800, 1000, 2000, 4000, 6000, 8000, 10,000} $\mu s/\mu m^2$ (from left to right).

Figure 4.8: The microscopy image of axons from AxonDeepSeg. Two adjacent axons are selected.

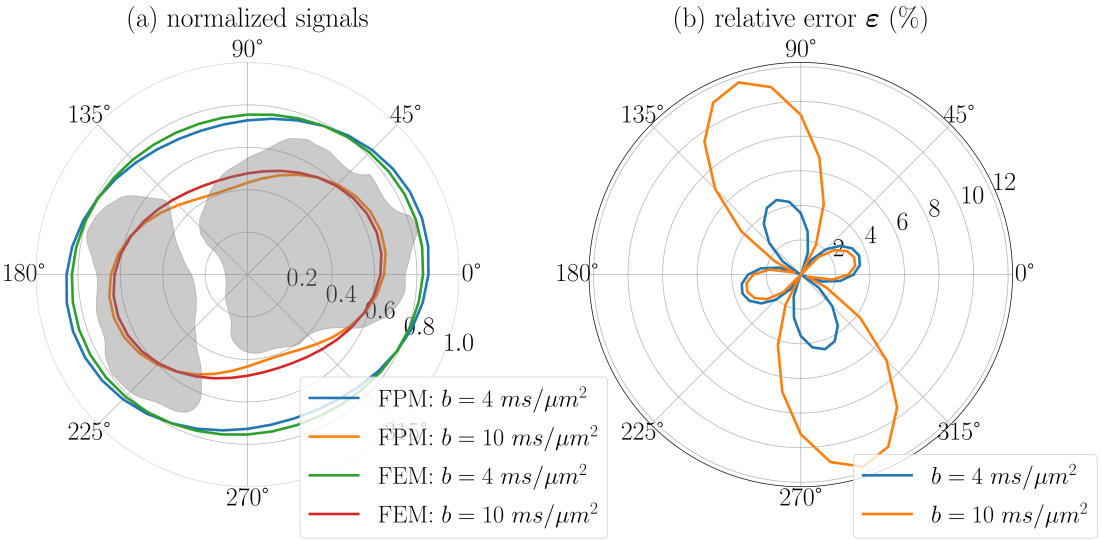


Figure 4.9: Comparison of FPM with FEM. **(a)** the normalized signals simulated by FPM and FEM. The gray areas illustrate the shapes of the two adjacent axons. The physical parameters are: $\mathcal{D}_0 = 2 \times 10^{-3}\ \mu m^2/\mu s$, $\delta = 2\ ms$, $\Delta = 100\ ms$. The discretization parameters of the FPM are: $\eta = 50\ \mu s$, $\nu_{max} = 2\ \mu m^{-1}$, $\Delta\nu = 0.05\ \mu m^{-1}$, $\Delta x = 0.01\ \mu m$, and $\Delta t = 50\ \mu s$. The signals are simulated in 40 directions evenly distributed on a unit circle. **(b)** the relative errors in percent.

4.5 Summary

In this chapter, we derived a new representation of the diffusion MRI signal by solving the BT equation using potential theory. The decomposition of the single layer potential into singular and smooth parts allows the numerically efficient storage of the smooth part on a Fourier basis. Time integrals in the form of certain exponentials allow us to use time recursion to avoid history dependence. We numerically validated the convergence of our method and showed the error behavior in several simulation parameters.

One of the main features of our method is the availability of the spectrum of the smooth part of the magnetization field. The projection to the Fourier basis functions provides a unified spectrum space for different geometries. Since our method provides a Fourier-like representation of the diffusion MRI signal, this can potentially facilitate new physical and biological signal interpretation in the future.

A mixed basis approach developed by Nordin et al. [209, 210] resembles our method. They utilize a set of basis functions that consists of the Fourier functions and the surface functions (dipole potentials) to capture the most relevant part of the low-frequency spectrum of the Laplace operator in a confined geometry [209, 210]. In the mixed basis approach, the role of the Fourier functions is to mimic the free diffusion behavior. The influence of the boundaries is captured by the surface functions. However, in our method, the free diffusion part is represented by the first exponential term in eq. (4.16). The Fourier functions are used to capture the history part of the influence of the boundaries. Consequently, the spectrum spaces of the two methods are different. In addition, the mixed basis approach is more general in the sense that it doesn't require the narrow pulse assumption.

Our method is currently of theoretical interest only. It only solves two-dimensional problems and it is computationally intensive. As the first work addressing this subject, we restricted ourselves to the 2D diffusion MRI setting with impermeable interfaces. To extend FPM to 3 dimensions, the main changes to be made are the asymptotic trace formulas for the local part, i.e., eq. (4.35) - eq. (4.39), in particular, the curvature for 1D curves will need to be generalized to analogous quantities on 2D surfaces. The generalization of the curvature and the derivation of the asymptotic trace formulas are the major difficulties for the extension to three dimensions. As a consequence, the solution of the integral equation (eq. (4.50)) will have a new formulation in 3D. Another change involves spatial integration on 2D surfaces instead of on 1D curves, the former being more numerically complicated than the latter.

We also restrict ourselves to simplified conditions on the diffusion-encoding

gradient and permeability. Specifically, we derive our method under the narrow pulse assumption and impermeability assumption. The question of how to remove these assumptions remains to be studied.

Chapter 5

Simulation-Based Brain Microstructure Imaging

This chapter presents the fourth contribution of this thesis, a framework for training supervised learning models on synthetic data to estimate brain microstructure using diffusion MRI non-invasively. The framework relies on the NeuronSet we built in chapter 2 and the numerical matrix formalism optimized in section 3.3. Over 1,000 neuron meshes converted from digital neuronal reconstructions archived in NeuroMorpho.Org allow us to measure neuroanatomical parameters and simulate intracellular dMRI signals by solving the Bloch-Torrey partial differential equation. Moreover, we randomly combine neuron meshes with extracellular compartments to obtain a synthetic dataset comprising both the dMRI signals and more than 40 microstructure parameters of over 1.4 million artificial brain voxels. Unlike existing biophysical models, our approach achieves higher modeling accuracy while requiring fewer assumptions. The synthetic dataset is valuable for validating biophysical models and approximating the mappings from dMRI signals to microstructure parameters. We demonstrate exemplary multilayer perceptrons (MLPs) trained on the synthetic dataset for volume and area fraction estimation. They perform satisfactorily in synthetic test sets and give promising in vivo parameter maps using the MGH CDMD dataset. Most importantly, the in vivo volume fraction estimation depends less on the diffusion time, which is one of the desired properties of quantitative microstructure imaging.

5.1 Introduction

Brain microstructure imaging often relies on "inverting" a forward model describing the dMRI signal generation mechanism, as explained in section 1.3. Therefore, the accuracy of the forward model is of essential importance. The predominant forward models, i.e., biophysical models, typically subdivide a brain voxel into compartments described by simplified geometries such as cylinders with zero radii (sticks) [81, 211] and spheres (balls) [88]. Together with some additional assumptions, such as the Gaussian phase assumption (GPA), a biophysical model allows deriving an analytical signal expression as a function of the model parameters related to several microstructure parameters [69].

One often fits the signal expression to experimental data to estimate the model parameters. However, the indeterminacy inherent in some biophysical models makes the parameter estimation unstable [212]. Moreover, an accurate fit does not necessarily justify the underlying biophysical model, and the estimated model parameters might be biophysically meaningless [68, 94]. Subtle effects like neurite undulation are excluded from biophysical models because of mathematical complications [81, 88, 188]. In addition to the error brought by the simplified geometric models, the validity of some assumptions, such as GPA, remains undetermined [68, 99]. Besides, the validity regimes of several signal expressions depend on the length scales of underlying microstructure [91]. A voxel may exhibit multiple length scales (e.g., various soma radii) so that different validity regimes may co-exist or emerge progressively [94], making comprehensive model validation difficult.

To address the above shortcomings and achieve a more accurate forward model, we aim to replace the simplified geometries with realistic neuron meshes and the analytical intracellular signal expressions with diffusion MRI simulations. The numerical dMRI simulation methods, including both algorithms based on solving the Bloch-Torrey partial differential equation (BT equation) [43–46, 51, 52, 58] and Monte-Carlo methods [36–38, 60, 62–65], are gold-standard forward models for describing the formation of dMRI signals [33]. With realistic neuron meshes, numerical simulation can seamlessly incorporate effects arising from, for instance, neurite undulation or water exchange between soma and neurites.

The proposed framework relies on an ultra-fast dMRI simulator, a neuron mesh dataset, and machine learning (ML) techniques to estimate brain microstructure properties by leveraging dMRI simulation as the forward model. The simulations are conducted on neuron meshes to get the intracellular dMRI signals which allow the approximation of the signal attenuations from artificial brain voxels whose microstructure properties are computed from neuroanatomical parameters measured on neuron meshes. Artificial brain voxels'

signal attenuations and microstructure parameters form a synthetic dataset for training ML models. Figure 5.1 shows an overview of the framework.

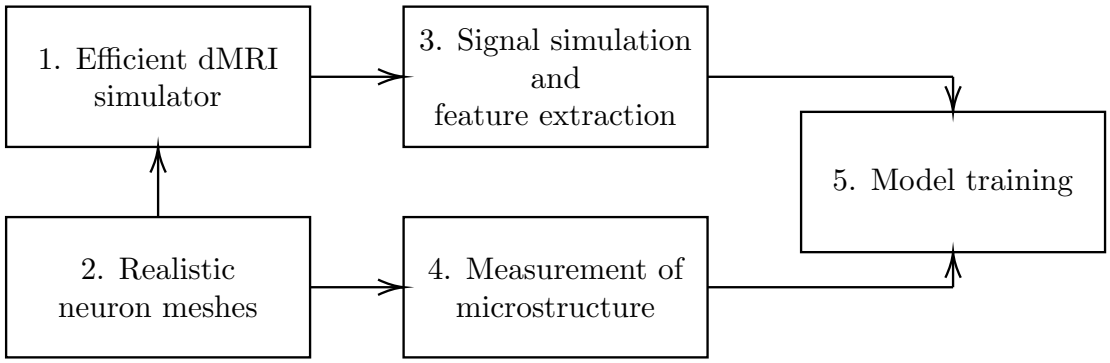


Figure 5.1: Overview of the simulation-driven supervised learning framework. The ultra-fast dMRI simulator and a large set of realistic neuron meshes are two cornerstones of the framework. The simulations are conducted on neuron meshes to get the intracellular dMRI signals which allow the approximation of the signal attenuations from artificial brain voxels whose microstructure parameters are computed from neuroanatomical measurements. Artificial brain voxels' signal attenuations and microstructure parameters form a synthetic dataset. Finally, we train machine learning models on the dataset.

To "invert" the gold-standard forward model, we leverage machine learning techniques. The adoption of ML models in dMRI dates back to the last century [213] and has seen a recent resurgence [104, 214–217]. Artificial neural networks are believed to be superior in function approximation [218–220], especially in high dimensions [221, 222]. This chapter leverages MLPs to approximate the underlying mappings from signals to microstructure parameters.

Specifically, we first generate a synthetic dataset containing both the dMRI signals and more than 40 microstructure parameters of over 1.4 million artificial brain voxels. Figure 5.2 summarizes the data-generating process and provides more details about the blocks 3-5 of fig. 5.1. The dMRI signals and the microstructure parameters of an artificial brain voxel form an entry in the synthetic dataset.

MLPs are trained in the dataset in a supervised way. We demonstrate several exemplary MLPs for volume and area fraction estimation in synthetic test sets and the MGH CDMD dataset [22]. Finally, the MLPs are compared with the state-of-the-art impermeable biophysical model, SANDI [88].

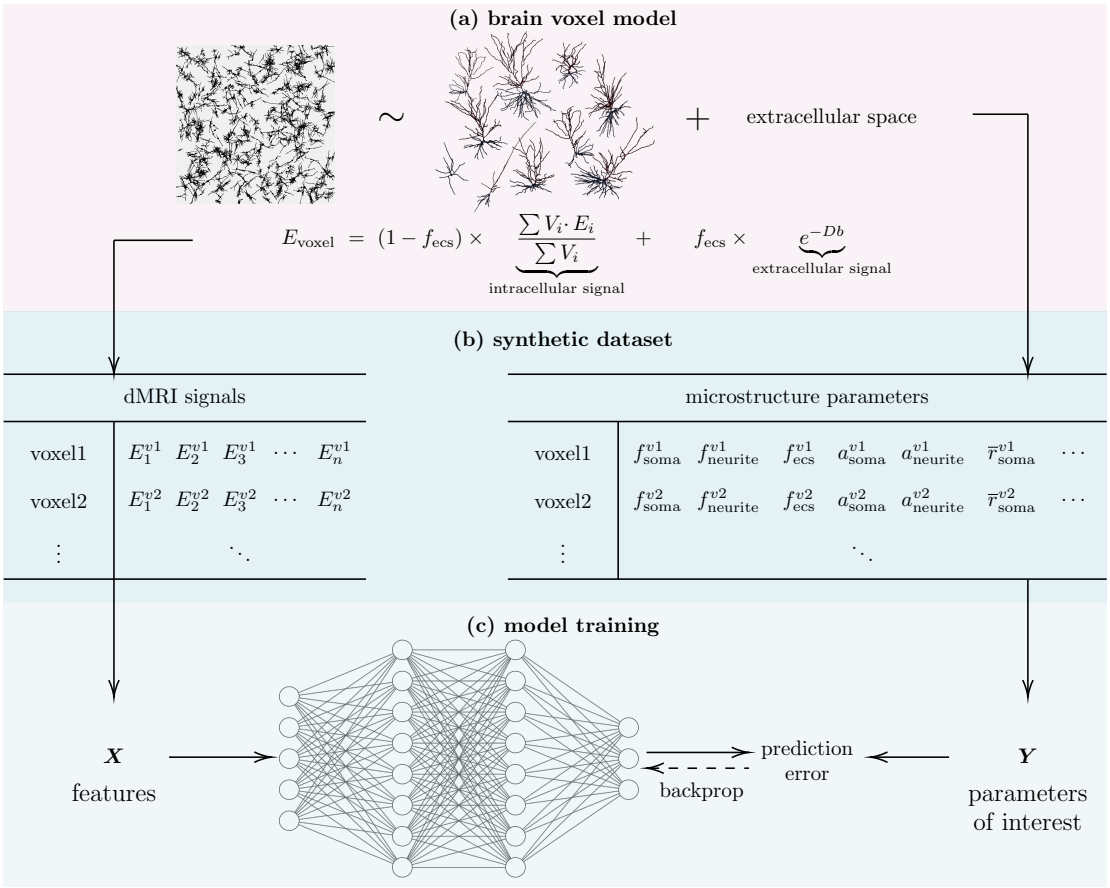


Figure 5.2: Summary of the synthetic dataset generation and model training based on the simplified brain voxel model. The intracellular signal is the volume-weighted signal of numerous randomly picked neurons. The ECS is modeled as a free diffusion space. Our synthetic dataset consists of the dMRI signals and microstructure parameters of 1.4 million artificial brain voxels. Features like direction-averaged signals are computed from the dMRI signals to predict some microstructure parameters. Finally, we train an MLP to fit the mapping from the features to the parameters of interest. The picture of the MLP is for illustration purposes only. The actual structure is different from the one shown in the figure.

5.2 Experimental data

Before we dive into the synthetic dataset generation, let us introduce the experimental data used in this chapter. MGH CDMD [22] is an open-access diffusion MRI dataset providing processed in vivo human brain scans for 26 healthy subjects, seven of which are scanned twice. The diffusion MRI data were acquired on the 3T Connectome MRI scanner (Magnetom CONNECTOM, Siemens Healthineers), and a 64-channel phased array head coil [223] was used for signal reception. The maximum slew rate is 62.5 mT/m/ms. The diffusion encoding sequence is PGSE (see fig. 3.2(a) and eq. (3.6)), whose parameters are:

- the pulse duration $\delta = 8\ ms$, two inter-pulse durations $\Delta = 19,\ 49\ ms$;
- eight non-zero gradient intensities (31, 68, 105, 142, 179, 216, 253, 290 mT/m) corresponding to eight b-values (72, 346, 825, 1509, 2400, 3491, 4789, 6292 $\mu s/\mu m^2$) at the short diffusion time $\delta/\Delta = 8/19\ ms$, and eight b-values (204, 981, 2340, 4279, 6800, 9902, 13,584, 17,848 $\mu s/\mu m^2$) at the long diffusion time $\delta/\Delta = 8/49\ ms$;
- one interspersed image without diffusion-encoding gradient ($g = 0$) for every 16 diffusion-weighted images;
- 32 diffusion encoding directions uniformly distributed on a sphere for $b < 2400\ \mu s/\mu m^2$ and 64 uniform directions for $b \geq 2400\ \mu s/\mu m^2$.

Other imaging parameters are as follows: the echo time $TE = 77\ ms$, repetition time $TR = 3800\ ms$, field of view (FOV) = $216 \times 216\ mm$, slice thickness = $2\ mm$, voxel size = $2 \times 2 \times 2\ mm^3$. The diffusion MRI data were processed to correct gradient nonlinearity, eddy currents, and susceptibility-induced distortions. The estimated median signal-to-noise ratio (SNR) is 21 [22, 224]. MGH CDMD provides the real part of dMRI signals for some subjects. In this chapter, we only use the signal magnitude. More details about the data acquisition and processing can be found in the work of Tian et al. [22].

5.3 Synthetic dataset generation

This section aims to construct a dataset comprising simulated dMRI signals and microstructure parameters of artificial brain voxels. In practice, a gray matter brain voxel of $1\ mm^3$ is a medium comprising tens of thousands of cell bodies, millions of neurites, blood vessels, extracellular space, etc. [12]. We make various simplifications to brain voxels to model such complex tissue. First, we ignore compartments like blood vessels because cells and ECS occupy most of the

volume. Second, we model the ECS compartment by an isotropic free diffusion space. Ideally, we could wrap the neuron meshes with another mesh to model ECS. However, neurons are tightly intertwined in real brain tissue. Building a geometric model for ECS requires densely packing a large number of neurons in a tiny cube to achieve a reasonable ECS volume fraction ($\sim 20\%$) [162]. Besides, these neurons cannot intersect with each other. Neuron packing is still an open problem that we have not solved. As a compromise, we keep the free diffusion model for the ECS compartment. Third, we take the averaged signals from hundreds of neurons to represent the intracellular signals of a brain voxel.

In addition, we keep the two assumptions made by the simplified form of BT equation in section 3.1.2. They are (1) cell membranes are assumed to be impermeable; (2) the transverse relaxation is assumed to be homogeneous in a brain voxel so that the signal normalization (eq. (3.11)) can cancel the effect of transverse relaxation.

Briefly, our artificial brain voxel consists of numerous impermeable neurons and an ECS. Due to the difficulty regarding neuron packing, we do not build actual ECS meshes. The ECS is modeled as a free diffusion space parameterized by its volume fraction f_{ecs}.

This section describes the steps to generate the synthetic dataset illustrated in fig. 5.2(b). In section 5.3.1, we present the dMRI protocol and the simulation parameters. We use the optimized numerical MF to perform dMRI simulations on individual neuron meshes in NeuronSet. All simulations were completed within three weeks. According to the efficiency comparison made in sections 3.3.2 and 3.3.3, it would take *30* weeks if we adopt another simulation methods, such as finite element methods, Monte-Carlo methods, or the numerical MF without the optimization made in this thesis.

In section 5.3.2, we explain how to compute voxelwise signal attenuations based on the simplified brain voxel model presented above. Furthermore, some microstructural properties of artificial brain voxels can be computed from the measurements on neuron meshes. In this way, each artificial brain voxel is related to a set of simulated signals and a group of microstructure parameters. By randomly combining neurons in NeuronSet and adding ECS compartments, we got 1.4 million artificial brain voxels whose signals and microstructure parameters form the synthetic dataset.

5.3.1 Simulating signals from individual neurons

We start with the simulation on individual neuron meshes using a similar dMRI protocol as MGH CDMD:

- PGSE sequences with $\delta/\Delta = 8/19$ or $8/49\ ms$;

- 64 non-zero gradient intensities linearly space between 0 (not included) and 290 mT/m;

- 32 diffusion encoding directions uniformly distributed on a hemisphere (equivalent to 64 directions on a sphere because simulated signals are antipodally symmetric).

The diffusivity inside neuron is fixed to be $3 \times 10^{-3}\ \mu m^2/\mu s$, the water self-diffusion coefficient at $37\ ^\circ C$ [34]. We chose it because cytoplasm consists of 80 - 97 % water [225] [11, p. 128]. Macromolecules and organelles can indeed hinder the diffusion of water molecules. However, given the high proportion of water, we do not think the reduction is significant.

The simulations are performed by the well-optimized numerical MF that requires two simulation parameters, H and τ_{min}, as explained in section 3.2.2. The parameter H controls the fineness of neuron meshes, and τ_{min} determines the number of retained eigenvalues. We choose $H = -1$ and $\tau_{min} = 76\ \mu s$ for the numerical MF used in this chapter. Tetgen uses an adaptive method to discretize the volume and add new points to improve the mesh quality when H is set to -1 [111]. The minimum characteristic time scale of $76\ \mu s$ means that we keep all eigenvalues less than $4.39\ \mu m^{-2}$ whose characteristic length scales are greater than $1.5\ \mu m$. Next, we validate the choice of H and τ_{min}.

Validation of simulation parameters

To validate the choice about $H = -1$ and $\tau_{min} = 76\ \mu s$, we compare the numerical matrix formalism with a FEM simulator implemented in SpinDoctor [44]. The FEM simulations with refined discretization in space and time give the reference solutions. The simulation parameters of the FEM simulator are $H = 0.5\ \mu m^3$, $rtol = 10^{-5}$, $atol = 10^{-7}$. We refine the neuron meshes by setting H to $0.5\ \mu m^3$. A comparison between $H = -1$ and $H = 0.5\ \mu m^2$ is given in fig. 5.3.

As for diffusion MRI protocol, we fix the gradient intensity to the maximum value used in MGH CDMD ($290\ mT/m$) because a strong gradient often suffers large numerical errors [50, 116]. Nine gradient directions are evenly distributed in a semicircle and parameterized by an angle χ.

We denote by S_{MF} and S_{FEM} the signals simulated by the numerical matrix formalism and the finite element method, respectively. Figure 5.3(c) gives the relative errors in percent ($|S_{\text{MF}} - S_{\text{FEM}}|/|S_{\text{FEM}}| \times 100\%$) at nine directions for three randomly picked cells[1]. The relative errors are below 4%, which indicates a satisfactory simulation accuracy with the chosen simulation parameters.

[1]The IDs of the three cells in NeuroMorpho.Org are NMO_01042, NMO_85592, and NMO_85632.

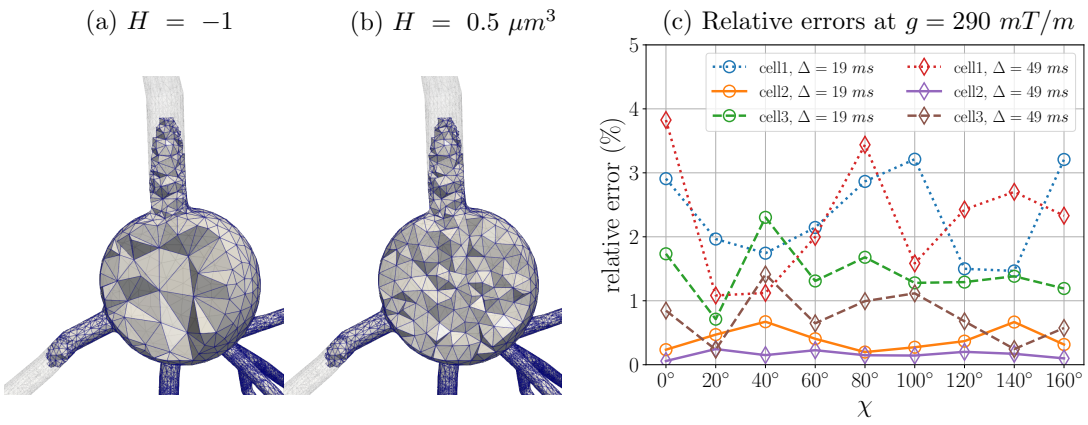


Figure 5.3: Simulation accuracy of the numerical matrix formalism with $H = -1$ and $\tau_{min} = 76\ \mu s$. **(a)** a tetrahedral mesh for the numerical matrix formalism. The default discretization routine of Tetgen generates the tetrahedra. Some parts of the mesh are made transparent to show the internal meshing. **(b)** the refined tetrahedral mesh for the FEM simulator. The maximum volume of tetrahedra is $0.5\ \mu m^3$. Some parts of the mesh are made transparent to show the mesh refinement. **(c)** the relative errors of the numerical matrix formalism for three randomly picked cells with two diffusion times. The gradient intensity is fixed to $290\ mT/m$. In NeuroMorpho.Org, the IDs of the three cells are NMO_01042 (cell1), NMO_85592 (cell2), and NMO_85632 (cell3). The meshes in (a) and (b) correspond to cell2. When $H = -1$, the numbers of FE nodes of the three cells are 32294, 48551, and 79992, respectively. When $H = 0.5\ \mu m^3$, the numbers of FE nodes are 109660, 80940, and 163905, respectively. The FEM simulations are conducted on the refined meshes to give the reference solution S_{FEM}. The relative errors in percent are $|S_{\text{MF}} - S_{\text{FEM}}|/|S_{\text{FEM}}| \times 100\%$.

5.3.2 Computing signals from artificial brain voxels

Once the signals from every neuron mesh in NeuronSet are obtained, we can compute the signal attenuation from artificial brain voxels by adding intra- and extracellular signals. Suppose an artificial brain voxel contains M neurons and an ECS compartment whose volume fraction is f_{ecs}. According to eq. (3.20), the signal attenuation arising from the brain voxel can be computed by

$$E^v(g, \boldsymbol{u}_g, \delta, \Delta) = (1 - f_{\text{ecs}}) \times \frac{\sum_{i=1}^{M} V_i \cdot E_i}{\sum_{i=1}^{M} V_i} + f_{\text{ecs}} \times e^{-D_{\text{ecs}} b}, \tag{5.1}$$

where the subscription i indicates the i-th neuron, V_i is the neuronal volume measured on the i-th neuron mesh, E_i is the signal attenuation of the i-th neuron, and the diffusivity is $D_{\text{ecs}} = 3 \times 10^{-3}\ \mu m^2/\mu s$. The choice of D_{ecs} will be discussed in section 5.7.4. The signal attenuation from an artificial brain voxel

is denoted by E^v. Since we employ PGSE sequences, E^v is a function of g, $\boldsymbol{u}_g$, δ, and Δ.

We randomly pick M neurons from the mesh database to diversify the brain voxel compositions. The number M ranges from 1 to 500. Each combination of M cells is then aggregated with ten different ECSs whose volume fractions follow a Gaussian distribution $\mathcal{N}(\mu = 0.5, \sigma^2 = 0.25^2)$. The Gaussian distribution is chosen to have a wide distribution without giving too much weight to extreme cases (f_{ecs} close to 0 or 1). The combination of cells and ECS produces signal attenuations from 1.4 million distinct artificial brain voxels.

5.3.3 Brain voxel microstructure parameters

Section 2.5.2 provides over 30 neuroanatomical parameters measured on neuron meshes, which allow us to compute the microstructure parameters of an artificial brain voxel consisting of M neuron meshes and an ECS compartment whose volume fraction is f_{ecs}. Some important brain voxel microstructure parameters are

1. soma volume fraction: $f_{\text{soma}} = (1 - f_{\text{ecs}}) \dfrac{\sum_{m=1}^{M} V_{\text{soma}}^m}{\sum_{m=1}^{M} V_{\text{neuron}}^m}$,
2. neurite volume fraction: $f_{\text{neurite}} = (1 - f_{\text{ecs}}) \dfrac{\sum_{m=1}^{M} V_{\text{neurite}}^m}{\sum_{m=1}^{M} V_{\text{neuron}}^m}$,
3. soma area fraction: $a_{\text{soma}} = \dfrac{\sum_{m=1}^{M} A_{\text{soma}}^m}{\sum_{m=1}^{M} A_{\text{neuron}}^m}$,
4. neurite area fraction: $a_{\text{neurite}} = \dfrac{\sum_{m=1}^{M} A_{\text{neurite}}^m}{\sum_{m=1}^{M} A_{\text{neuron}}^m}$,
5. average soma radius: $\overline{r}_{\text{soma}} = \dfrac{\sum_{m=1}^{M} r_{\text{soma}}^m}{M}$,
6. volume weighted average soma radius: $\overline{r}_{\text{soma}}^{vw} = \dfrac{\sum_{m=1}^{M} V_{\text{soma}}^m r_{\text{soma}}^m}{\sum_{m=1}^{M} V_{\text{soma}}^m}$,

where the superscript indicated the m-th neuron. The list is not exhaustive. The definitions of over 40 brain voxel microstructure parameters are listed in section D.1. We emphasize that not all microstructure parameters can be probed by diffusion MRI. Whether we can estimate a parameter mainly depends on the dynamics of MR physics.

This study focuses on estimating volume and area fractions. Figure 5.4 presents the distribution of the volume and area fractions in the synthetic

dataset. We denote the three volume fractions of an artificial brain voxel by $\boldsymbol{L}_{\text{vol}} = [f_{\text{soma}}, f_{\text{neurite}}, f_{\text{ecs}}]^T$, and the area fractions by $\boldsymbol{L}_{\text{area}} = [a_{\text{soma}}, a_{\text{neurite}}]^T$. Note that all fractions are positive and the sum of elements of $\boldsymbol{L}_{\text{vol}}$ or $\boldsymbol{L}_{\text{area}}$ is one.

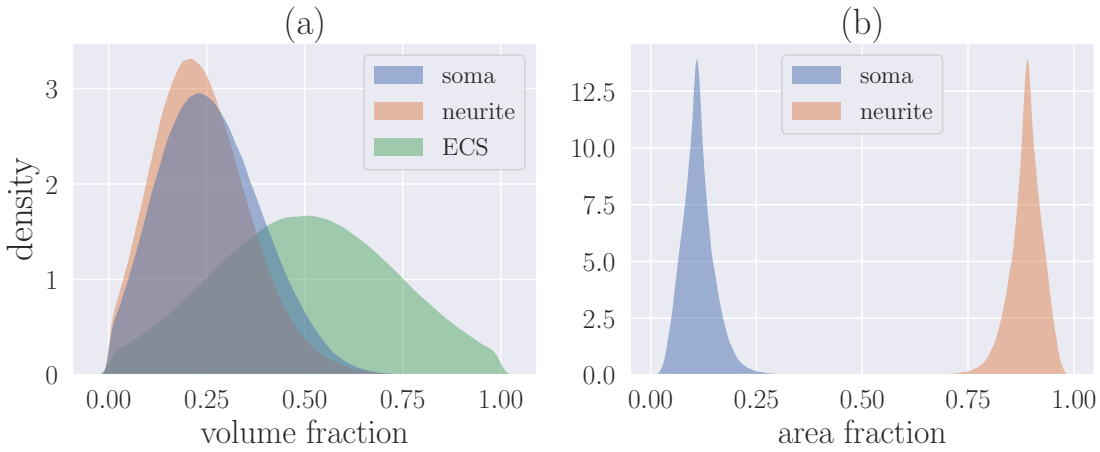


Figure 5.4: The distribution of volume and area fractions in the synthetic dataset. **(a)** the volume fraction distribution of soma, neurites, and ECS. The ECS volume fraction follows a Gaussian distribution $\mathcal{N}(\mu = 0.5,\ \sigma^2 = 0.25^2)$. The soma and neurite volume fractions are computed based on realistic neuron meshes. **(b)** the area fraction distribution of soma and neurites.

The signal attenuations and the microstructure parameters from 1.4 million artificial brain voxels form the synthetic dataset. The numerical experiments conducted in section 3.4 hint at the possibility of mappings from signals to microstructure parameters. The next section concerns training machine learning models to approximate the mappings that are implicitly contained in the synthetic dataset.

5.4 Model training

This section describes the steps for training machine learning models with the synthetic dataset. We follow the standard training process described in the book of Goodfellow et al. [226]. The main steps are

1. preprocessing the raw data to build derived values (features) for subsequent training steps;

2. setting up the training configuration, including the type of ML model, the

training set $\mathcal{T}_{\text{train}}$, the test set $\mathcal{T}_{\text{test}}$, the loss function, the optimization algorithm, etc.

3. determining the hyperparameters using the training set $\mathcal{T}_{\text{train}}$;
4. training ML models with the chosen hyperparameters on the training set $\mathcal{T}_{\text{train}}$;
5. evaluating the performance of trained models on the test set $\mathcal{T}_{\text{test}}$.

5.4.1 Data preprocessing

Each artificial brain voxel has 4096 signals simulated with 2 diffusion times, 64 gradient intensities, and 32 gradient directions. They are the raw data stored in the synthetic dataset. The goal of the data preprocessing is to build derived values, also referred to as *features* in the ML community, intended to be informative and low dimensional, facilitating subsequent learning steps. Features are the inputs of a machine learning model.

First type of features: direction-averaged signals

In this study, the first step of data preprocessing involves reducing the dimensionality of the raw data. We average the simulated signal attenuations over all measured directions to get the direction-averaged signal

$$\overline{E}^v(g,\delta,\Delta) = \frac{1}{N_{\text{dir}}}\sum_{i=1}^{N_{\text{dir}}} E^v(g,\boldsymbol{u}_g^i,\delta,\Delta), \tag{5.2}$$

where N_{dir} is the number of gradient directions and $\boldsymbol{u}_g^i$ is the i-th gradient direction for a given gradient intensity g. Averaging over directions is a common practice to reduce the data dimensionality in the dMRI literature [227]. It is also helpful in denoising the experimental data. However, the averaging removes all the orientation-dependent information. We can not estimate, for example, the orientation of white matter tracts using direction-averaged signals.

The direction-averaged signals are the first type of features we will use. We denote the direction-averaged signals from a brain voxel at the short diffusion time ($\delta/\Delta = 8/19\ ms$) by $\boldsymbol{F}_{\text{sig19}}$, and the long diffusion time ($\delta/\Delta = 8/49\ ms$) by $\boldsymbol{F}_{\text{sig49}}$. They are n-dimensional vectors

$$\boldsymbol{F}_{\text{sig19}} = \left[\overline{E}^v(g_1,8,19),\ ...,\ \overline{E}^v(g_n,8,19)\right]^T \in [0,1]^n, \tag{5.3}$$

$$\boldsymbol{F}_{\text{sig49}} = \left[\overline{E}^v(g_1,8,49),\ ...,\ \overline{E}^v(g_n,8,49)\right]^T \in [0,1]^n, \tag{5.4}$$

where the number of signals n will be determined in section 5.4.3.

Second type of features: five markers

Incorporating domain-specific knowledge helps obtain more concise features. The statistical study presented in section 3.4.4 shows that the inflection point of a signal curve helps define six markers useful for the soma size estimation. This chapter adopts the same idea and uses five markers as the second set of features. Compared to the six markers defined in section 3.4.4, we keep the first four markers, i.e., x_0, y_0, c_0, and c_1, that are relatively easy to obtain using simulated signals. We replace the last two markers with the apparent diffusion coefficient D_e defined in eq. (3.13).

The five markers are the second type of features used in this chapter. We denote the markers obtained using direction-averaged signals at the short diffusion time by $\boldsymbol{F}_{\text{mk19}}$, and markers at the long diffusion time by $\boldsymbol{F}_{\text{mk49}}$. They are five-dimensional vectors

$$\boldsymbol{F}_{\text{mk19}} = \left[x_0^{19}, y_0^{19}, c_0^{19}, c_1^{19}, D_e^{19}\right]^T \in \mathbb{R}^5, \tag{5.5}$$

$$\boldsymbol{F}_{\text{mk49}} = \left[x_0^{49}, y_0^{49}, c_0^{49}, c_1^{49}, D_e^{49}\right]^T \in \mathbb{R}^5. \tag{5.6}$$

The superscripts ^{19} and ^{49} indicate the diffusion times.

To summarize, we adopt two types of features in this chapter. One includes the direction-averaged signals, and another is composed of the five markers derived from signals. By incorporating domain-specific knowledge, one can also derive other features from the signals to improve the sensitivity to certain microstructure parameters.

5.4.2 Training configuration

Machine learning, especially deep learning, has developed rapidly in the last decade with the help of the increase in computing power. Neural networks are believed to be superior in function approximation [218–220], especially in high dimensions [221, 222]. In this chapter, we choose multilayer perceptrons (MLPs) [228, 229] to infer the microstructure parameters of interest.

The datasets for training MLPs are subsets derived from the synthetic dataset. We denote a training dataset by $\mathcal{T} = \{(\boldsymbol{X}_i, \boldsymbol{Y}_i), i \in \{1, ..., N_{\text{voxel}}\}\}$ where N_{voxel} (= 1.4 million) is the number of the artificial brain voxels. We refer to a tuple $(\boldsymbol{X}, \boldsymbol{Y})$ as a data point. The input of an MLP is denoted by $\boldsymbol{X}$. The ground-truth output, also known as the *label* in ML community, is denoted by $\boldsymbol{Y}$. The eight combinations of input and output studied in this chapter are

1. Case 1: $\boldsymbol{X} = \boldsymbol{F}_{\text{sig19}}$, $\boldsymbol{Y} = \boldsymbol{L}_{\text{vol}}$;
2. Case 2: $\boldsymbol{X} = \boldsymbol{F}_{\text{sig49}}$, $\boldsymbol{Y} = \boldsymbol{L}_{\text{vol}}$;

3. Case 3: $\boldsymbol{X} = \boldsymbol{F}_{\text{mk19}}$, $\boldsymbol{Y} = \boldsymbol{L}_{\text{vol}}$;
4. Case 4: $\boldsymbol{X} = \boldsymbol{F}_{\text{mk49}}$, $\boldsymbol{Y} = \boldsymbol{L}_{\text{vol}}$;
5. Case 5: $\boldsymbol{X} = \boldsymbol{F}_{\text{sig19}}$, $\boldsymbol{Y} = \boldsymbol{L}_{\text{area}}$;
6. Case 6: $\boldsymbol{X} = \boldsymbol{F}_{\text{sig49}}$, $\boldsymbol{Y} = \boldsymbol{L}_{\text{area}}$;
7. Case 7: $\boldsymbol{X} = \boldsymbol{F}_{\text{mk19}}$, $\boldsymbol{Y} = \boldsymbol{L}_{\text{area}}$;
8. Case 8: $\boldsymbol{X} = \boldsymbol{F}_{\text{mk49}}$, $\boldsymbol{Y} = \boldsymbol{L}_{\text{area}}$.

To improve the robustness of MLPs to noise, we add Racian noise to the direction-averaged signals used in items 1, 2, 5 and 6. We keep the same SNR as the MGH CDMD database ($S_0/\sigma_{\text{R}} = 21$), where σ_{R} is the Rician scaling parameter. We recall that all elements in $\boldsymbol{Y}$ are positive and the sum is one, which is an additional constraint that should be accounted for when training.

We randomly select one million data points from $\mathcal{T}$ to form the training set $\mathcal{T}_{\text{train}}$; the rest (over 450,000 data points) makes up the test set $\mathcal{T}_{\text{test}}$ which is separated and not used for model training. The only role of the test set is to assess the generalization of a trained MLP [226].

An MLP is a nonlinear function h parameterized by its weights $\boldsymbol{\theta}$ [226]. The model training is to find optimal weights $\boldsymbol{\theta}^*$ that minimize the distance between the MLP's output and the ground truth (known as the label in the ML literature)

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{\#\mathcal{T}_{\text{train}}} \sum_{i=1}^{\#\mathcal{T}_{\text{train}}} \|\boldsymbol{Y}_i - h(\boldsymbol{X}_i; \boldsymbol{\theta})\|_2^2. \tag{5.7}$$

Here, we use the mean squared error (MSE) as the loss function. The minimization is possible if an underlying function ζ mapping $\boldsymbol{X}$ to $\boldsymbol{Y}$ exists. Once the optimization converged, the trained MLP could be a good approximation of the underlying function, i.e., $h(\cdot; \boldsymbol{\theta}^*) \simeq \zeta$ in the sense of minimizing L2 distance in the training set. Nonetheless, such an underlying function may not exist, and convergence is not guaranteed. The generalization of the trained MLP to unseen data also needs to be assessed.

The function ζ varies with the choices of $\boldsymbol{X}$ and $\boldsymbol{Y}$, and the MR physics determines its existence. Even if ζ exists, we must be careful about the activation function, initial weights, and the optimization algorithm to reach the convergence [230].

We employ the Gaussian error linear unit (GELU) [231], a ReLU-like (Rectified Linear Unit) activation function that incorporates the properties of stochastic regularizers such as dropout [232]. The weights $\boldsymbol{\theta}$ are initialized using Kaiming initialization [233] because we employ ReLU-like activation functions. The

optimization is performed with a variant of the Adam optimizer that has a long-term memory of past gradients to enhance the convergence [234, 235]. The initial learning rate is 0.01, the batch size is 10,000, and the maximum number of epochs is 500.

The architecture of an artificial neural network can also significantly affect its performance. Finding a suitable artificial neural network architecture for brain microstructure estimation is a subject worth investigating in the future. In this chapter, we concentrate on four-layer MLPs. To guarantee the outputs are all positive and sum to unity, we append a softmax function [236] to the output layer. The implementation and training of MLPs are performed with PyTorch [237].

Finally, we assess the performance of a trained MLP in the held-out test set $\mathcal{T}_{\text{test}}$. We use the L1-norm to evaluate the test loss

$$l_{\text{test}} = \frac{1}{dim(\boldsymbol{Y})\#\mathcal{T}_{\text{test}}} \sum_{i=1}^{\#\mathcal{T}_{\text{test}}} \|\boldsymbol{Y}_i - h(\boldsymbol{X}_i; \boldsymbol{\theta}^*)\|_1, \ (\boldsymbol{X}_i, \boldsymbol{Y}_i) \in \mathcal{T}_{\text{test}}. \tag{5.8}$$

The test loss estimates the mean absolute error between the ground truth and the predicted values in the test set.

5.4.3 Hyperparameter tuning

Four-layer MLPs have several hyperparameters, namely, the size of the input layer n, the first hidden layer n_1, the second hidden layer n_2, and the output layer n_3. To determine the hyperparameters, we split out 20% of the training set $\mathcal{T}_{\text{train}}$ as the validation set $\mathcal{T}'_{\text{v}}$. The remaining eighty percent constitute a new training set $\mathcal{T}'_{\text{train}}$. We train MLPs on $\mathcal{T}'_{\text{train}}$ and compute the validation error using L1-loss on $\mathcal{T}'_{\text{v}}$. The validation errors help determine the hyperparameters. We focus on the following hyperparameters for the eight cases listed in section 5.4.2

1. Case 1 and Case 2: $(n, n_1, n_2, n_3) = (16, 16, 8, 3)$, $(16, 32, 16, 3)$, $(32, 32, 16, 3)$, $(32, 64, 32, 3)$, $(64, 64, 32, 3)$, or $(64, 128, 64, 3)$;

2. Case 5 and Case 6: $(n, n_1, n_2, n_3) = (16, 16, 8, 2)$, $(16, 32, 16, 2)$, $(32, 32, 16, 2)$, $(32, 64, 32, 2)$, $(64, 64, 32, 2)$, or $(64, 128, 64, 2)$;

3. Case 3 and Case 4: $(n, n_1, n_2, n_3) = (5, 10, 5, 3)$, $(5, 10, 10, 3)$, $(5, 20, 10, 3)$, $(5, 20, 20, 3)$, $(5, 30, 15, 3)$, or $(5, 30, 30, 3)$;

4. Case 7 and Case 8: $(n, n_1, n_2, n_3) = (5, 10, 5, 2)$, $(5, 10, 10, 2)$, $(5, 20, 10, 2)$, $(5, 20, 20, 2)$, $(5, 30, 15, 2)$, or $(5, 30, 30, 2)$;

It is worth noting that, for cases 1, 2, 5, and 6, the size of the input layer n equals the number of direction-averaged signals (see eqs. (5.3) and (5.4)).

In total, we trained forty-eight MLPs (six sets of hyperparameters for each case). Figure 5.5 demonstrates the final validation errors of the forty-eight MLPs. It can be seen that a more complex network structure usually has a lower validation error. Hence, the selected hyperparameters for the above four items are $(64, 128, 64, 3)$, $(64, 128, 64, 2)$, $(5, 30, 30, 3)$, and $(5, 30, 30, 2)$, respectively.

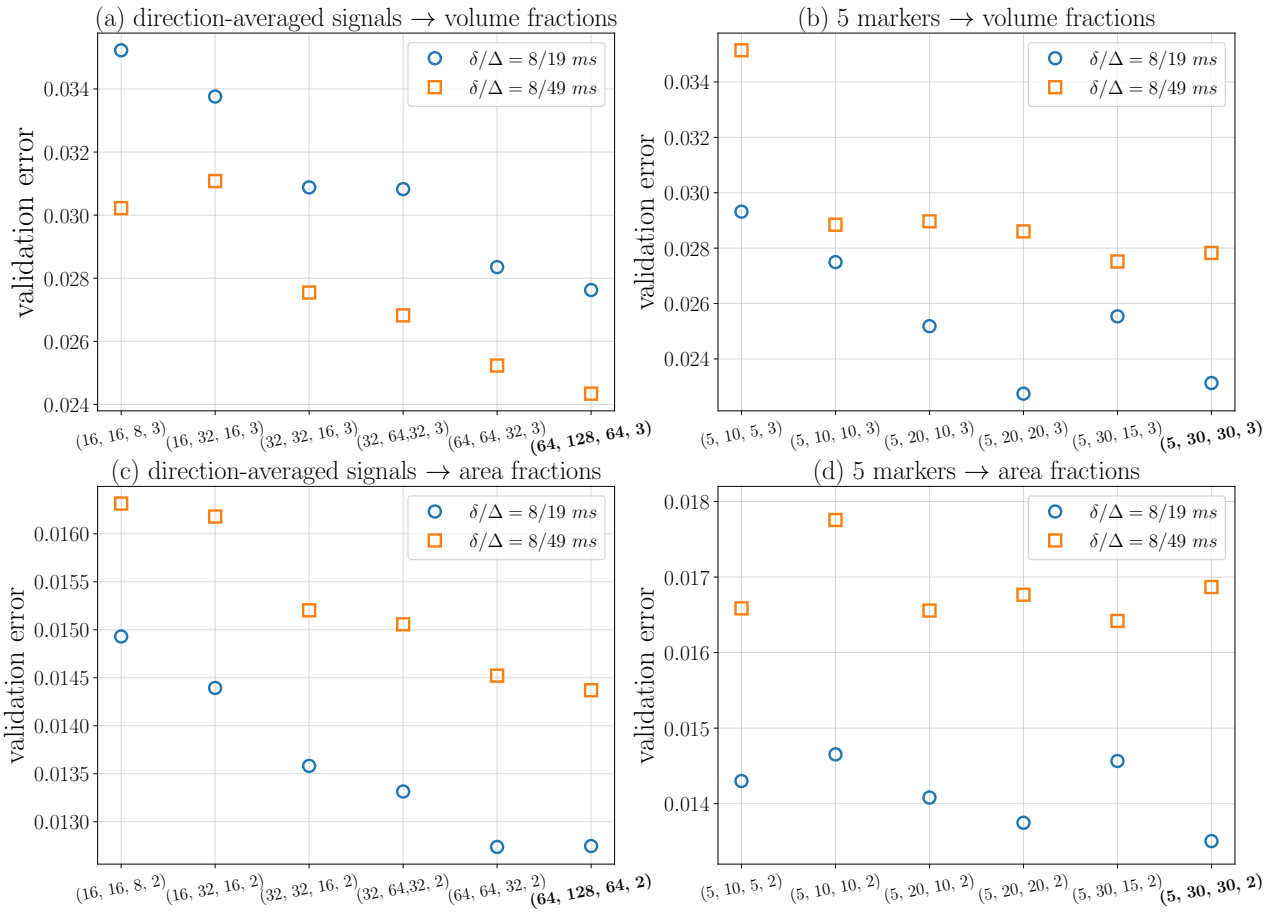


Figure 5.5: The validation errors for tuning hyperparameters. A blue circle corresponds to an MLP for the short diffusion time. An orange square is for the long diffusion time. The labels of the x-axis are the hyperparameters. The selected hyperparameters are in bold. **(a)** the validation errors of twelve MLPs whose hyperparameters are shown on the x-axis. The MLPs predict the volume fractions using the direction-averaged signals. **(b)** the validation errors of MLPs for predicting volume fractions using the five markers. **(c)** the validation errors of MLPs for predicting area fractions using the direction-averaged signals. **(d)** the validation errors of MLPs for area fraction estimation using the five markers.

Finally, we train MLPs with the chosen hyperparameters on the original train-

ing set $\mathcal{T}_{\text{train}}$. In the next section, we assess the final performance of each trained MLP in the held-out test set $\mathcal{T}_{\text{test}}$.

5.4.4 Validation on test set

We use the L1-loss to quantify the test loss (see eq. (5.8)). For clarity, we assign each trained MLP a name and list the network structures, the input, the output, the final test losses, and the R^2 scores in table 5.1. The average test loss is about 0.01, while the volume or area fraction is of the order of 0.1. The R^2 scores for volume fraction estimation are around 0.9, and about 0.6 for area fraction estimation. Note that four MLPs, `mlp_sig_vol_19`, `mlp_sig_vol_49`, `mlp_sig_area_19`, and `mlp_sig_area_49`, are tested under a noise condition similar to MGH CDMD (SNR = 21). The remaining MLPs, which take the five markers as input, are assessed on noise-free test sets.

MLP name	MLP structure	diffusion time δ/Δ [ms]	input	output	test loss	R^2 scores
`mlp_sig_vol_19`	64, 128, 64, 3	8/19	64 signals	$f_{\text{soma}}/f_{\text{neurite}}/f_{\text{ecs}}$	0.0223	0.94/0.95/0.98
`mlp_sig_vol_49`	64, 128, 64, 3	8/49	64 signals	$f_{\text{soma}}/f_{\text{neurite}}/f_{\text{ecs}}$	0.0187	0.95/0.94/0.99
`mlp_mk_vol_19`	5, 30, 30, 3	8/19	5 markers	$f_{\text{soma}}/f_{\text{neurite}}/f_{\text{ecs}}$	0.0171	0.95/0.94/0.99
`mlp_mk_vol_49`	5, 30, 30, 3	8/49	5 markers	$f_{\text{soma}}/f_{\text{neurite}}/f_{\text{ecs}}$	0.0218	0.93/0.90/0.99
`mlp_sig_area_19`	64, 128, 64, 2	8/19	64 signals	$a_{\text{soma}}/a_{\text{neurite}}$	0.0157	0.66
`mlp_sig_area_49`	64, 128, 64, 2	8/49	64 signals	$a_{\text{soma}}/a_{\text{neurite}}$	0.0174	0.62
`mlp_mk_area_19`	5, 30, 30, 2	8/19	5 markers	$a_{\text{soma}}/a_{\text{neurite}}$	0.0153	0.66
`mlp_mk_area_49`	5, 30, 30, 2	8/49	5 markers	$a_{\text{soma}}/a_{\text{neurite}}$	0.0196	0.51

Table 5.1: Summary of the eight MLPs with hyperparameters determined in section 5.4.3. The structure of an MLP is represented by four numbers n, n_1, n_2, n_3, i.e., the input layer size n, the first hidden layer size n_1, the second hidden layer size n_2, and the output layer size n_3. The inputs of the MLPs are either the direction-averaged signals at 64 gradient intensities or the five markers. The outputs are volume or area fractions. Four MLPs, `mlp_sig_vol_19`, `mlp_sig_vol_49`, `mlp_sig_area_19`, and `mlp_sig_area_49`, are tested under the noise condition similar to MGH CDMD (SNR = 21). The rest of the MLPs, which take the five markers as input, are assessed on noise-free test sets. We list the final test losses of MLPs (using the held-out test set). The three R^2 scores of the first four MLPs are for soma, neurite, and ECS, respectively. For example, the R^2 scores of soma, neurite, and ECS volume fraction estimation using `mlp_sig_vol_19` are 0.94, 0.95, and 0.98, respectively. Because the sum of soma and neurite area fractions is unity, the soma area fraction estimation has the same R^2 score as neurite.

To further demonstrate the performance of each MLP, we plot the absolute and relative error distribution in the held-out test set, which contains over 450,000 data points. We denote the ground-truth fraction by f_{gt}, which, for instance, could be a soma volume fraction or neurite area fraction. The predicted

value is denoted by f_{pred}. We define the absolute error as the difference between the prediction and the ground-truth value $f_{\text{pred}} - f_{\text{gt}}$, and the relative error as $(f_{\text{pred}} - f_{\text{gt}})/f_{\text{gt}} \times 100\%$. We do not take the absolute value, so a negative error corresponds to underestimation.

Figure 5.6 summarizes the distributions of the absolute errors for the eight MLPs using box plots. Section D.2 explains the box plot definition. The MLPs' estimations have median absolute errors close to zero. Over fifty percent of the absolute errors for volume fraction estimation are below 0.025. The MLPs for area fraction estimation have similar performance. Most of the absolute errors are below 0.025.

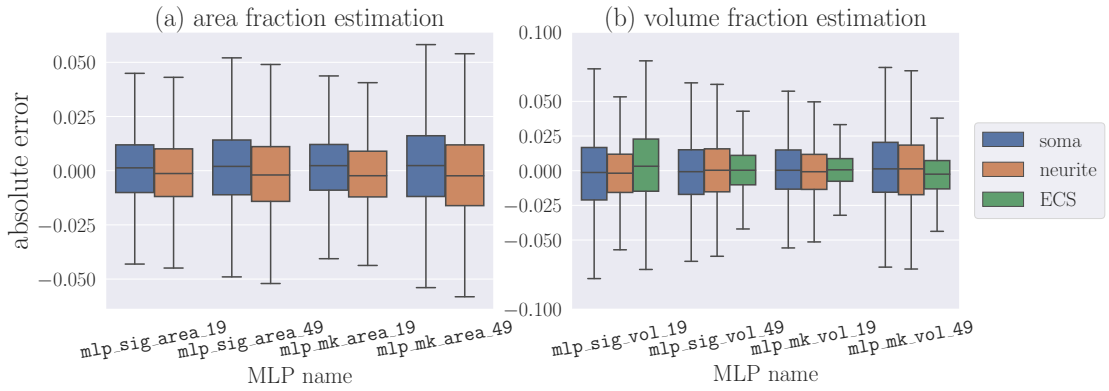


Figure 5.6: The box plots summarizing the distributions of the absolute errors. The center line of a box indicates the median. A box's lower and upper sides (hinges) denote the first and third quartiles. The range between the two hinges is the interquartile range, which contains 50% of data points. The whiskers extend to the range of 1.5 times the interquartile range. Outliers are ignored for clarity. **(a)** the absolute error distributions for area fraction estimation. **(b)** the absolute error distributions for volume fraction estimation.

Similarly, fig. 5.7 demonstrates the distributions of the relative errors. The median relative errors are close to zero, meaning the predictions do not suffer significant bias in the synthetic test set. Over fifty percent of the relative errors for volume fraction estimation are below 10%. The soma area fraction estimation has a much larger relative error than the neurite area fraction estimation. This is because the average soma area fraction is ~ 0.1, whereas the average neurite area fraction is ~ 0.9 (see fig. 5.4(c)). Because soma and neurite have similar absolute errors, a nine times difference in the relative error is reasonable. Nevertheless, most relative errors for soma area fraction estimation are below 20%.

The performance of the eight MLPs shows that they can predict the desired

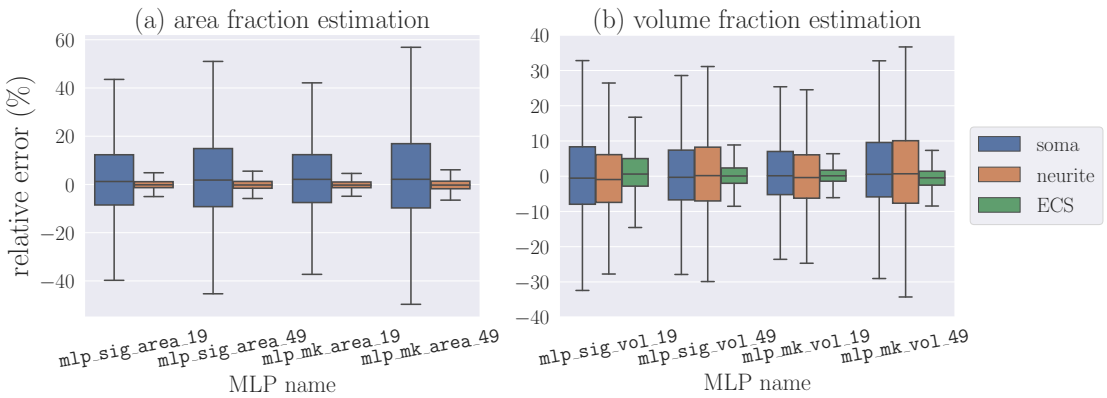


Figure 5.7: The box plots summarizing the distributions of the relative errors. A box plot denotes the median, interquartile range, and 1.5 times the interquartile range by the center line, hinges, and whiskers. Outliers are ignored for clarity. **(a)** the relative error distributions for area fraction estimation. **(b)** the relative error distributions for volume fraction estimation.

microstructure parameters with acceptable errors. Next, we apply the eight MLPs to the MGH CDMD dataset to get in vivo parameter maps.

5.5 In vivo microstructure parameter estimation

In this section, we apply the MLPs trained in the previous section to the experimental data in the MGH CDMD dataset to infer volume and area fractions.

5.5.1 Signal interpolation

Real-world dMRI signals are acquired at much fewer gradient intensities than simulation. For example, the dMRI signals in the MGH CDMD dataset are sampled with only eight gradient intensities. However, we train MLPs with the direction-averaged signals at 64 fixed gradient intensities or the five markers whose computation requires signals at numerous gradient intensities. We need to interpolate the experimental data to get 64 signals or compute the five markers based on eight experimental direction-averaged signals.

Practitioners can choose gradient intensities within a fixed range for an MRI scanner. Suppose we have one measurement near zero gradient intensity, one near the maximum gradient, and several acquisitions in between. With these measurements, it is possible to interpolate the direction-averaged signals within the fixed range.

We demonstrate the way of interpolation using signals from MGH CDMD.

Let's focus on the dMRI signals from a brain voxel with a fixed sequence, e.g., PGSE sequence with $\delta/\Delta = 8/19\ ms$. Some preprocessing steps before the interpolation include signal normalization (eq. (3.20)) and directional averaging (eq. (5.2)) to get the direction-averaged signal $\overline{E}$. To perform interpolation, we express $\overline{E}$ as a function of $\beta = 1/\sqrt{b}$. In increasing order, we denote the eight bvalues by $b_1, ..., b_8$. So b_1 is the smallest non-zero bvalue. The corresponding β's are $\beta_1, ..., \beta_8$. Since we took the directional averaging and fixed the gradient sequence time profile, b or β can now fully characterize $\overline{E}$.

We adopt the fourth-order B-spline interpolation implemented in Scipy [238]. A vanilla cubic spline suffers a large fluctuation (see fig. 5.8). To moderate the fluctuation, we adopt the Gaussian phase assumption when bvalues are smaller than b_1. The GPA allows us to approximate $\overline{E}$ by $e^{D_e \cdot b} = e^{D_e/\beta^2}$ for $b \leq b_1$. Furthermore, the GPA provides two boundary conditions which are the continuity of the first and second derivatives at β_1:

$$\overline{E}'(\beta_1) = 2D_e \cdot e^{-D_e/\beta_1^2}/\beta_1^3, \tag{5.9}$$

$$\overline{E}''(\beta_1) = \left(4D_e^2 - 6D_e\beta_1^2\right) e^{-D_e/\beta_1^2}/\beta_1^6, \tag{5.10}$$

where the primes indicate derivative with respect to β.

At the high bvalue end (small β), we adopt the "natural" boundary condition [239]

$$\overline{E}''(\beta_8) = 0. \tag{5.11}$$

The boundary conditions help moderate the fluctuation of the interpolation and allow us to sample g's or bvalues within the maximum gradient intensity and find the inflection point. Figure 5.8 demonstrates the measured and interpolated signals and the tangent line passing through the estimated inflection point. It is worth noting that one should not extrapolate the direction-averaged signals beyond the maximum gradient intensity.

Validation of the interpolation method

The interpolation inevitably brings in errors. We assess the interpolation error using data from the MGH CDMD dataset. The direction-averaged signals at eight gradient intensities from a brain voxel are split into two subsets. The first set contains signals at N_1 ($4 \leq N_1 < 8$) gradient intensities, including the lowest and the highest gradients. The second set includes the rest N_2 signals.

Following the interpolation method described above, we obtain the fourth-order B-spline polynomial using the first set. We then predict the signals stored in the second set using the polynomial. The measured and predicted signals are denoted by $\overline{E}_{\mathsf{m}}$ and $\overline{E}_{\mathsf{p}}$, respectively. The interpolation error is assessed by the relative error $(\overline{E}_{\mathsf{p}} - \overline{E}_{\mathsf{m}})/\overline{E}_{\mathsf{m}} \times 100\%$.

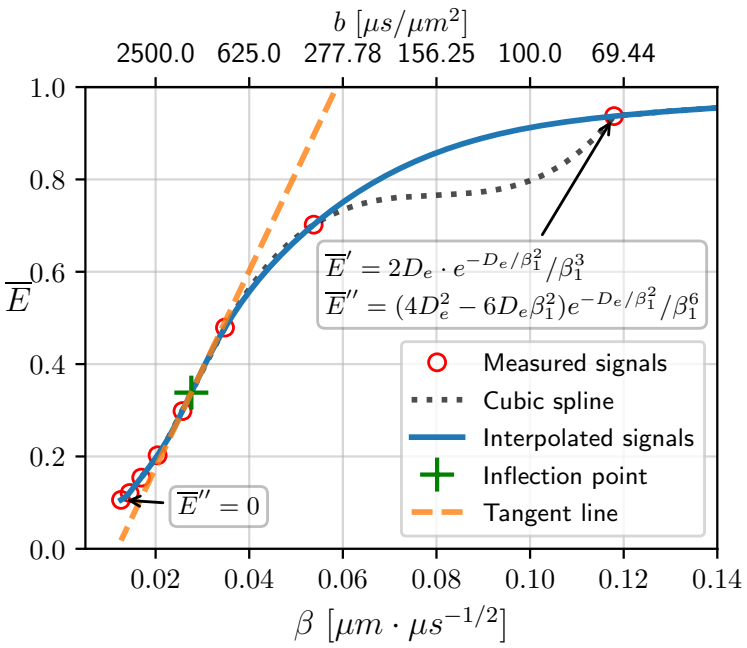


Figure 5.8: Fourth-order B-spline interpolation of direction-averaged signals. Red circles represent the direction-averaged signals at eight non-zero bvalues measured from a white matter voxel of the first subject (`sub_001`) in MGH CDMD. The voxel index is (19, 25, 73). A vanilla cubic spline interpolation represented by the dotted black line suffers a large fluctuation. By incorporating the three boundary conditions annotated in the boxes, the fourth-order B-spline method interpolates the eight measured signals giving the solid blue line. The rightmost root of the second derivative of the interpolated signals gives the inflection point shown as the green cross. The dashed orange line is the tangent line passing through the inflection point.

We adopt two splitting strategies:

1. the first set contains four direction-averaged signals whose gradient intensities are 31, 105, 179, and 290 mT/m; the second set includes four signals at 68, 142, 216, and 253 mT/m; i.e., $N_1 = 4$ and $N_2 = 4$;

2. the first set contains six direction-averaged signals whose gradient intensities are 31, 68, 105, 179, 253, and 290 mT/m; the second set includes signals at 142 and 216 mT/m; i.e., $N_1 = 6$ and $N_2 = 2$;

The relative errors are computed for all brain voxels at the N_2 gradient intensities. For example, the first subject in MGH CDMD has 142,201 brain voxels. We can obtain $142,201 \times 4$ relative errors using the first splitting strategy. The box plots in fig. 5.9 summarize the distribution of the relative errors for the first four

subjects in MGH CDMD. We refer to the first strategy as "4/4" and the second as "6/2".

We notice that the first strategy can adequately interpolate the direction-averaged signals when $\delta/\Delta = 8/19\ ms$. However, the interpolation with four signals becomes biased for the long diffusion time. Using more measured signals can help reduce interpolation errors. The second strategy is satisfactory in both cases. More than 50% predicted signals have a relative error inferior to 5%. Almost all relative errors are below 15%. We believe the second strategy is adequate for signal interpolation. We actually interpolate with eight direction-averaged signals. We can expect the actual interpolation error is even lower.

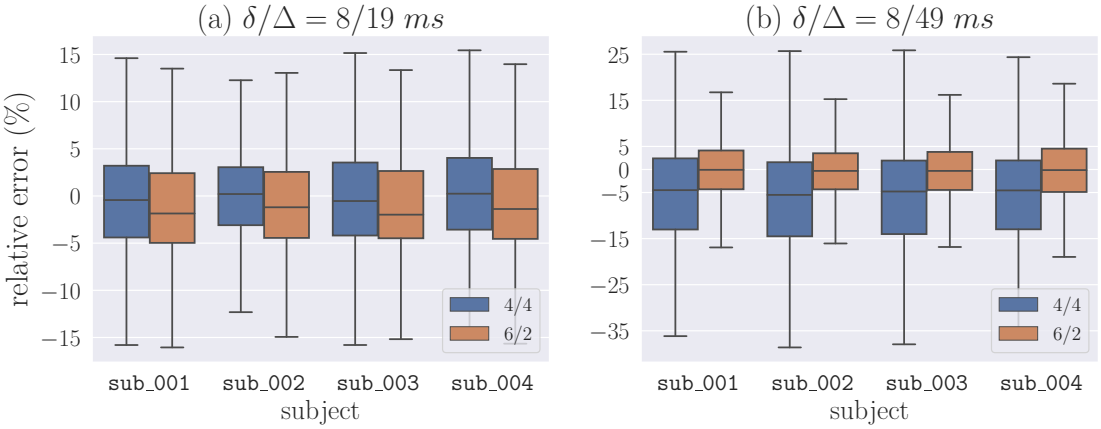


Figure 5.9: The box plots summarizing the distribution of the interpolation errors for the first four subjects in MGH CDMD. A box plot presents the median, interquartile range, and 1.5 times the interquartile range by the center line, hinges, and whiskers. Outliers are ignored for clarity. The interpolation error is assessed by the relative error $(\overline{E}_{\mathrm{p}} - \overline{E}_{\mathrm{m}})/\overline{E}_{\mathrm{m}} \times 100\%$. We refer to the first splitting strategy as "4/4" and the second as "6/2". It turns out that the second strategy is satisfactory for signal interpolation. More than 50% predicted signals have a relative error inferior to 5%. Almost all relative errors are below 15%.

5.5.2 In vivo parameter maps

We now apply the trained MLPs to the MGH CDMD dataset. Specifically, the eight direction-averaged signals from a brain voxel are interpolated to get features, namely the 64 signals ($\boldsymbol{F}_{\mathrm{sig19}}$ or $\boldsymbol{F}_{\mathrm{sig49}}$) or the five markers ($\boldsymbol{F}_{\mathrm{mk19}}$ or $\boldsymbol{F}_{\mathrm{mk49}}$). We obtain a parameter map by applying an MLP to every brain voxel of a subject.

Parameter maps during training

During the model training, the test error of an MLP decreases, meaning that the performance improves on the synthetic test set. However, the MLP may not generalize well on experimental data. We plot several parameter maps of neurite volume fraction in different training stages in fig. 5.10 to show that the performance also improves on experimental measurements. As the test error decreases, the contrast of, for example, the cerebellar white matter becomes more pronounced. Besides, the test loss becomes stable after 400 epochs. As mentioned in section 5.4, we stop the training at 500 epochs.

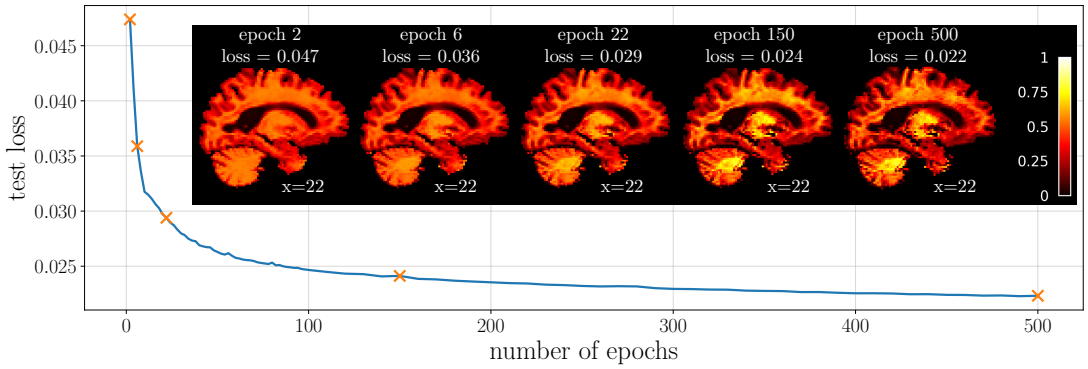


Figure 5.10: The estimation improvement on the test set and the experimental data as the test loss decreases. We plot the test losses during the training of `mlp_sig_vol_19`. The MLP at five distinct training stages is picked to infer the neurite volume fraction on experimental data. Orange crosses mark the selected epochs in the error curve. We present the evolution of a parameter map for the first subject (`sub_001`) in MGH CDMD. As the test error decreases, more details appear.

In vivo parameter maps

We obtain in vivo parameter maps by applying the trained MLPs to every brain voxel of a subject. The second subject in MGH CDMD (`sub_002`) serves as an exemple. The parameter maps of two additional subjects are reported in sections D.6 and D.7. Figure 5.11(a) to fig. 5.11(d) show the volume fraction estimation by applying the first four MLPs in table 5.1 to the scanned data of `sub_002`. We also append the SANDI's parameter maps to fig. 5.11. We shall explain the comparison with SANDI in section 5.6.

In addition to the volume fraction maps, the area fraction maps are shown in fig. 5.12. Because the sum of soma and neurite area fractions is unity, we only present the parameter maps for soma.

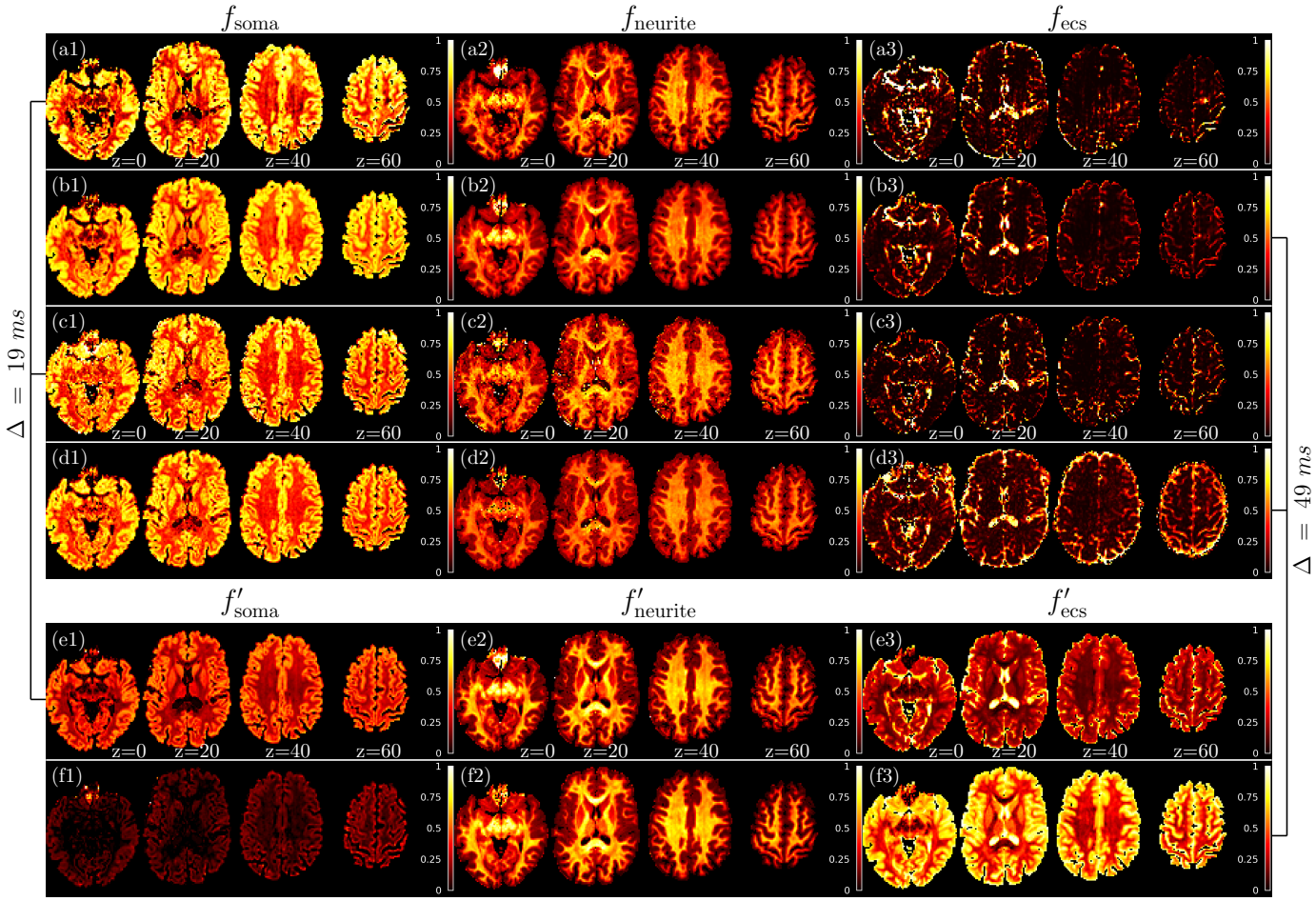


Figure 5.11: The comparison of volume and signal fractions. The first column is for soma volume fraction f_{soma} or soma signal fraction f'_{soma} ((e1) and (f1)), the second for neurite, and the third for ECS. Three rows, (a), (c), and (e), are for the short diffusion time ($\delta/\Delta = 8/19\ ms$). The remaining rows are for the long diffusion time ($\delta/\Delta = 8/49\ ms$). The first four rows, (a) (b), (c), and (d), are obtained by respectively applying `mlp_sig_vol_19`, `mlp_sig_vol_49`, `mlp_mk_vol_19`, and `mlp_mk_vol_49`, to the experimental data from `sub_002`. The last two rows, (e) and (f), show the signal fractions obtained by fitting the SANDI model to the direction-averaged signals from `sub_002`.

5.6 Comparison with SANDI

The state-of-the-art impermeable biophysical model for soma and neurite density imaging, SANDI [88], has similar assumptions to our brain voxel model. We both assume that the soma and neurite membranes are impermeable, somas are spherical, and ECS is a free diffusion space. The differences in modeling

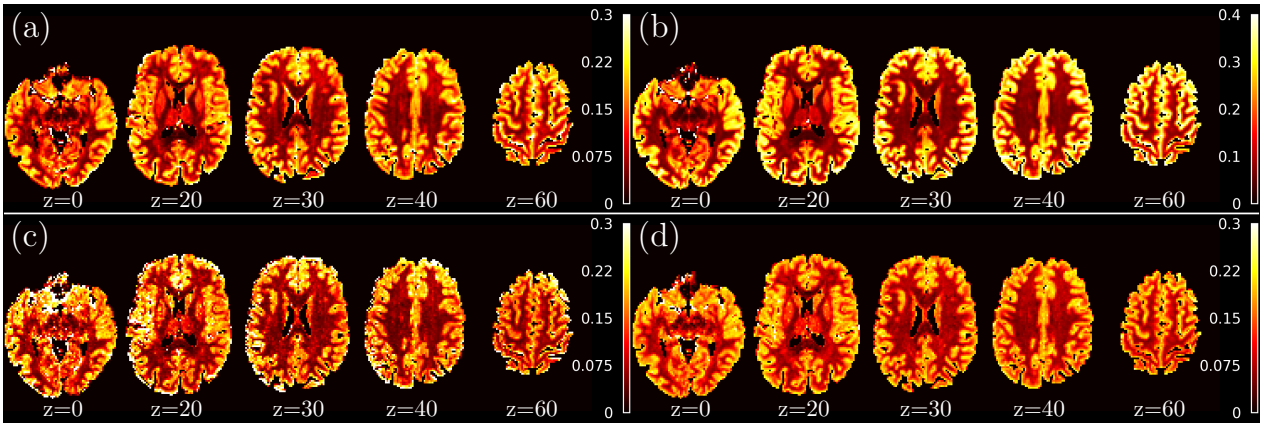


Figure 5.12: The maps of soma area fraction. The first column is for the short diffusion time ($\delta/\Delta = 8/19\ ms$). The second column is for the long diffusion time ($\delta/\Delta = 8/49\ ms$). The four subplots, (a) (b), (c), and (d), are obtained by respectively applying `mlp_sig_area_19`, `mlp_sig_area_49`, `mlp_mk_area_19`, and `mlp_mk_area_49`, to the experimental data from `sub_002`. Note that the color limits of (b) are different from others.

are:

1. SANDI considers the neurites as a set of randomly oriented sticks (long cylinders with zero radii), whereas our neurites have realistic radii and length, faithful undulation, and real dispersion;

2. SANDI utilizes disconnected soma and neurites because the water exchange between them is believed to be negligible when $t_d \leq 20\ ms$. In contrast, our neurites are connected to the soma, forming a continuous space.

3. Our model assumes the transverse relaxation (T2) is homogeneous in all compartments within a brain voxel so that the signal normalization can cancel the transverse relaxation effect (see eqs. (3.11) and (3.20)), whereas SANDI considers the T2 values of the intra- and extra-cellular compartments within a brain voxel are different.

Due to the impermeability assumption and item 2, the validity regime of SANDI is $t_d \leq 20\ ms$. So the short diffusion time $\delta/\Delta = 8/19\ ms$ is in the regime, while the long diffusion time is not.

The significant difference is in the dMRI signal generation. The direction-averaged signal of SANDI has an explicit expression [88]

$$\overline{E}_{\text{SANDI}} = f'_{\text{soma}} e^{-D'_s b} + f'_{\text{neurite}} \sqrt{\frac{\pi}{4bD_{\text{in}}}} erf\left(\sqrt{bD_{\text{in}}}\right) + f'_{\text{ecs}} e^{-D_{\text{ecs}} b}, \tag{5.12}$$

where D_{in} is the longitudinal apparent diffusion coefficient in the sticks; D_{ecs} is the apparent diffusion coefficient in ECS; f'_{soma}, f'_{neurite}, and f'_{ecs} are the signal fractions for soma, neurite, and ECS, respectively. The soma term $e^{-D'_s b}$ is derived, under the GPA, by Murday and Cotts on a spherical liquid particle with radius being r_s and liquid self-diffusion coefficient being D_s [90]. The quantity D'_s, a function of δ, Δ, r_s, and D_s, has an explicit formulation (eq. (1.41)). We refer to the soma term as the "MC equation" following the terminology adopted by Balinov et al. [91]. For eq. (5.12) to hold, it is necessary to add at least three additional assumptions:

1. the Gaussian phase assumption and the MC equation hold under the experimental condition;
2. the signal from a spherical "apparent" soma can approximate the volume-weighted average signal from a group of somas;
3. the stick power-law scaling, which is the neurite signal term $\sqrt{\frac{\pi}{4bD_{\text{in}}}} erf(\sqrt{bD_{\text{in}}})$, is valid.

With these assumptions, the direction-averaged signal of SANDI ($\overline{E}_{\text{SANDI}}$) is an explicit function whose variables are δ, Δ, b, f'_{soma}, f'_{neurite}, f'_{ecs}, r_s, D_s, D_{in}, and D_{ecs}. Among them, δ, Δ, and b are known experimental parameters, D_s is fixed to be $3 \times 10^{-3}\ \mu m^2/\mu s$ [88]. The remaining six variables are the microstructure parameters to be estimated. Similar to the volume fractions, the sum of the three signal fractions is one. Although the signal fractions often characterize soma and neurite density, they differ from the volume fractions due to the compartmental difference in the T2 values.

SANDI estimates the six parameters of biophysical interest by fitting eq. (5.12) to the direction-averaged signals. We adopt the open-source AMICO framework[2] to perform the fitting [93]. In contrast to our method, the SANDI model has an analytical signal expression, which allows one to recompute the direction-averaged signals by substituting SANDI's estimations into the signal formula eq. (5.12).

Since our neuron models are more realistic than SANDI's and the signal simulation requires fewer biophysical assumptions, it is worth comparing SANDI with the MLPs trained in our framework.

5.6.1 Fitting SANDI to simulated signals

We start with fitting the SANDI model to the simulated signals from artificial brain voxels. The direction-averaged signals in the test set without noise injection serve as the simulated signals. They are linearly sampled at 64 gradient

[2] `https://github.com/daducci/AMICO/wiki/Fitting-the-SANDI-model`

intensities from 0 to 290 mT/m and free of transverse relaxation, myelin water, and CSF contamination, which is an ideal condition for fitting the SANDI model. Since the relaxation effects are not included, the signal fractions should be the volume fractions in this ideal case.

AMICO's optimization needs to predetermine some parameter distributions, namely the ranges of r_s, D_{in}, and D_{ecs}. We use the ranges suggested in [88] and guarantee dense samplings. The training data distributions are 50 values of D_{in} linearly spaced in the range $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 50 values of D_{ecs} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 50 values of r_s linearly spaced in $[1, 12]\ \mu m$. In addition, the L1 and L2 regularization terms are 0 and 0.005, respectively.

To demonstrate that the SANDI model has been properly fitted to the simulated signals, we substitute the estimated parameters into SANDI's signal expression (eq. (5.12)) to recompute the direction-averaged signals. We call them the recomputed signals. We randomly pick several artificial brain voxels and compare the simulated and recomputed signals. Figure 5.13 shows the signal comparison for the short diffusion time ($\delta/\Delta = 8/19\ ms$). We stress that the y-axes of fig. 5.13 are in logarithmic scale, which magnifies the signal differences. Sections D.3 and D.4 includes the signal comparison for the long diffusion time and linear-scale plots where the differences are almost invisible. Because the differences between the two groups of signals are minimal, we think the signal fitting is satisfactory.

We then assess whether the estimated signal fractions are related to the volume fractions. As mentioned above, the signal fractions should equal the volume fractions because the simulated signals are free of noise and relaxation. We annotate the SANDI's estimations and the ground-truth values in the upper right corner of each subplot of fig. 5.13. Even though the SANDI model fits the simulated signals well, some estimated parameters are not correctly related to the ground truth.

We plot the absolute and relative errors of SANDI's signal fraction estimation in fig. 5.14 to better demonstrate the discrepancy. The errors of MLPs' volume fraction estimation are shown in fig. 5.6(b) and fig. 5.7(b).

In addition to the errors, at the short diffusion time, the R^2 scores for soma, neurite, and ECS signal fraction estimations are -2.88, 0.89, and -0.54, respectively. They are -1.74, 0.87, and -0.18 at the long diffusion time. It is worth stressing that SANDI correctly predicts the neurite fraction at the two diffusion times. However, the soma and ECS fraction estimations do not correlate with ground-truth values.

We need to note that the fitting of the SANDI model and the parameter distributions required by AMICO affect the final performance of SANDI. We only report the best results that we got using the AMICO framework. A better fit-

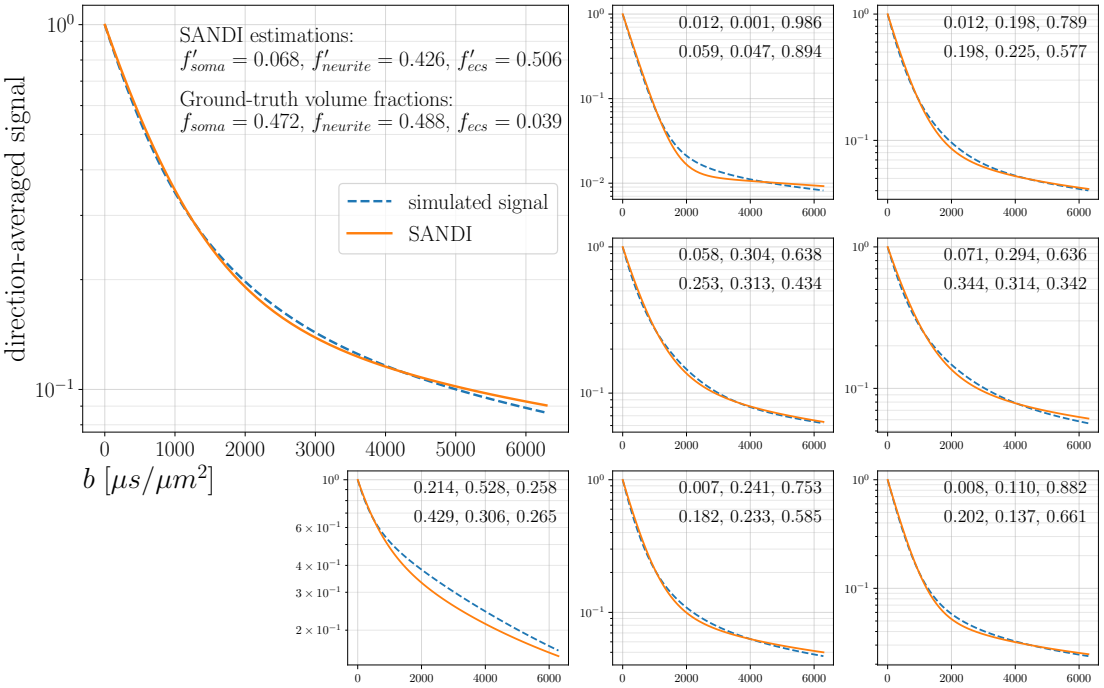


Figure 5.13: The comparison between the simulated (blue dashed line) and recomputed signals (solid line) of eight randomly picked artificial brain voxels. The recomputed signals are obtained by substituting SANDI's estimations into eq. (5.12). The diffusion time is $\delta/\Delta = 8/19\ ms$. The first subplot represents the meaning of the six numbers annotated in the upper right corner of each subplot. The numbers in the first row are the estimated soma, neurite, and ECS signal fractions. The numbers below are the ground-truth volume fractions.

ting method and a more realistic data distribution could help SANDI yield better results.

5.6.2 Fitting SANDI to experimental signals

Next, we fit the SANDI model to the experimental signals in MGH CDMD. The parameter distributions are 10 values of D_{in} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 10 values of D_{ecs} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 10 values of r_s linearly spaced in $[1, 12]\ \mu m$; L1 and L2 regularization terms are 0 and 0.005, respectively. For comparison with MLPs, we present the signal fraction maps for the second subject `sub_002` in the last two rows of fig. 5.11.

We reduce the number of samples to 10 in each range because the SANDI fitting with 50 samples yields unreasonable parameter maps. There are significant gaps between the measured and recomputed signals. We show the parameter maps obtained using the previous distribution with 50 samples in section D.5.

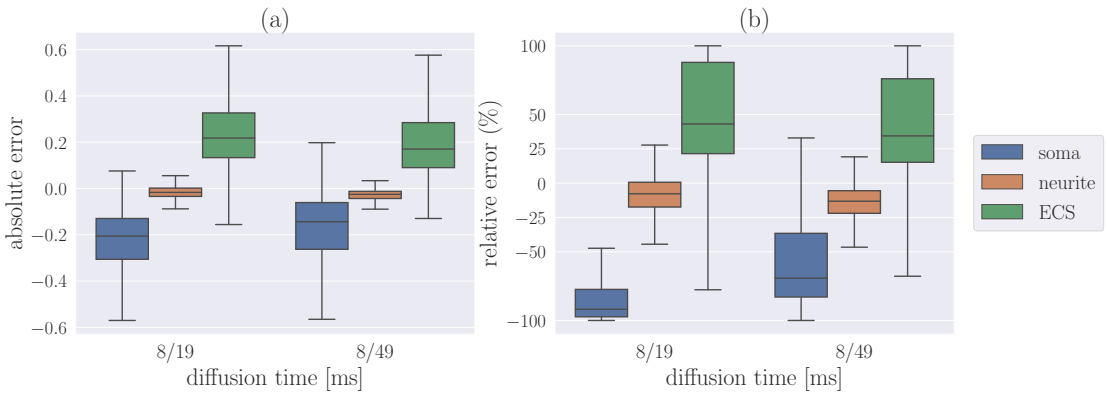


Figure 5.14: The box plots summarizing the distributions of SANDI's absolute and relative errors on the synthetic test set. A box plot denotes the median, interquartile range, and 1.5 times the interquartile range by the center line, hinges, and whiskers. Outliers are ignored, and the relative errors are capped at $\pm 100\%$ for clarity. For the short diffusion time, the R^2 scores of soma, neurite, and ECS signal fraction estimations are -2.88, 0.89, and -0.54, respectively. They are -1.74, 0.87, and -0.18 for the long diffusion time. **(a)** the absolute error distributions for SANDI's signal fraction estimation. **(b)** the relative error distributions for SANDI's signal fraction estimation.

We emphasize that the SANDI model is supposed to be invalid at the long diffusion time ($\delta/\Delta = 8/49\ ms$) because the cellular membrane permeability and the water exchange between soma and neurites may cause considerable effects. However, SANDI still gives reasonable estimations for the neurite signal fractions at the long diffusion time because the neurite maps are similar at the two diffusion times. We quantitatively study the dependence on diffusion time in the next section.

5.6.3 Independence of diffusion time

We present the voxelwise joint distributions for estimated soma and neurite volume fractions in fig. 5.15. All brain white and gray matter voxels of `sub_002` are included. We also plot the distribution for SANDI's estimation for comparison. The soma volume fractions estimated by `mlp_sig_vol_19` and `mlp_sig_vol_49` lie roughly in the identity line. The neurite volume fractions are mostly independent of diffusion time. Even though SANDI is believed to be inapplicable to a diffusion time greater than $20\ ms$, the neurite signal fraction estimations are consistent at the two diffusion times.

The above results persist in other subjects. We present the joint distributions of two additional subjects in sections D.6 and D.7.

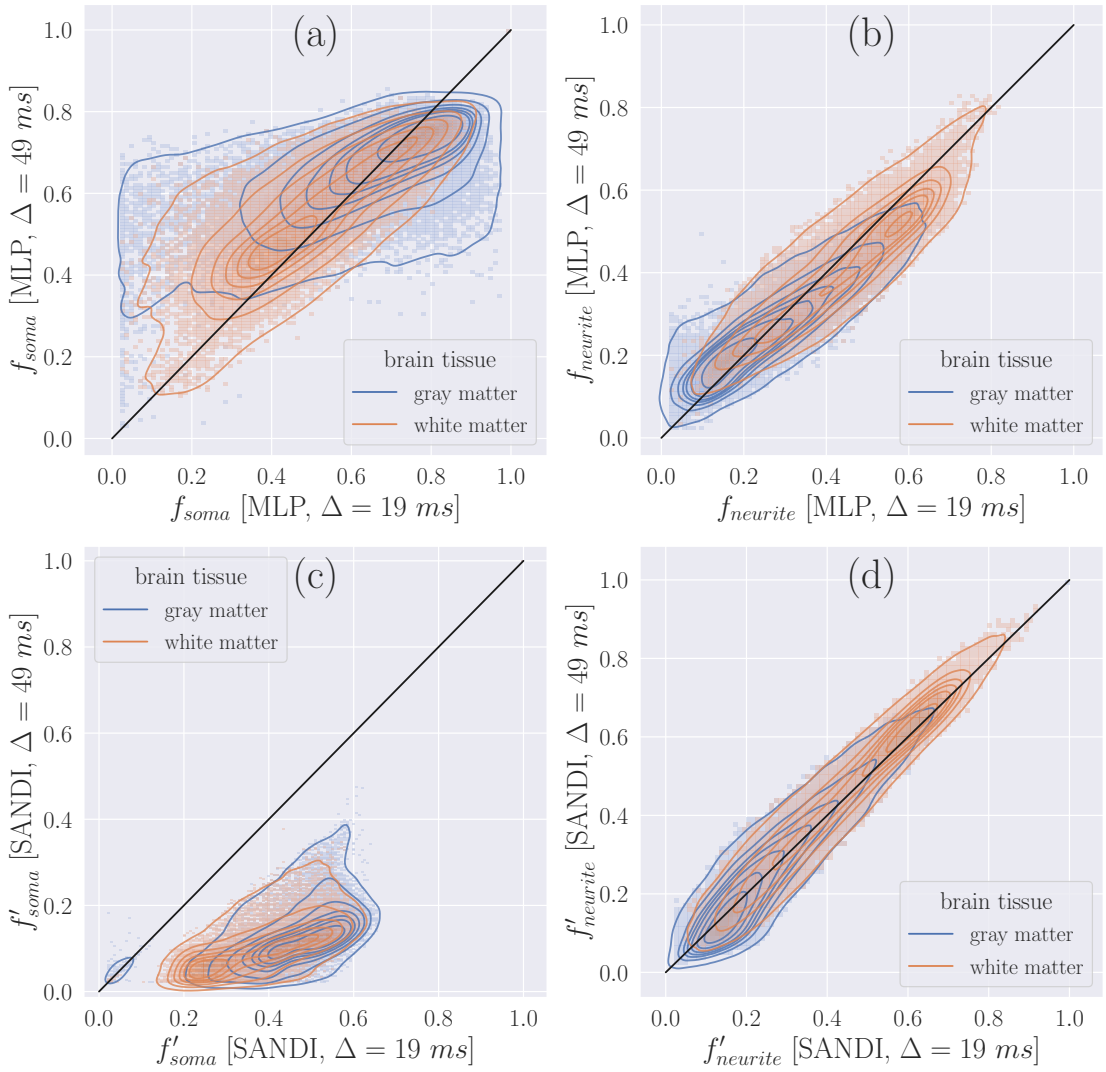


Figure 5.15: The voxelwise joint distribution of estimated fractions at two diffusion times. All brain white and gray matter voxels of `sub_002` are included. The x- and y-axes represent the estimated fractions at $\delta/\Delta = 8/19\ ms$ and $\delta/\Delta = 8/49\ ms$, respectively. The black lines are the identity lines. **(a)** the distribution of the soma volume fractions estimated by `mlp_sig_vol_19` and `mlp_sig_vol_49`. **(b)** the distribution of the neurite volume fractions estimated by `mlp_sig_vol_19` and `mlp_sig_vol_49`. **(c)** the distribution of the soma signal fractions estimated by the SANDI model at the two diffusion times. **(d)** the distribution of the neurite signal fractions estimated by the SANDI model at the two diffusion times.

5.7 Discussion

This chapter proposes a novel framework that employs realistic neuron modeling and diffusion MRI simulation to replace simplified biophysical models and analytical intracellular signal expressions. Effects arising from, for instance, neurite undulation or water exchange between soma and neurites, which are hard to include in biophysical models, are seamlessly incorporated into our simulations. Consequently, our framework can achieve higher geometric modeling accuracy while requiring fewer biophysical assumptions. Moreover, the microstructure parameters do not need to be explicitly expressed in the signal, allowing us to explore new contrasts, such as area fractions.

The proposed framework has two cornerstones. The first one is the realistic neuron meshes in NeuronSet that we generated using *swc2mesh*. These neuron meshes allow us both to obtain neuroanatomical parameters and to simulate intracellular signals.

The second cornerstone is the numerical matrix formalism implemented in the SpinDoctor toolbox [44, 47]. In section 3.3, we achieved a ten-fold speedup by optimizing the eigendecomposition algorithm and leveraging the GPU computation. Now it takes three weeks instead of thirty weeks to complete all simulations on 1213 neurons in NeuronSet. The simulation accuracy has been validated in section 5.3.1.

Nonetheless, neuron meshes are merely the building blocks of an artificial brain voxel. To construct a voxel phantom, we need to densely pack the neurons so that ECS has a reasonable volume fraction and neurons must not intersect each other. The neuron packing is still a highly challenging open question. We circumvented this problem by modeling the ECS as a free diffusion space and assuming impermeable cell membranes. This enables us to compute the signals from a voxel without explicitly constructing the numerical phantom. However, the simplified ECS model and impermeability assumption introduce errors that are the major limitations of our method.

These two cornerstones bring advanced modeling capabilities allowing us to build a synthetic dataset for supervised learning. Figure 5.2 summarizes the generation of the dataset consisting of 1.4 million artificial brain voxels and the subsequent model training. Each voxel corresponds to thousands of directional dMRI signals and over forty microstructure parameters. The dataset contains rich information that helps investigate the relationships between dMRI signals and microstructure. Besides, the dataset is also a good reference for validating biophysical models.

In addition to the size, the dataset quality can significantly affect the final

performance of ML models[3], especially for supervised learning. We maintain the quality of neuron meshes and the simulation accuracy to ensure the dMRI signals and the ground-truth microstructure parameters are accurate. Moreover, the training data distribution should be relevant to the distribution we encounter in real-world applications [226, 240]. This requirement has also been recognized by the diffusion MRI community [241]. Palombo et al. [159] provide some reference distributions for neuron modeling based on over 3500 neuron skeletons from NeuroMorpho.Org, among which over 1000 are human cells. Because we also used the cells stored in NeuroMorpho.Org, our neuron meshes naturally follow all reference distributions.

To adapt the trained MLPs to various experimental acquisition settings, we proposed an interpolation method by imposing three boundary conditions on fourth-order B-spline interpolators. As long as enough measurements are given, the interpolation method can effectively mitigate the fluctuation caused by vanilla splines and give satisfactory approximations to the measured signals. One can freely sample signals within maximum gradient intensity or compute signal features using the proposed interpolation method.

Next, we focus on approximating the mappings from the dMRI signals to the volume or area fractions using MLPs. Our preliminary work suggests that such mappings exist and may be high-dimensional [116]. The MLP hyperparameters are determined on validation sets. Two types of features are studied: the direction-averaged signals and the five markers. We note the most salient points of the trained MLPs here:

1. it is possible to predict the volume or area fractions by solely feeding direction-averaged signals to MLPs;

2. the five markers form a set of concise features that can effectively predict the volume and area fractions;

3. the dependence of the volume fraction estimation on diffusion time is minimal while using the MGH CDMD dataset;

4. it is possible to investigate new microstructure parameters, such as area fractions, using the proposed framework.

We discuss them in the following sections.

5.7.1 Synthetic data experiments

We conducted a comprehensive performance evaluation of the eight MLPs on synthetic test sets by reporting the test losses (L1-norm), R^2 scores, and the

[3] Garbage in, garbage out.

absolute and relative errors in section 5.4.4. The average L1-loss between predictions and ground truth for the volume fraction estimation is around 0.01, as shown in table 5.1. The absolute errors of more than fifty percent of predictions are less than 0.025, and most relative errors are inferior to 10% (see fig. 5.6(b) and fig. 5.7(b)). Among the three compartments, ECS volume fraction estimation is better than that of soma and neurite. Most importantly, MLPs' volume fraction predictions are almost unbiased, and their R^2 scores for the three compartments are greater than 0.9 (see table 5.1). For area fraction estimation, the R^2 scores of the four corresponding MLPs are around 0.6. The soma area fraction estimation suffers large relative errors (see fig. 5.7(a)). Nonetheless, the predictions are still unbiased. These results indicate that the MLPs are good estimators in the synthetic dataset.

However, we cannot correlate SANDI's estimations to the ground-truth soma and ECS volume fractions. Admittedly, the model fitting is crucial for SANDI. We believe the fitting in section 5.6.1 is good enough for two reasons. First, the signals recomputed by the SANDI's signal expression eq. (5.12) can explain well the original simulated signals (see fig. 5.13). Second, the fitting correctly predicts the ground-truth neurite volume fraction with relatively small errors (see fig. 5.14). We reiterate that the signal fractions should equal the volume fractions in the synthetic test set because the simulated signals are free of transverse relaxation.

Figure 5.13 and fig. 5.14 show that significant errors occur in SANDI's estimation of soma and ECS signal fractions. This is not surprising because the soma term $f'_{\text{soma}}e^{-D'_s b}$ in eq. (5.12) has the same form as the ECS term $f'_{\text{ecs}}e^{-D_{\text{ecs}}b}$, causing an indeterminacy problem. We recall that D'_s (eq. (1.41)) is a function of δ, Δ, soma radius r_{is}, and the intra-soma diffusion coefficient D_{is}. Therefore, based on the sum of the two exponentials ($f'_1 e^{-D_1 b} + f'_2 e^{-D_2 b}$), there is no way to tell which exponential belongs to soma and which belongs to ECS by solely changing the bvalue. The MLPs' performance in test sets suggests they do not suffer from such a problem. However, we stress that the performance of SANDI depends on the model fitting and the parameter distributions. A better fitting method and a more realistic data distribution could help SANDI yield better results.

5.7.2 In vivo parameter maps

We obtained the parameter maps by applying the MLPs to every brain voxel of a subject in MGH CDMD. Figure 5.10 demonstrates the evolution of a parameter map during the model training. The improvement in estimation ability indicates the generalization of the MLPs to both the synthetic test set and the experimental data.

Figure 5.11 presents the estimated volume fractions. There are no significant differences between the two types of features. However, rows (c) and (d)

obtained using the five markers are noisier than (a) and (b). This is because the computation of the five markers, which requires the derivatives of the direction-averaged signals, is sensitive to noise. In addition, the MLPs, `mlp_mk_vol_19` and `mlp_mk_vol_49`, that take the five markers as input are trained with noiseless data. So they are not robust to noise.

In fig. 5.11, the maps of f_{soma} correctly highlight the gray matter and the cerebral nuclei. In contrast, the maps of f_{neurite} are prominent in the white matter, especially the WM tracts located at the corpus callosum, the corona radiata, and the brain stem. The neurite volume fraction is mostly in line with SANDI's neurite signal fraction, except for the fact that f'_{neurite} is slightly higher than f_{neurite}.

The average f_{ecs} values are 0.12 in the cerebrum, which is inferior to the typical ECS volume fraction (0.2) obtained using the real-time iontophoresis method [162]. The underestimation may be because the simplified ECS model is invalid in GM and WM. Indeed, the estimated width of the ECS ranges from 10 nm to 64 nm [161], and the ECS tortuosity is about 1.6 [162]. The water diffusion in such a complex medium could significantly differ from the free diffusion. The ECS modeling is the main limitation of the proposed method.

The soma area fraction is a new contrast obtained using the proposed framework. The maps of a_{soma} can also properly highlight the gray matter and the cerebral nuclei. The two MLPs that take the 64 signals as input produce cleaner parameter maps. However, the area fraction estimation is inconsistent at the two diffusion times. For example, fig. 5.12(b) has higher fraction estimations in gray matter than fig. 5.12(a). The moderate performance is also reflected by the low R^2 scores in the test set. But the area fraction maps are satisfactory as a proof of concept to demonstrate the potential of the proposed framework for investigating new microstructure parameters.

The above results qualitatively demonstrate that the MLPs trained in the proposed framework can yield encouraging estimations. We further validate the parameter maps by investigating the consistency across diffusion times.

5.7.3 Independence of diffusion time

Due to the lack of real-world ground truth (for example, the actual soma volume fractions in some brain voxels of a subject), validating parameter maps remains largely qualitative. Given this limitation, the community has begun to seek consistency across acquisition parameters, sequences, and scanners [242–244], instead of qualitative visual assessment. In our case, we focus on the dependence of the volume fractions on diffusion times. Indeed, microstructure imaging aims to infer the objective tissue properties based on dMRI signals. We want the estimated tissue properties to be independent of how they are measured.

Figure 5.11 demonstrates that the parameter maps given by the MLPs are mostly consistent between the short and long diffusion times. For a more quantitative comparison, we plot the voxelwise joint distribution of the estimated soma and neurite volume fractions at the two diffusion times in fig. 5.15. If the estimation is consistent, the scatter points should lie around the identity line. It turns out that the estimated f_{neurite} is mainly invariant to the change of diffusion time. The soma volume fraction f_{soma} locates fairly near the identity line with a relatively broad variance. We stress that the cell membrane permeability may not be ignored when $t_d \geq 20\ ms$ [187]. Nonetheless, the MLPs can still give consistent and reasonable estimations at the long diffusion time, suggesting that the effect of cell membrane water exchange on volume fraction estimation is still minor when $t_d \sim 46\ ms$.

Interestingly, the SANDI model can also give consistent neurite signal fraction estimations even though it is believed to be inapplicable when $t_d \geq 20\ ms$. The consistent results agree with the good performance of SANDI in the synthetic test set for neurite signal fraction estimation (see fig. 5.14). In addition, the distributions of f_{neurite} and f'_{neurite} are quite similar (see fig. 5.15(b) and fig. 5.15(d)).

Recently, evidence from experiments and simulations shows that the stick power-law scaling is valid in WM and GM [85, 86, 99, 227]. In gray matter, however, the aggregation of neurites and somas modifies the concavity of the direction-averaged signals. Even though the stick power-law scaling is not apparent in the GM [99], the direction-averaged intra-neurite signals still follow the power-law scaling, as demonstrated in our previous work [116]. The consistency of the neurite signal fraction estimation at two diffusion times further supports the stick power-law scaling.

The advantage of the proposed supervised learning framework manifests itself by giving consistent estimations at two diffusion times for both neurite and soma volume fractions. It took years for the community to reach a consensus about the stick power-law scaling, while it only took hundreds of epochs for an MLP to give a similar neurite fraction estimation.

5.7.4 Limitations

The proposed method is primarily limited by the geometric modeling capability, especially by the modeling of the ECS. The ECS is a complex medium whose tortuosity is about 1.6 [162]. The thickness of the ECS ranges from 10 nm to 64 nm [161], and the ECS volume fraction is about 20% [162]. Geometrically modeling such a complex medium is highly challenging. To circumvent this, we used a simplified brain voxel model by assuming that cell membranes are impermeable and the ECS is a free diffusion space. In addition, we chose a very high diffusion coefficient in the ECS because the ECS includes CSF in our model. To guarantee

that the trained MLPs give reasonable ECS volume fraction estimation in, for example, the ventricles, we chose the high diffusion coefficient, as did [81, 245]. Instead of keeping D_{ecs} fixed, a better way could be to draw D_{ecs} from a distribution. However, since the free diffusion space may not be able to model ECS, the best solution is to achieve the geometric modeling of ECS. Some work on this issue has yielded exciting results. For example, Lee et al. [246, 247] achieve the geometric modeling of myelin sheaths in a corpus callosum sample of a mouse brain. Their method requires 3D electron microscopy images of the sample and the semi-automatic segmentation of the images. With this method, the shape and position of axons are well preserved, which could help us to construct a more realistic model for the ECS. However, the method does not currently work automatically, thus limiting its widespread use.

Another factor affecting the realism of the brain voxel model is the number and type of neurons in a brain voxel. According to a recent brain histology study [12], a gray matter brain voxel of $\sim 1\ mm^3$ comprises 57,216 cells and hundreds of millions of neurites. We are unable to model tens of thousands of neurons geometrically. Moreover, in the human brain, there are similar numbers of neurons and glial cells [248]. Including more glial cells in the neuron mesh dataset would allow for a more realistic distribution of cell types.

The average thickness of the human cerebral cortex is about $2.5\ mm$ [249]. A brain voxel of $1\ mm^3$ can span multiple layers of the cerebral cortex, thus containing a variety of neurons. In order to simulate the composition of real brain voxels as well as possible, we randomly combined the neurons in NeuronSet regardless of the brain region where the neuron was originally located. The primary motivation for the random combination is to diversify the shape of the neurons in artificial brain voxels. In addition, we put hundreds of neurons in an artificial brain voxel to simulate the massive signal-averaging effect (eq. (3.20)) in real brain voxels. However, this could create unrealistic combinations of neurons. A future improvement is to combine neurons from the same brain region (see the regional distribution of cells in NeuronSet in fig. 2.17). This requires more neurons in each brain region.

Furthermore, we can better optimize several hyperparameters of the proposed method. They include the distribution of the ECS volume fraction, the number of direction-averaged signals, and the maximum number of neurons in an artificial brain voxel. We chose them through a limited number of trials in this study. A grid search could help find better hyperparameters.

Unlike biophysical models that could apply to a range of diffusion times, the MLPs are trained for a particular time profile only. If one wants to employ a new diffusion time, the dMRI simulations must be rerun on all neurons. The simulations could be highly time-consuming if one adopts a simulation method

other than the numerical matrix formalism. Because the eigendecomposition has already been obtained, the time overhead of computing new signals by the numerical matrix formalism is reduced but still considerable.

5.7.5 Future perspectives

The proposed framework can readily help in many aspects of diffusion MRI. First, the synthetic dataset can be used to validate biophysical models. Second, we have over forty rotationally-invariant microstructure parameters. This paper focuses only on five. It is worth investigating other parameters, for example, the average soma radius. Third, the direction-averaged signals obtained with several acquisition protocols (e.g., various diffusion times or different sequences) can be fed into an MLP together to achieve joint estimation. This could help reduce indeterminacy. The MLP training is the same for diffusion-encoding sequences other than PGSE. However, some biophysical models' signal expressions are derived only for PGSE sequences. If one adopts another type of sequence, it is necessary to derive and validate new signal expressions. The derivation of analytical signal expressions is not trivial at all.

Four extensions to the framework are foreseeable. First of all, we could remove the impermeability assumption. Agdestein et al. [50] have extended the numerical matrix formalism to include permeable compartments. We can solve the complete BT equation system, eqs. (3.1) to (3.5), with permeable membranes using numerical matrix formalism. Note that the computational optimization made in this thesis also applies to the permeable case. The main challenge is the ECS mesh generation. We have been able to wrap a neuron mesh with a thin envelope to achieve ECS modeling with reasonable volume fractions. An example can be found in section A.3. However, dense neuron packing is still inevitable to get a brain voxel phantom. Some recent advances in computer graphics provide promising approaches to this problem. The basic idea is to allow some flexibility in the neurons and squeeze them into a cube. Second, we could remove the assumption about homogeneous transverse relaxation and employ different compartmental T2 values. The simulation with transverse relaxation is straightforward because T2 relaxation just introduces some exponential multipliers to the computation, as shown by eq. (3.14). Third, we can generate more cellular meshes based on myriad neuron tracing data stored in NeuroMorpho.Org. Finally, the neuron meshes also contain orientation information. Estimating orientation-dependent microstructure using the synthetic dataset can also be expected.

5.8 Summary

We proposed a novel framework leveraging a highly-efficient simulator, modern computer graphics algorithms, and supervised learning methods to infer the brain microstructure in vivo using diffusion MRI. The fundamental tools of the framework have been made publicly available. We demonstrated that the framework helps to approximate the underlying mappings from diffusion MRI signals to several microstructure parameters. As proof of concept, we presented how to estimate volume and area fractions using the direction-averaged signals or the five signal markers via training MLPs on a synthetic dataset generated by the framework. Qualitatively, the MLPs gave promising parametric maps. Quantitatively, the estimated volume fractions depended less on the diffusion time than a state-of-the-art method. Although the obtained parameter maps still require further validation, we believe the proposed framework can substantially help achieve quantitative microstructure imaging and promote broader adoption of diffusion MRI simulation.

Chapter 6

Conclusion

The ultimate goal of this thesis is to facilitate simulation-driven brain microstructure imaging with diffusion MRI. We made notable contributions to neuron modeling and diffusion MRI simulation to achieve this goal. The simulation-driven supervised learning framework presented in chapter 5 is a promising prototype. This chapter presents a summary of the results and future perspectives.

We developed an open-source neuron mesh generator, *swc2mesh*, that can automatically and robustly convert valuable neuron tracing data to realistic neuron meshes. We have carefully designed the generator to maintain a good balance between the mesh quality and size. A neuron mesh dataset, NeuronSet, which contains 1213 simulation-ready cellular meshes and their neuroanatomical measurements, has been built using the mesh generator. The number of meshes in itself demonstrates the capability of the neuron mesh generator. We believe the NeuronSet is beneficial for diffusion MRI simulation and for other neuroscience research like neuroanatomy. A foreseeable application of *swc2mesh* is to study the contributions of individual neuronal components to diffusion MRI signals because the neuron mesh generator provides an easy way to construct meshes for individual neuronal components like a single dendrite. Furthermore, it is possible to edit the tabular neuron tracing data (SWC files) to precisely modify the neuron shape. This allows us to study, for example, the effect of decreased neurite radii or lengths on dMRI signals.

Diffusion MRI simulation is another piece of the puzzle. We adopted the numerical FE-based matrix formalism method. Integrating matrix formalism with a finite element method (FEM) brings significant advantages in terms of computational efficiency. We further optimized the numerical MF by speeding up the eigendecomposition algorithm and leveraging GPU computation. A tenfold speedup is achieved. Calculations that previously took an hour now take only six minutes. In addition, with similar precision, the optimized numerical matrix formalism is **20** times faster than FEM and **65** times faster than a GPU-

based Monte-Carlo method. By performing simulations on the realistic neuron meshes, we have investigated the effect of soma-neurite water exchange, the neurite power-law scaling, and the relationship between soma size and signals. The combination of the neuron mesh generator and the ultra-fast numerical matrix formalism provides powerful modeling and computational capabilities. We believe these capabilities can bring more insight into diffusion MRI.

The Fourier potential method is a new method based on the potential theory that provides a Fourier-type representation of the diffusion MRI signal. This method has been derived theoretically and implemented numerically. We validated the convergence of the method and showed that the error behavior is in line with our error analysis. However, the method is currently of theoretical interest only. Future work is required to generalize the method to 3D and go beyond the narrow pulse assumption.

Finally, we proposed a simulation-driven framework with high geometrical modeling accuracy for brain microstructure imaging. By harnessing the aforementioned powerful modeling and computational capabilities, we constructed a synthetic dataset that contains the dMRI signals and microstructure parameters of 1.4 million artificial brain voxels. We demonstrated that the dataset helps approximate the underlying mappings from dMRI signals to volume and area fractions. Unlike existing methods, our method does not require the microstructure parameters to be explicitly included in a signal expression. Therefore, one can freely investigate any parameters that can be measured on neuron meshes. However, determining which microstructure parameters can be appropriately estimated requires profound physical insight.

We presented eight exemplary MLPs trained on the synthetic dataset for estimating volume and area fractions. In the synthetic test set, the MLPs have lower estimation errors than the SANDI model. In the experimental dataset, the MLPs and SANDI give similar results for neurite fraction estimation. By comparing the estimations at two diffusion times, we showed that the MLPs' estimations depend less on the diffusion time. Although further validation is definitely required, we believe the proposed framework is a satisfactory prototype for simulation-driven brain microstructure imaging.

The main limitation of this method concerns the ECS modeling. Neurons are tightly intertwined in real brain tissue. Building a geometrical model for ECS requires densely packing a large number of neurons in a tiny cube to achieve a reasonable ECS volume fraction. Besides, these neurons cannot intersect with each other. We have not solved this highly challenging geometrical modeling problem yet. Consequently, we still assume the ECS is a free diffusion space, which could introduce biases. Some recent advances in computer graphics provide promising approaches to this problem. These approaches based on linear

elasticity theory can perfectly prevent the intersection. Therefore, we can allow some flexibility in the neurons and squeeze them into a cube. However, this requires further work. If we obtain the geometrical models for ECS, the extension of the proposed method to include permeable membranes is foreseeable.

To conclude, this thesis contributed to neuron modeling, diffusion MRI simulation, and brain microstructure imaging. We hope that these contributions will bring the significance of geometric modeling and simulation to the attention of the community and lead to the design of more advanced microstructure imaging methods to explore the mysteries of the human brain.

Appendices

Appendix A

Supplementary Material to Chapter 2

This appendix contains three sections. Section A.1 provides an exhaustive list of neuroanatomical parameters that could be relevant to dMRI. Section A.2 gives the complete list of all neuron meshes in Neuron Module. Some neuroanatomical measurements of neurons in Neuron Module are also included. Section A.3 shows an example of a neuron wrapped by an ECS compartment. To guarantee a realistic volume fraction of ECS ($\sim 20\%$), the mesh of ECS must be extremely narrow, as shown in fig. A.2.

A.1 List of neuroanatomical parameters

In this section, we list various neuroanatomical measurements from a single neuron. We first present some primitive parameters that can be directly measured from neuron skeletons or meshes:

1. neuron volume, V_{neuron};
2. neuron area, A_{neuron};
3. soma radius, r_{soma};
4. total neurite length measured by L-measure, $L^{\text{lm}}_{\text{neurite}}$;
5. number of stems, N_{stems};
6. maximum stem Euclidean distance, $D_{\text{max,euc}}$;
7. maximum stem path distance, $D_{\text{max,path}}$;

8. average stem Euclidean distance, $\overline{D}_{\text{euc}}$;
9. average stem path distance, $\overline{D}_{\text{path}}$.

We then list some secondary neuroanatomical parameters that be calculated from the above parameters:

10. soma volume, $V_{\text{soma}} = \frac{4}{3}\pi r_{\text{soma}}^3$;
11. soma area, $A_{\text{soma}} = 4\pi r_{\text{soma}}^2$;
12. neurite volume, $V_{\text{neurite}} = V_{\text{neuron}} - V_{\text{soma}}$;
13. neurite area, $A_{\text{neurite}} = A_{\text{neuron}} - A_{\text{soma}}$;
14. total neurite length, $L_{\text{neurite}} = L_{\text{neurite}}^{\text{lm}} - N_{\text{stems}} \times r_{\text{soma}}$;
15. average neurite radius (based on neurite volume): $r_{\text{neurite}}^v = \sqrt{\frac{V_{\text{neurite}}}{\pi L_{\text{neurite}}}}$;
16. average neurite radius (based on neurite area): $r_{\text{neurite}}^a = \frac{A_{\text{neurite}}}{2\pi L_{\text{neurite}}}$;
17. neuron volume-area ratio, $V_{\text{neuron}}/A_{\text{neuron}}$;
18. neurite volume-area ratio, $V_{\text{neurite}}/A_{\text{neurite}}$;
19. soma volume-area ratio, $V_{\text{soma}}/A_{\text{soma}} = r_{\text{soma}}/3$;
20. neuronal soma volume fraction, $V_{\text{soma}}/V_{\text{neuron}}$;
21. neuronal soma area fraction, $A_{\text{soma}}/A_{\text{neuron}}$;
22. neuronal neurite volume fraction, $V_{\text{neurite}}/V_{\text{neuron}}$;
23. neuronal neurite area fraction, $A_{\text{neurite}}/A_{\text{neuron}}$;
24. average stem length, $\overline{L}_{\text{stem}} = L_{\text{neurite}}/N_{\text{stem}}$;
25. neurite irregularity, $\frac{r_{\text{neurite}}^v - r_{\text{neurite}}^a}{r_{\text{neurite}}^v + r_{\text{neurite}}^a}$ (= 0, if neurites are cylindrical);
26. total stem Euclidean distance, $D_{\text{euc}} = \overline{D}_{\text{euc}} \times N_{\text{stems}}$;
27. total stem path distance, $D_{\text{path}} = \overline{D}_{\text{path}} \times N_{\text{stems}}$;
28. maximum contraction, $D_{\text{max,euc}}/D_{\text{max,path}}$;

29. average contraction, $\overline{D}_{\text{euc}}/\overline{D}_{\text{path}}$;

30. ratio between soma and neuron volume-area ratios, $\dfrac{V_{\text{soma}}/A_{\text{soma}}}{V_{\text{neuron}}/A_{\text{neuron}}}$;

31. ratio between neurite and neuron volume-area ratios, $\dfrac{V_{\text{neurite}}/A_{\text{neurite}}}{V_{\text{neuron}}/A_{\text{neuron}}}$;

32. ratio between soma and neurite volume-area ratios, $\dfrac{V_{\text{soma}}/A_{\text{soma}}}{V_{\text{neurite}}/A_{\text{neurite}}}$.

The list is not exhaustive. We refer to the L-measure website (`http://cng.gmu.edu:8080/Lm/help/index.htm`) for the definitions of over 40 neuroanatomical parameters. However, those parameters not listed here might be less relevant to diffusion MRI.

A.2 List of neurons in Neuron Module

Table A.1 lists the names, IDs, and other metadata of the group of 36 pyramidal neurons and the group of 29 spindle neurons in Neuron Module. Table A.2 shows the neuron mesh sizes and some neuroanatomical measurements for the neurons. The neuron models and the measurement data are from [110] and [119].

Table A.1: Metadata of the 65 neurons in Neuron Module.

Index	Cell name	Cell ID	Brain region	Cell type
1	02a_pyramidal2aFI	NMO_01042	fronto-insula	pyramidal
2	02b_pyramidal1aACC	NMO_01043	anterior cingulate	pyramidal
3	02b_pyramidal1aFI	NMO_01044	fronto-insula	pyramidal
4	03a_pyramidal9aFI	NMO_01045	fronto-insula	pyramidal
5	03a_spindle2aFI	NMO_01078	fronto-insula	von economo
6	03a_spindle6aFI	NMO_01079	fronto-insula	von economo
7	03b_pyramidal2aACC	NMO_01046	anterior cingulate	pyramidal
8	03b_pyramidal3aACC	NMO_01047	anterior cingulate	pyramidal
9	03b_pyramidal3aFI	NMO_01048	fronto-insula	pyramidal
10	03b_pyramidal4aFI	NMO_01049	fronto-insula	pyramidal
11	03b_pyramidal9aFI	NMO_01050	fronto-insula	pyramidal
12	03b_spindle4aACC	NMO_01080	anterior cingulate	von economo
13	03b_spindle5aACC	NMO_01081	anterior cingulate	von economo
14	03b_spindle6aACC	NMO_01082	anterior cingulate	von economo

Table A.1 continued from previous page

Index	Cell name	Cell ID	Brain region	Cell type
15	03b_spindle7aACC	NMO_01083	anterior cingulate	von economo
16	04a_pyramidal4aACC	NMO_01051	anterior cingulate	pyramidal
17	04a_pyramidal5aACC	NMO_01052	anterior cingulate	pyramidal
18	04b_pyramidal5aFI	NMO_01053	fronto-insula	pyramidal
19	04b_pyramidal6aACC	NMO_01054	anterior cingulate	pyramidal
20	04b_pyramidal6aFI	NMO_01055	fronto-insula	pyramidal
21	04b_pyramidal7aACC	NMO_01056	anterior cingulate	pyramidal
22	04b_spindle3aFI	NMO_01084	fronto-insula	von economo
23	05a_pyramidal10aACC	NMO_01057	anterior cingulate	pyramidal
24	05a_pyramidal8aACC	NMO_01058	anterior cingulate	pyramidal
25	05b_pyramidal7aFI	NMO_01059	fronto-insula	pyramidal
26	05b_pyramidal8aFI	NMO_01060	fronto-insula	pyramidal
27	05b_pyramidal9aACC	NMO_01061	anterior cingulate	pyramidal
28	05b_spindle5aFI	NMO_01085	fronto-insula	von economo
29	06a_pyramidal11aACC	NMO_01062	anterior cingulate	pyramidal
30	06b_pyramidal10aFI	NMO_01063	fronto-insula	pyramidal
31	06b_pyramidal12aACC	NMO_01064	anterior cingulate	pyramidal
32	06b_spindle8aACC	NMO_01086	anterior cingulate	von economo
33	07a_pyramidal13aACC	NMO_01065	anterior cingulate	pyramidal
34	07b_pyramidal14aACC	NMO_01066	anterior cingulate	pyramidal
35	07b_spindle9aACC	NMO_01087	anterior cingulate	von economo
36	08a_spindle13aACC	NMO_01088	anterior cingulate	von economo
37	08o_pyramidal11aFI	NMO_01067	fronto-insula	pyramidal
38	09o_spindle7aFI	NMO_01089	fronto-insula	von economo
39	09o_spindle8aFI	NMO_01090	fronto-insula	von economo
40	10a_pyramidal15aACC	NMO_01068	anterior cingulate	pyramidal
41	10a_spindle18aACC	NMO_01091	anterior cingulate	von economo
42	11a_pyramidal16aACC	NMO_01069	anterior cingulate	pyramidal
43	11o_pyramidal12aFI	NMO_01070	fronto-insula	pyramidal
44	12a_spindle19aACC	NMO_01092	anterior cingulate	von economo
45	12o_spindle9aFI	NMO_01093	fronto-insula	von economo
46	13o_spindle10aFI	NMO_01094	fronto-insula	von economo
47	15o_spindle12aFI	NMO_01095	fronto-insula	von economo
48	16o_spindle13aFI	NMO_01096	fronto-insula	von economo
49	17o_pyramidal13aFI	NMO_01071	fronto-insula	pyramidal
50	18o_pyramidal14aFI	NMO_01072	fronto-insula	pyramidal
51	19o_spindle14aFI	NMO_01097	fronto-insula	von economo

Table A.1 continued from previous page

Index	Cell name	Cell ID	Brain region	Cell type
52	20o_pyramidal15aFI	NMO_01073	fronto-insula	pyramidal
53	21o_spindle15aFI	NMO_01098	fronto-insula	von economo
54	22o_pyramidal16aFI	NMO_01074	fronto-insula	pyramidal
55	23o_spindle16aFI	NMO_01099	fronto-insula	von economo
56	24o_pyramidal17aFI	NMO_01075	fronto-insula	pyramidal
57	25o_pyramidal18aFI	NMO_01076	fronto-insula	pyramidal
58	25o_spindle17aFI	NMO_01100	fronto-insula	von economo
59	26o_spindle18aFI	NMO_01101	fronto-insula	von economo
60	27o_spindle19aFI	NMO_01102	fronto-insula	von economo
61	28o_spindle20aFI	NMO_01103	fronto-insula	von economo
62	28o_spindle21aFI	NMO_01104	fronto-insula	von economo
63	29o_spindle22aFI	NMO_01105	fronto-insula	von economo
64	30o_spindle23aFI	NMO_01106	fronto-insula	von economo
65	31o_pyramidal19aFI	NMO_01077	fronto-insula	pyramidal

Table A.2: Mesh size and neuroanatomical measurements for all neurons in Neuron Module.

Index	# vertices	Mean neurite diameter [μm]	Length [μm]	Soma vol. [μm^3]	Neuron vol. [μm^3]
1	119156	1.27	404.85	19701.05	25639.61
2	45216	1.58	363.08	9065.56	11579.71
3	105384	1.62	381.56	22475.52	29804.96
4	81530	2.15	532.30	22557.27	30189.28
5	38202	1.74	387.16	13406.27	17684.23
6	44000	1.66	501.47	33458.19	37812.72
7	28183	1.48	189.29	2977.21	4487.44
8	27607	1.14	188.45	6005.06	6891.04
9	151362	1.84	496.35	32510.62	46154.08
10	96177	1.33	414.70	35253.85	39324.87
11	66162	1.92	430.06	15263.14	20532.57
12	17370	1.43	336.33	3098.39	4070.19
13	26345	1.49	221.52	11925.78	13242.53
14	26792	1.33	398.36	4027.74	6058.67
15	21618	1.18	369.51	4982.41	6076.52
16	150897	1.52	705.96	5684.55	13637.33
17	89256	1.86	410.59	15010.43	24648.35

Table A.2 continued from previous page

Index	# vertices	avg. neurite diameter [μm]	Height [μm]	Soma vol. [μm^3]	Neuron vol. [μm^3]
18	95784	1.78	480.13	10312.87	17184.26
19	87195	1.41	465.88	3129.97	7497.12
20	90482	1.56	310.21	14718.05	21708.21
21	622553	1.3	610.42	17060.60	28552.49
22	51265	2.71	391.14	22569.99	28404.13
23	201506	1.74	281.20	16604.06	21826.41
24	139975	1.29	430.37	24709.77	29778.79
25	208203	2.18	281.02	25720.11	32731.05
26	124350	1.66	361.45	32527.06	44679.46
27	366659	1.56	650.60	23948.05	40014.54
28	22457	2.35	381.88	15383.08	18190.63
29	319574	1.46	437.60	17222.02	29995.02
30	106808	1.92	365.18	43127.81	52179.53
31	277718	1.52	324.94	17181.33	24931.32
32	15163	1.92	342.21	18237.49	19462.92
33	155854	1.37	325.73	6254.53	8738.01
34	309789	1.67	350.40	16053.07	22772.96
35	54952	1.75	437.87	21344.83	27307.48
36	46293	1.74	814.45	9911.07	14113.32
37	419651	1.91	421.68	11512.38	24326.94
38	38992	2.90	472.87	22052.10	27905.89
39	60755	2.05	376.73	11923.76	15189.32
40	56184	1.40	341.48	8522.11	10960.84
41	25797	1.57	457.90	5895.17	7219.28
42	222732	1.27	486.31	8807.01	12263.84
43	380293	1.91	369.34	70786.62	79516.92
44	31841	2.05	431.22	12178.08	15618.67
45	29320	3.41	305.31	29983.79	36678.18
46	43081	2.69	516.92	39866.55	46022.15
47	101548	3.60	604.57	53192.65	79170.43
48	18266	2.17	364.66	17467.88	18888.13
49	326989	1.89	340.77	11004.30	21167.19
50	338453	1.74	288.41	69851.56	78999.20
51	25786	2.18	232.21	10507.15	12905.43
52	247116	1.82	383.18	22344.32	27667.19
53	28822	2.36	286.33	17567.69	29466.53

Table A.2 continued from previous page

Index	# vertices	avg. neurite diameter [μm]	Height [μm]	Soma vol. [μm^3]	Neuron vol. [μm^3]
54	389878	1.94	585.35	18776.05	29441.43
55	30073	1.67	420.05	10429.13	13482.93
56	245058	2.04	371.99	40986.40	47377.09
57	71209	1.80	364.05	18587.13	23572.15
58	52919	1.79	358.70	7897.44	13563.26
59	36239	2.27	442.65	52911.93	56084.44
60	50807	1.73	275.08	20640.14	25423.96
61	56036	3.00	520.69	35442.59	51267.07
62	17581	2.62	298.57	35579.06	37783.31
63	18414	3.52	402.84	62928.22	83279.12
64	26357	3.05	322.56	24006.79	28303.55
65	619390	2.26	303.55	65950.80	86376.72

A.3 A neuron mesh wrapped by a thin ECS

We present an example of a neuron[1] wrapped in a thin envelope to achieve ECS modeling (see fig. A.1 and fig. A.2). The ECS volume fraction is 0.31.

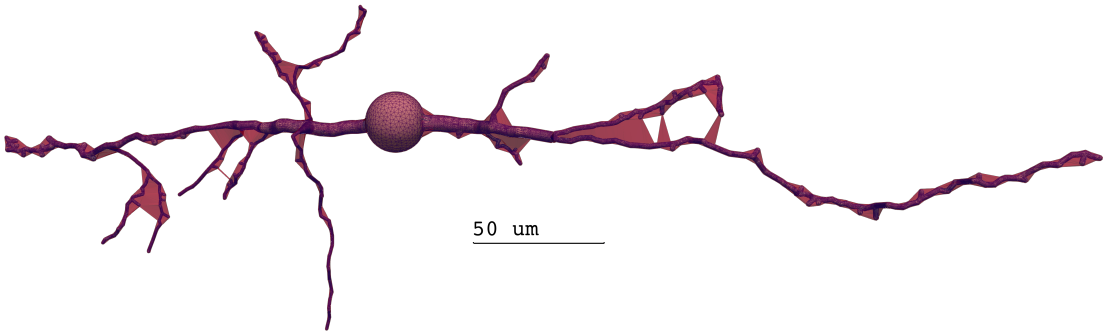


Figure A.1: A neuron wrapped in an ECS. The neuron is represented by a solid white mesh inside a transparent pink envelope, the ECS, whose volume fraction is 0.31.

[1] NeuroMorpho.Org ID of the neuron is NMO_01078.

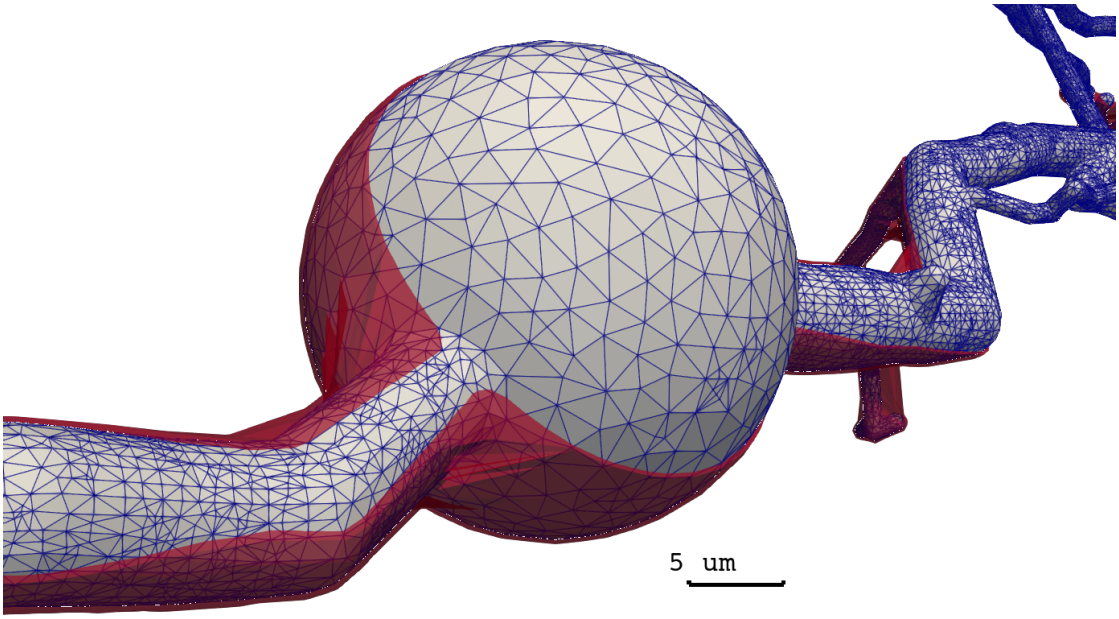


Figure A.2: Zoomed-in of the above neuron. A solid white mesh represents the neuron. The red envelope represents the ECS. Half of the ECS is removed to show the neuron inside.

Appendix B

Supplementary Material to Chapter 3

B.1 Proof of eq. (3.18)

We give the proof of eq. (3.18). When the gradient intensity is zero, the BT equation degenerates to the diffusion equation:

$$\frac{\partial}{\partial t}\varphi_i(\boldsymbol{x},t) = D_i\nabla\cdot\nabla\varphi_i(\boldsymbol{x},t), \qquad \boldsymbol{x}\in\Omega_i,\ t\in[0,TE], \tag{B.1}$$

$$\varphi_i(\boldsymbol{x},0) = 1, \qquad \boldsymbol{x}\in\Omega_i, \tag{B.2}$$

$$D_i\nabla\varphi_i(\boldsymbol{x},t)\cdot\boldsymbol{n}_i(\boldsymbol{x}) = 0, \qquad \boldsymbol{x}\in\partial\Omega_i,\ t\in[0,TE]. \tag{B.3}$$

We integrate eq. (B.1) over the domain Ω_i to get

$$\frac{d}{dt}\int_{\Omega_i}\varphi_i(\boldsymbol{x},t)d\boldsymbol{x} = \int_{\Omega_i} D_i\nabla\cdot\nabla\varphi_i(\boldsymbol{x},t)d\boldsymbol{x}. \tag{B.4}$$

The divergence theorem allows us to compute the right-hand side of eq. (B.4):

$$\int_{\Omega_i} D_i\nabla\cdot\nabla\varphi_i(\boldsymbol{x},t)d\boldsymbol{x} = \int_{\partial\Omega_i} D_i\nabla\varphi_i(\boldsymbol{x},t)\cdot\boldsymbol{n}_i(\boldsymbol{x})ds_{\boldsymbol{x}} = 0. \tag{B.5}$$

So the integral $\int_{\Omega_i}\varphi_i(\boldsymbol{x},t)d\boldsymbol{x}$ is a constant. Finally, we have

$$\int_{\Omega_i}\varphi_i(\boldsymbol{x},TE;g=0)d\boldsymbol{x} = \int_{\Omega_i}\varphi_i(\boldsymbol{x},0)d\boldsymbol{x} = \int_{\Omega_i} 1 d\boldsymbol{x} = V_i. \tag{B.6}$$

B.2 Monte-Carlo simulation

In section 3.3.3, we utilize a GPU-accelerated Monte-Carlo method implemented in the *disimpy* [65] package. Two simulation parameters control the precision of the Monte-Carlo method: the number of random walkers N_{walkers} and the number of time steps N_{t}.

To compare the simulation efficiency of FEM, numerical MF, and the Monte-Carlo method, we need to tune the simulation parameters so that the maximum relative error of each method is close to 2%. For the Monte-Carlo method, we reach a 2% relative error by gradually increasing N_{walkers} and N_{t}. Table B.1 shows six combinations of parameters. In section 3.3.3, we report the execution times of the fourth row because the maximum relative error is the closest to 2%.

Table B.1: Execution times and maximum relative errors of Monte-Carlo simulations with *disimpy*.

N_{walkers}	N_{t}	Max. relative error (%)	Preparation times (s)	Computation times (s)
10^3	10^3	68.1	1483.99	15.79
10^5	10^3	4.01	1468.33	25.44
10^5	10^4	4.32	1475.76	198.1
10^5	5×10^4	**2.36**	1474.41	902.55
10^5	8×10^4	3.99	1497.92	1414.77
5×10^5	5×10^4	0.99	1531.33	1852.63

B.3 Additional neuron simulations

We show some simulation results on other neuron meshes in our collection. In fig. B.1, we compare the diffusion MRI signals due to two different dendrite branches, one from *04b_spindle3aFI* and one from *03b_spindle7aACC*. The first branch has a single main trunk, whereas the second branch divides into two main trunks. We see at the higher b-value $b = 4000\ \mu s/\mu m^2$, at the longest diffusion time, the signal shape is more elongated (perpendicular to the main trunk direction) for the first dendrite branch than the second.

In fig. B.2, we show 3-dimensional simulation results of the spindle neuron *03a_spindle2aFI*. We plot in fig. B.2 the signal attenuations in 720 directions uniformly distributed in the unit sphere for $b = 1000\ \mu s/\mu m^2$ and $b = 4000\ \mu s/\mu m^2$. We see that the shape of the signal attenuations in these 720 directions is ellipsoid at the lower b-value, and the shape becomes more complicated at the larger

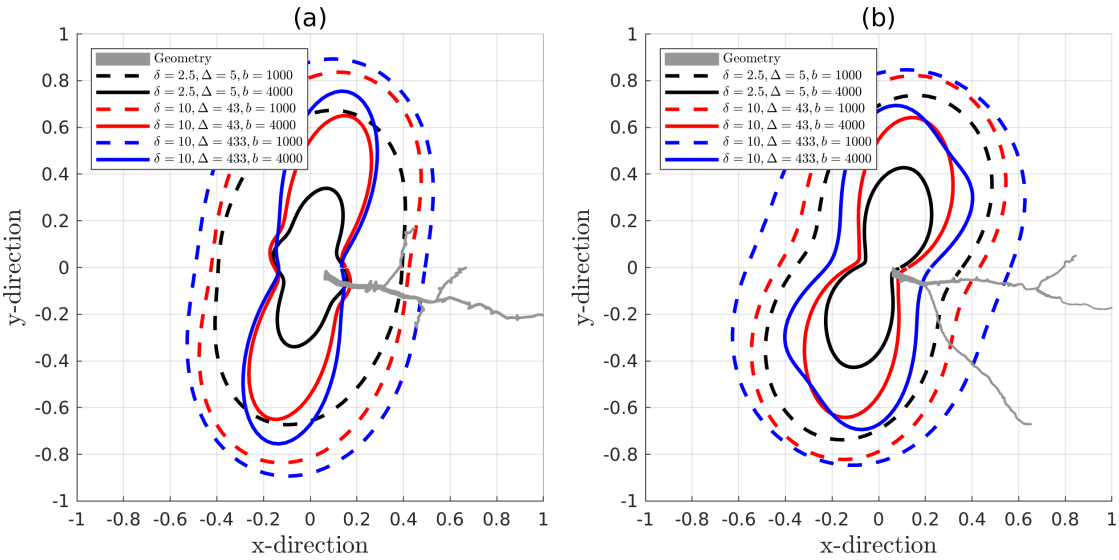


Figure B.1: The signal attenuations in 180 directions lying on the $x-y$ plane, uniformly distributed on a unit circle. The distance from each data point to the origin represents the magnitude of the signal attenuation. The simulation parameters are $rtol = 10^{-3}, atol = 10^{-5}, Htetgen = 0.5\ \mu m^3$. The diffusion coefficient is $2 \times 10^{-3}\ \mu m^2/\mu s$. (a) one dendrite branch of *04b_spindle3aFI* (volume mesh has 29854 vertices). (b) one dendrite branch of *03b_spindle7aACC* (volume mesh has 10145 vertices).

b-value. At $b = 4000\ \mu s/\mu m^2$, there is more signal attenuation at the shorter diffusion time than at the higher diffusion time.

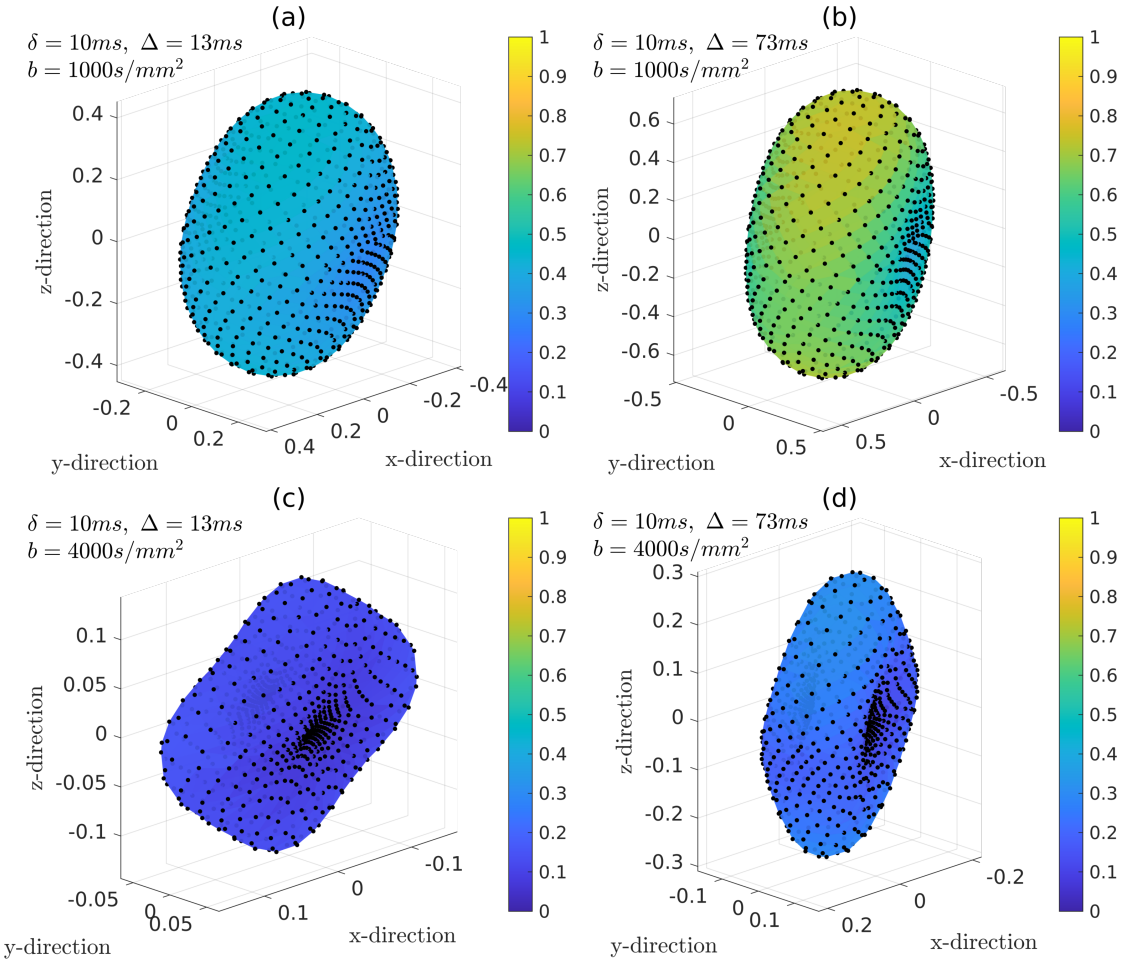


Figure B.2: The signal attenuations for the neuron *03a_spindle2aFI* in 720 directions uniformly distributed on a unit sphere. The color and the distance to the origin of each data point represent the magnitude of the signal attenuation. The simulation parameters are $rtol = 10^{-3}, atol = 10^{-5}, Htetgen = 0.5\ \mu m^3$. The diffusion coefficient is $2 \times 10^{-3}\ \mu m^2/\mu s$. **(a)** PGSE ($\delta/\Delta = 10/13\ ms$), $b = 1000\ \mu s/\mu m^2$. **(b)** PGSE ($\delta/\Delta = 10/73\ ms$), $b = 1000\ \mu s/\mu m^2$. **(c)** PGSE ($\delta/\Delta = 10/13\ ms$), $b = 4000\ \mu s/\mu m^2$. **(d)** PGSE ($\delta/\Delta = 10/73\ ms$), $b = 4000\ \mu s/\mu m^2$. The number of FE nodes for the neuron is 49833.

Appendix C

Numerical Implementation of FPM

C.1 Overview

Figure C.1 is a flowchart describing the Fourier Potential Method (FPM). The domain of definition of μ (and the intermediate variable $K_{long}[\mu]$) is on the boundary, $\boldsymbol{x}_0 \in \Gamma = \bigcup_{j=1}^{J} \partial\Omega_j$. The domain of definition of $\hat{f}$ is in the Fourier domain, $\boldsymbol{\nu} \in [-\nu_{max}, \nu_{max}]^2$.

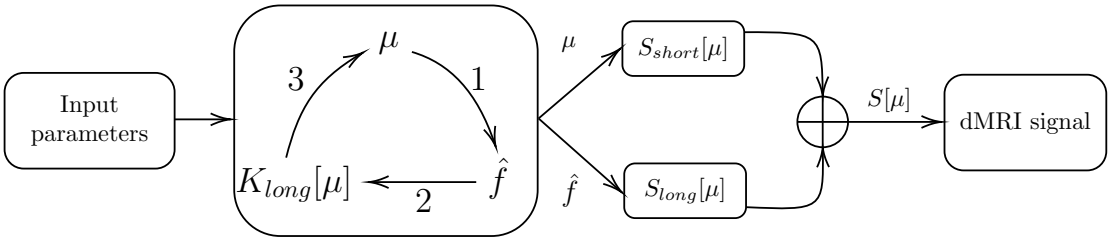


Figure C.1: Flowchart describing the workflow of the Fourier potential method.

The input parameters of our method include numerical descriptions of cellular membranes, diffusion MRI protocols, and simulation settings. All necessary inputs are listed in table C.1. The second block in fig. C.1 depicts the main procedure for solving eq. (4.27). Specifically, the loop in the second block contains three steps:

1. compute the Fourier coefficients $\hat{f}$ at time t:

$$\hat{f}(\boldsymbol{\nu}, t) = e^{-4\pi^2 D_0 \|\boldsymbol{\nu}\|^2 \Delta t} \hat{f}(\boldsymbol{\nu}, t - \Delta t) + \int_{t-\eta-\Delta t}^{t-\eta} \int_{\Gamma} e^{-4\pi^2 D_0 \|\boldsymbol{\nu}\|^2 (t-\tau)} \mu(\boldsymbol{y}, \tau) e^{-2\pi\imath \boldsymbol{\nu}\cdot\boldsymbol{y}} ds_{\boldsymbol{y}} d\tau. \quad \text{(C.1)}$$

2. compute the long time part $K_{long}[\mu]$ at time t:

$$K_{long}[\mu](\boldsymbol{x}_0, t) = D_0 \sum_{\boldsymbol{\nu}} 2\pi\imath\boldsymbol{\nu} \cdot \boldsymbol{n}\hat{f}(\boldsymbol{\nu}, t)e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0}d\nu^2. \tag{C.2}$$

3. compute the density μ at time t:

$$\mu(\boldsymbol{x}_0, t) = \frac{2\left(2\pi\imath\rho\mathbf{q} \cdot \boldsymbol{n}e^{-4\pi^2 D_0\|\mathbf{q}\|^2 t}e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{x}} - K_{long}[\mu](\boldsymbol{x}_0, t)\right)}{1 - \sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{x}_0)}. \tag{C.3}$$

We iterate the loop over all time steps until the density μ and the Fourier coefficients $\hat{f}$ are solved for the interval $[0, \Delta - \delta]$. Then we can compute $S_{short}[\mu]$ and $S_{long}[\mu]$ whose sum is the single layer potential. Finally, the diffusion MRI signal is obtained by integrating the single layer potential. The details of each block and a streamlined description of the numerical implementation are shown in the following sections.

C.2 Input parameters

Our method is numerically implemented in PyTorch [237]. The program needs input parameters describing cell membranes, diffusion MRI protocols, and simulation settings. Table C.1 lists the input parameters and their corresponding numerical counterparts.

To describe cellular membranes, we approximate them using numerous segments. Figure C.2 shows an example of approximating two irregular shapes by two polygons. The spatial discretization of cellular membranes allows us to perform numerical integration on the boundaries.

As a notation convention, we denote by $[a : i : b]$ an equispaced array starting at a, ending at b (included), and using i as the increment between elements. For example, $[0 : 1 : 5]$ is equivalent to $\{0, 1, 2, 3, 4, 5\}$.

Table C.1: Input parameters of FPM and their numerical counterparts.

Parameters		Explanation	Numerical counterparts
Cellular membranes $\bigcup_{j=1}^{J} \partial\Omega_j$	$\{\mathbf{x}_i\}_{i\in\{1,\dots,N\}}\ [\mu m]$	Cellular membranes/boundaries are discretized into N segments. $\mathbf{x}_i$ is the center of the i-th segment.	an array of size $2 \times N$
	$\{\Delta l_i\}_{i\in\{1,\dots,N\}}\ [\mu m]$	Δl_i is the length of the i-th segment.	an array of size $1 \times N$
	$\{\mathbf{n}_i\}_{i\in\{1,\dots,N\}}$	$\mathbf{n}_i$ is the out-pointing normal vector at the point $\mathbf{x}_i$.	an array of size $2 \times N$
	$\{\xi_i\}_{i\in\{1,\dots,N\}}\ [\mu m^{-1}]$	ξ_i is the curvature of the boundaries at the point $\mathbf{x}_i$.	an array of size $1 \times N$
	$\sum_{j=1}^{J} \lvert\Omega_j\rvert\ [\mu m^2]$	Total area of cells	a numeric variable
Diffusion MRI protocols	$D_0\ [\mu m^2/\mu s]$	Intrinsic diffusion coefficient in the neuron compartments	a numeric variable
	ρ	Initial magnetization	1
	$\Delta\ [\mu s]$	Inter-pulse duration	a numeric variable
	$\delta\ [\mu s]$	Pulse duration	a numeric variable
	$\gamma[rad \cdot \mu s^{-1} m T^{-1}]$	Gyromagnetic ratio	a numeric variable
	$\mathbf{g}\ [mT/m]$	Magnetic field gradient	a 2D vector
	$\mathbf{q}\ [\mu m^{-1}]$	$\mathbf{q} = \delta\gamma\mathbf{g}/2\pi$	a 2D vector
Simulation settings	$\Delta t\ [\mu s]$	Time step	a numeric variable
	$\eta\ [\mu s]$	Duration of the short time part of layer potentials	a numeric variable
	$\Delta\nu\ [\mu m^{-1}]$	Frequency step	a numeric variable
	$\nu_{max}\ [\mu m^{-1}]$	Maximum frequency in the truncated spectrum	a numeric variable

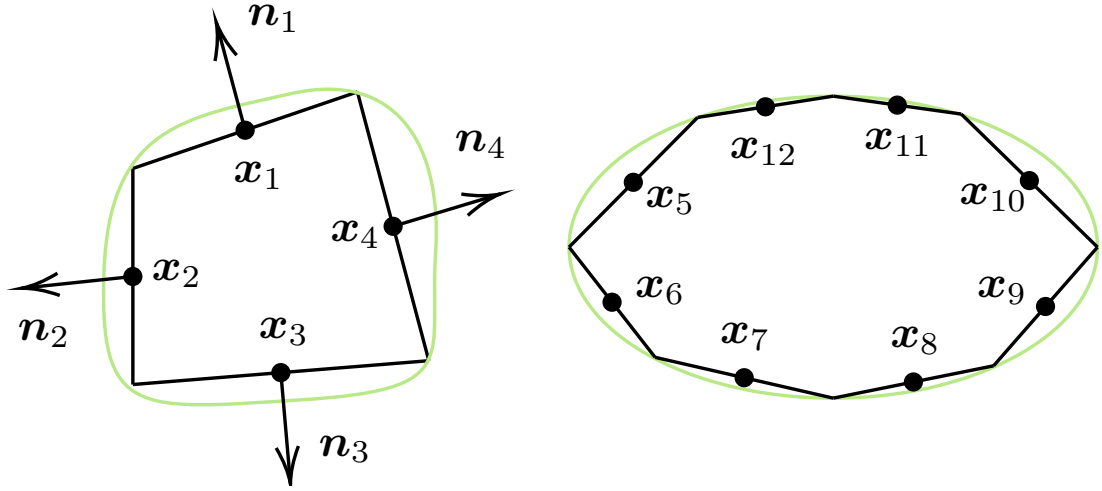


Figure C.2: An example of a computational domain consisting of two irregular shapes (light green curves) approximated by two polygons (black segments). For visualization purposes, the spatial discretization is coarse.

C.3 Iteration of μ, $\hat{f}$ and $K_{long}[\mu]$

Before the computation, we prepare Neumann data based on the boundary conditions. The complex-valued Neumann data is defined as

$$\mathcal{N}(\boldsymbol{x}_0, t) = 2\pi\imath\rho\mathbf{q}\cdot\boldsymbol{n}_{\boldsymbol{x}_0} e^{-4\pi^2 D_0\|\mathbf{q}\|^2 t} e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{x}_0}, \quad t\in[0,\Delta-\delta]. \tag{C.4}$$

Algorithm 2 is the pseudo-code for the numerical computation of Neumann data, which can be turned into a one-liner using array programming provided by Matlab, Numpy, or GPU programming languages.

Algorithm 2: Neumann data

Data: D_0, $\mathbf{q}$, $\{\boldsymbol{x}_i\}$, $\{\boldsymbol{n}_i\}$, $\mathcal{T}=[0:\Delta t:\Delta-\delta]$
Result: an array *NeuData* of size $\#\{\boldsymbol{x}_i\}\times\#\mathcal{T}$
begin
  """*Numerical implementation of eq. (C.4).*"""
  forall $(\boldsymbol{x}_i, t_m)\in\{\boldsymbol{x}_i\}\times\mathcal{T}$ **do** *# can be parallelized*
    $NeuData(\boldsymbol{x}_i, t_m) = 2\pi\imath\rho\mathbf{q}\cdot\boldsymbol{n}_i e^{-4\pi^2 D_0\|\mathbf{q}\|^2 t_m} e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{x}_i}$
  end
end

The time interval $t\in[0,\eta]$

We directly use the asymptotic expressions for the layer potentials to initialize μ. The Fourier coefficients $\hat{f}$ and $K_{long}[\mu]$ are initialized to zero:

$$\mu(\boldsymbol{x}_0, t) = \frac{2\mathcal{N}(\boldsymbol{x}_0, t)}{1-\sqrt{\frac{D_0 t}{\pi}}\xi(\boldsymbol{x}_0)}, \tag{C.5}$$

$$\hat{f}(\boldsymbol{\nu}, t) = 0, \tag{C.6}$$

$$K_{long}[\mu](\boldsymbol{x}_0, t) = 0. \tag{C.7}$$

Algorithm 3 shows the calculation of the density function $\mu(\boldsymbol{x}_0.t)$ when t is inferior to η.

Algorithm 3: Computation of the density μ for $t \in [0,\ \eta]$

Data: D_0, $NeuData$, $\{\boldsymbol{x}_i\}$, $\{\xi_i\}$, $\mathcal{T}_1 = [0 : \Delta t : \eta]$
Result: an array *mu* of size $\#\{\boldsymbol{x}_i\} \times \#\mathcal{T}$
Initialization: All elements in *mu* are set to be 0.
begin
  """*Numerical implementation of eq. (C.5).*"""
  forall $(\boldsymbol{x}_i, t_m) \in \{\boldsymbol{x}_i\} \times \mathcal{T}_1$ **do** *# can be parallelized*
    $mu(\boldsymbol{x}_i, t_m) = 2NeuData(\boldsymbol{x}_i, t_m)/(1 - \sqrt{\frac{D_0 t_m}{\pi}}\xi(\boldsymbol{x}_i))$
  end
end

The time interval $t \in [\eta + \Delta t, 2\eta]$

We compute $\hat{f}$ using the known analytical form of μ in $[0, \eta]$ from eq. (C.5), and update $K_{long}[\mu]$ and μ:

$$\hat{f}(\boldsymbol{\nu}, t) = e^{-4\pi^2 D_0 \|\boldsymbol{\nu}\|^2 \Delta t} \hat{f}(\boldsymbol{\nu}, t - \Delta t) + \hat{f}_{temp1}(\boldsymbol{\nu}, t), \tag{C.8}$$

$$K_{long}[\mu](\boldsymbol{x}_0, t) = D_0 \sum_{\boldsymbol{\nu}=-\boldsymbol{\nu}_{max}}^{\boldsymbol{\nu}_{max}} 2\pi\imath\boldsymbol{\nu} \cdot \boldsymbol{n} \hat{f}(\boldsymbol{\nu}, t) e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0} \Delta\nu^2, \tag{C.9}$$

$$\mu(\boldsymbol{x}_0, t) = 2\left(1 - \sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{x}_0)\right)^{-1} \left[\mathcal{N}(\boldsymbol{x}_0, t) - K_{long}[\mu](\boldsymbol{x}_0, t)\right]. \tag{C.10}$$

The Fourier update term $\hat{f}_{temp1}$ is

$$\begin{aligned}
\hat{f}_{temp1}(\boldsymbol{\nu}, t) &= \int_\Gamma \int_{t-\eta-\Delta t}^{t-\eta} e^{-4\pi^2 D_0 \|\boldsymbol{\nu}\|^2 (t-\tau)} e^{-2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{y}} \mu(\boldsymbol{y}, \tau) d\tau ds_{\boldsymbol{y}} \\
&= \int_\Gamma 4\pi\imath\rho\mathbf{q} \cdot \boldsymbol{n} e^{-2\pi\imath(\mathbf{q}+\boldsymbol{\nu})\cdot\boldsymbol{y}}\ p\ ds_{\boldsymbol{y}} \qquad \text{(C.11)} \\
p &= \int_{t-\eta-\Delta t}^{t-\eta} e^{-4\pi^2 D_0 \left[\|\boldsymbol{\nu}\|^2 (t-\tau) + \|\mathbf{q}\|^2 \tau\right]} \left(1 - \sqrt{\frac{D_0\tau}{\pi}}\xi(\boldsymbol{y})\right)^{-1} d\tau \qquad \text{(C.12)}
\end{aligned}$$

where the time integration follows the trapezoidal rule:

$$p = \begin{cases} -\frac{2\pi e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 t}}{D_0 \xi^2(\boldsymbol{y})} \left[\xi(\boldsymbol{y}) \sqrt{\frac{D_0}{\pi}} \left(\sqrt{t-\eta} - \sqrt{t-\eta-\Delta t} \right) + \right. \\ \left. \ln\left(\frac{1-\xi(\boldsymbol{y})\sqrt{\frac{D_0}{\pi}(t-\eta)}}{1-\xi(\boldsymbol{y})\sqrt{\frac{D_0}{\pi}(t-\eta-\Delta t)}} \right) \right], \|\boldsymbol{\nu}\| = \|\mathbf{q}\|; \\ e^{-4\pi^2 D_0 \left[\|\mathbf{q}\|^2 (t-\eta) + \|\boldsymbol{\nu}\|^2 \eta \right]} \left[\frac{1 + e^{4\pi^2 D_0 (\|\mathbf{q}\|^2 - \|\boldsymbol{\nu}\|^2)\Delta t} (4\pi^2 D_0 (\|\mathbf{q}\|^2 - \|\boldsymbol{\nu}\|^2)\Delta t - 1)}{\Delta t (4\pi^2 D_0 (\|\mathbf{q}\|^2 - \|\boldsymbol{\nu}\|^2))^2 \left(1 - \xi(\boldsymbol{y}) \sqrt{\frac{D_0}{\pi}(t-\eta-\Delta t)} \right)} + \right. \\ \left. \frac{e^{4\pi^2 D_0 (\|\mathbf{q}\|^2 - \|\boldsymbol{\nu}\|^2)\Delta t} - 4\pi^2 D_0 (\|\mathbf{q}\|^2 - \|\boldsymbol{\nu}\|^2)\Delta t - 1}{\Delta t (4\pi^2 D_0 (\|\mathbf{q}\|^2 - \|\boldsymbol{\nu}\|^2))^2 \left(1 - \xi(\boldsymbol{y}) \sqrt{\frac{D_0}{\pi}(t-\eta)} \right)} \right], \|\boldsymbol{\nu}\| \neq \|\mathbf{q}\|. \end{cases} \tag{C.13}$$

The evaluation of the two intermediate quantities, $\hat{f}_{temp1}$ and p, are shown in algorithms 4 and 5, respectively. Algorithm 6 shows the numerical implementation corresponding to the computation of eqs. (C.8) to (C.10) for the time interval $t \in [\eta + \Delta t, 2\eta]$.

Algorithm 4: Definition of `fhat_temp1`.

Function `fhat_temp1`($\boldsymbol{\nu}$, t, $\{\boldsymbol{x}_i\}$, $\{\Delta l_i\}$, $\{\boldsymbol{n}_i\}$, $\{\xi_i\}$, D_0, $\mathbf{q}$, η, Δt)**:**

"""*Numerical implementation of eq. (C.11).*"""

$result = 0$

 # spatial integration

forall $i \in [1:1:\#\{\boldsymbol{x}_i\}]$ **do** *# can be parallelized*

$res\_p = \mathrm{p}(\boldsymbol{\nu}, t, \mathbf{q}, \xi_i, \eta, \Delta t, D_0)$

$result = result + 4\pi\imath\rho\mathbf{q} \cdot \boldsymbol{n}_i e^{-2\pi\imath(\boldsymbol{\nu}+\mathbf{q})\cdot\boldsymbol{x}_i}\ res\_p\ \Delta l_i$

end

return *result*

Algorithm 5: Definition of p.

Function $\mathrm{p}(\boldsymbol{\nu}, t, \mathbf{q}, \xi, \eta, \Delta t, D_0)$**:**

"""*Numerical implementation of eq. (C.13).*"""

if $\|\boldsymbol{\nu}\| = \|\mathbf{q}\|$ **then**

$$result = -\frac{2\pi e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 t}}{D_0 \xi^2}\left[\xi\sqrt{\tfrac{D_0}{\pi}}\left(\sqrt{t-\eta}-\sqrt{t-\eta-\Delta t}\right)+ \ln\left(\frac{1-\xi\sqrt{\frac{D_0}{\pi}(t-\eta)}}{1-\xi\sqrt{\frac{D_0}{\pi}(t-\eta-\Delta t)}}\right)\right]$$

else

$$p1 = \frac{1+e^{4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t}(4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t-1)}{\Delta t(4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2))^2\left(1-\xi\sqrt{\frac{D_0}{\pi}(t-\eta-\Delta t)}\right)}$$

$$p2 = \frac{e^{4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t}-4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)\Delta t-1}{\Delta t(4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2))^2\left(1-\xi\sqrt{\frac{D_0}{\pi}(t-\eta)}\right)}$$

$$result = e^{-4\pi^2 D_0\left[\|\mathbf{q}\|^2(t-\eta)+\|\boldsymbol{\nu}\|^2\eta\right]}(p1+p2)$$

end

return *result*

Algorithm 6: Computation of $\hat{f}$, $K_{long}[\mu]$, and μ for $t \in [\eta + \Delta t,\ 2\eta]$

Data: $NeuData$, D_0, η, $\mathbf{q}$, Δt, $\Delta\nu$, $\{\boldsymbol{x}_i\}$, $\{\Delta l_i\}$, $\{\boldsymbol{n}_i\}$, $\{\xi_i\}$,
$\mathcal{V} = [-\nu_{max} : \Delta\nu : \nu_{max}]$, $\mathcal{T}_2 = [\eta + \Delta t : \Delta t : 2\eta]$

Result: two arrays *mu* and *K_long* whose sizes are $\#\{\boldsymbol{x}_i\} \times \#\mathcal{T}$, and an array *fhat* of size $\#\mathcal{V} \times \#\mathcal{V} \times \#\mathcal{T}$

Initialization: All elements in *K_long* and *fhat* are set to be 0.

begin

"""*Numerical computation of eqs. (C.8) to (C.10).*"""

foreach $t_m \in \mathcal{T}_2$ **do**

 # compute Fourier coefficient at time t_m

forall $\boldsymbol{\nu} \in \mathcal{V}^2$ **do** *# can be parallelized*

$temp = \texttt{fhat\_temp1}(\boldsymbol{\nu}, t_m, \{\boldsymbol{x}_i\}, \{\Delta l_i\}, \{\boldsymbol{n}_i\}, \{\xi_i\}, D_0, \mathbf{q}, \eta, \Delta t)$

$fhat(\boldsymbol{\nu}, t_m) = e^{-4\pi^2 D_0 \|\boldsymbol{\nu}\|^2 \Delta t} \cdot fhat(\boldsymbol{\nu}, t_m - \Delta t) + temp$

end

forall $i \in [1 : 1 : \#\{\boldsymbol{x}_i\}]$ **do** *# can be parallelized*

 # Inverse Discrete Fourier Transform (IDFT)

$K\_long(\boldsymbol{x}_i, t_m) = D_0 \sum_{\boldsymbol{\nu} \in \mathcal{V}^2} 2\pi\imath\boldsymbol{\nu} \cdot \boldsymbol{n}_i \; fhat(\boldsymbol{\nu}, t_m)\; e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_i} \Delta\nu^2$

$mu(\boldsymbol{x}_i, t_m) =$

$2\,[NeuData(\boldsymbol{x}_i, t_m) - K\_long(\boldsymbol{x}_i, t_m)] \,/\, \left(1 - \sqrt{\frac{D_0\eta}{\pi}}\xi_i\right)$

end

end

end

The time interval $t \in [2\eta + \Delta t, \Delta - \delta]$

In this interval, we iterate between $\hat{f}$ and μ (through the intermediate quanity $K_{long}[\mu]$). The required values to compute $\hat{f}$ at t are the density μ at $t - \eta$ and at $t - \eta - \Delta t$ which are obtained in previous steps:

$$\hat{f}(\boldsymbol{\nu}, t) = e^{-4\pi^2 D_0 \|\boldsymbol{\nu}\|^2 \Delta t} \hat{f}(\boldsymbol{\nu}, t - \Delta t) + \hat{f}_{temp2}(\boldsymbol{\nu}, t), \tag{C.14}$$

$$K_{long}[\mu](\boldsymbol{x}_0, t) = D_0 \sum_{\boldsymbol{\nu}=-\boldsymbol{\nu}_{max}}^{\boldsymbol{\nu}_{max}} 2\pi\imath\boldsymbol{\nu} \cdot \boldsymbol{n} \hat{f}(\boldsymbol{\nu}, t) e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_0} \Delta\nu^2, \tag{C.15}$$

$$\mu(\boldsymbol{x}_0, t) = 2\left(1 - \sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{x}_0)\right)^{-1} \left[\mathcal{N}(\boldsymbol{x}_0, t) - K_{long}[\mu](\boldsymbol{x}_0, \tau)\right]. \tag{C.16}$$

The Fourier update term $\hat{f}_{temp2}(\boldsymbol{\nu}, t)$ is:

$$\hat{f}_{temp2}(\boldsymbol{\nu}, t) = \int_\Gamma 2\left(1 - \sqrt{\frac{D_0\eta}{\pi}}\xi(\boldsymbol{y})\right)^{-1} e^{-2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{y}} \left(2\pi\imath\rho\mathbf{q}\cdot\boldsymbol{n}e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{y}}h_1 - h_2\right) ds_{\boldsymbol{y}}, \tag{C.17}$$

where the time integration h_1 in the first part has an analytical expression

$$\begin{aligned} h_1 &= \int_{t-\eta-\Delta t}^{t-\eta} e^{-4\pi^2 D_0(\|\mathbf{q}\|^2\tau + \|\boldsymbol{\nu}\|^2(t-\tau))} d\tau \\ &= \begin{cases} \Delta t \cdot e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2 t} & \|\mathbf{q}\| = \|\boldsymbol{\nu}\| \\ e^{-4\pi^2 D_0\left[\|\mathbf{q}\|^2(t-\eta) + \|\boldsymbol{\nu}\|^2\eta\right]} \frac{e^{4\pi^2 D_0(\|\mathbf{q}\|^2 - \|\boldsymbol{\nu}\|^2)\Delta t} - 1}{4\pi^2 D_0(\|\mathbf{q}\|^2 - \|\boldsymbol{\nu}\|^2)} & \|\mathbf{q}\| \neq \|\boldsymbol{\nu}\| \end{cases} \end{aligned} \tag{C.18}$$

and the time integration h_2 is done via the trapezoidal rule

$$\begin{aligned} h_2 &= \int_{t-\eta-\Delta t}^{t-\eta} K_{long}[\mu](\boldsymbol{y}, \tau) e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2(t-\tau)} d\tau \\ &= \begin{cases} \frac{\Delta t}{2}\left[K_{long}[\mu](\boldsymbol{y}, t-\eta) + K_{long}[\mu](\boldsymbol{y}, t-\eta-\Delta t)\right] & \|\boldsymbol{\nu}\| = 0 \\ \Big[\frac{1 - e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t}(4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t + 1)}{(4\pi^2 D_0\|\boldsymbol{\nu}\|^2)^2\Delta t} K_{long}[\mu](\boldsymbol{y}, t-\eta-\Delta t) + \\ \frac{e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t} + 4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t - 1}{(4\pi^2 D_0\|\boldsymbol{\nu}\|^2)^2\Delta t} K_{long}[\mu](\boldsymbol{y}, t-\eta)\Big] e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\eta} & \|\boldsymbol{\nu}\| \neq 0 \end{cases} \end{aligned} \tag{C.19}$$

The evaluation of the three intermediate quantities, $\hat{f}_{temp2}$, h_1 and h_2, are shown in algorithms 7 to 9. Algorithm 10 shows the numerical implementation corresponding to the computation of eqs. (C.14) to (C.16).

Algorithm 7: Definition of `fhat_temp2`.

Function `fhat_temp2`($\boldsymbol{\nu}$, t, $\{\boldsymbol{x}_i\}$, $\{\Delta l_i\}$, $\{\boldsymbol{n}_i\}$, $\{\xi_i\}$, D_0, $\mathbf{q}$, η, Δt, K_long)**:**

"""*Numerical implementation of eq. (C.17).*"""

$result = 0$

forall $i \in [1:1:\#\{\boldsymbol{x}_i\}]$ **do** *# can be parallelized*

$res\_h1 = 2\pi\imath\rho\mathbf{q}\cdot\boldsymbol{n}_i\, e^{-2\pi\imath\mathbf{q}\cdot\boldsymbol{x}_i}\cdot \mathtt{h1}(\boldsymbol{\nu}, t, D_0, \mathbf{q}, \eta, \Delta t)$

$res\_h2 = \mathtt{h2}(\boldsymbol{\nu}, D_0, \eta, \Delta t, K\_long(\boldsymbol{x}_i, t-\eta-\Delta t), K\_long(\boldsymbol{x}_i, t-\eta))$

$result = result + 2\left(1-\sqrt{\frac{D_0\eta}{\pi}}\xi_i\right)^{-1} e^{-2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_i}\,(res\_h1 - res\_h2)\cdot\Delta l_i$

end

return *result*

Algorithm 8: Definition of `h1`.

Function `h1`($\boldsymbol{\nu}$, t, D_0, $\mathbf{q}$, η, Δt)**:**

"""*Numerical implementation of eq. (C.18).*"""

if $\|\boldsymbol{\nu}\| = \|\mathbf{q}\|$ **then**

$result = \Delta t\cdot e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2 t}$

else

$$result = \frac{e^{-4\pi^2 D_0\left[\|\mathbf{q}\|^2(t-\eta-\Delta t)+\|\boldsymbol{\nu}\|^2(\eta+\Delta t)\right]} - e^{-4\pi^2 D_0\left[\|\mathbf{q}\|^2(t-\eta)+\|\boldsymbol{\nu}\|^2\eta\right]}}{4\pi^2 D_0(\|\mathbf{q}\|^2-\|\boldsymbol{\nu}\|^2)}$$

end

return *result*

Algorithm 9: Definition of `h2`.

Function `h2`($\boldsymbol{\nu}$, D_0, η, Δt, $K1$, $K2$)**:**

"""*Numerical implementation of eq. (C.19).*"""

if $\|\boldsymbol{\nu}\| = 0$ **then**

$result = \frac{\Delta t}{2}(K1+K2)$

else

$weight1 =$

$\left[1 - e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t}(4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t + 1)\right] / \left[(4\pi^2 D_0\|\boldsymbol{\nu}\|^2)^2\Delta t\right]$

$weight2 = \left[e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t} + 4\pi^2 D_0\|\boldsymbol{\nu}\|^2\Delta t - 1\right] / \left[(4\pi^2 D_0\|\boldsymbol{\nu}\|^2)^2\Delta t\right]$

$result = (weight1\cdot K1 + weight2\cdot K2)e^{-4\pi^2 D_0\|\boldsymbol{\nu}\|^2\eta}$

end

return *result*

Algorithm 10: Computation of $\hat{f}$, $K_{long}[\mu]$, and μ for $t \in [2\eta + \Delta t,\ \Delta - \delta]$

Data: $NeuData$, D_0, η, Δt, $\Delta\nu$, $\mathbf{q}$, $\{\boldsymbol{x}_i\}$, $\{\Delta l_i\}$, $\{\boldsymbol{n}_i\}$, $\{\xi_i\}$,
$\mathcal{V} = [-\nu_{max} : \Delta\nu : \nu_{max}]$, $\mathcal{T}_3 = [2\eta + \Delta t : \Delta t : \Delta - \delta]$

Result: two arrays *mu* and *K_long* whose sizes are $\#\{\boldsymbol{x}_i\} \times \#\mathcal{T}$ and an array *fhat* of size $\#\mathcal{V} \times \#\mathcal{V} \times \#\mathcal{T}$

begin

"""Numerical computation of eqs. (C.14) to (C.16)."""

foreach $t_m \in \mathcal{T}_3$ **do**

 # compute Fourier coefficient at time t_m

forall $\boldsymbol{\nu} \in \mathcal{V}^2$ **do** *# can be parallelized*

$temp =$
`fhat_temp2`$(\boldsymbol{\nu}, t_m, \{\boldsymbol{x}_i\}, \{\Delta l_i\}, \{\boldsymbol{n}_i\}, \{\xi_i\}, D_0, \mathbf{q}, \eta, \Delta t, K\_long)$

$fhat(\boldsymbol{\nu}, t_m) = e^{-4\pi^2 D_0 \|\boldsymbol{\nu}\|^2 \Delta t} \cdot fhat(\boldsymbol{\nu}, t_m - \Delta t) + temp$

end

forall $i \in [1 : 1 : \#\{\boldsymbol{x}_i\}]$ **do** *# can be parallelized*

 # Inverse Discrete Fourier Transform (IDFT)

$K\_long(\boldsymbol{x}_i, t_m) = D_0 \sum_{\boldsymbol{\nu} \in \mathcal{V}^2} (2\pi\imath\boldsymbol{\nu} \cdot \boldsymbol{n}_i) fhat(\boldsymbol{\nu}, t_m) e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_i} \Delta\nu^2$

$mu(\boldsymbol{x}_i, t_m) =$
$2\left[NeuData(\boldsymbol{x}_i, t_m) - K\_long(\boldsymbol{x}_i, t_m)\right] / \left(1 - \sqrt{\frac{D_0\eta}{\pi}}\xi_i\right)$

end

end

end

C.4 Computation of the diffusion MRI signal

Finally, the signal attenuation E is

$$E = \rho e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 (\Delta - \delta)} + \frac{1}{\sum_{j=1}^{J} |\Omega_j|} \overline{\omega}(\mathbf{q}, \Delta - \delta), \tag{C.20}$$

where $\omega(\boldsymbol{x}_0, t) = S_{short}[\mu](\boldsymbol{x}_0, t) + S_{long}[\mu](\boldsymbol{x}_0, t)$, defined using μ and $\hat{f}$:

$$S_{short}[\mu](\boldsymbol{x}_0, t) = \sqrt{\frac{D_0 t}{\pi}} \mu(\boldsymbol{x}_0, t), \qquad t \in [0,\ \eta], \tag{C.21}$$

$$S_{short}[\mu](\boldsymbol{x}_0, t) = \sqrt{\frac{D_0 \eta}{\pi}} \mu(\boldsymbol{x}_0, t), \qquad t \in [\eta + \Delta t,\ \Delta - \delta], \tag{C.22}$$

$$S_{long}[\mu](\boldsymbol{x}_0, t) = D_0 \sum_{\boldsymbol{\nu} = -\boldsymbol{\nu}_{max}}^{\boldsymbol{\nu}_{max}} \hat{f}(\boldsymbol{\nu}, t) e^{2\pi \imath \boldsymbol{\nu} \cdot \boldsymbol{x}_0} \Delta \nu^2, \qquad t \in [0,\ \Delta - \delta]. \tag{C.23}$$

Then $\overline{\omega}$ is obtained by iteration from ω:

$$\overline{\omega}(\mathbf{q}, t) = \ e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t} \overline{\omega}(\mathbf{q}, t - \Delta t) - D_0\ \overline{\omega}_{temp}(\mathbf{q}, t), \tag{C.24}$$

where

$$\overline{\omega}_{temp}(\mathbf{q}, t) = \int_{\Gamma} 2\pi \imath \mathbf{q} \cdot \boldsymbol{n} e^{2\pi \imath \mathbf{q} \cdot \boldsymbol{y}}\ u\ ds_{\boldsymbol{y}}, \tag{C.25}$$

$$u = \int_{t - \Delta t}^{t} e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 (t - \tau)} \omega(\boldsymbol{y}, \tau) d\tau. \tag{C.26}$$

By applying the trapezoidal rule to the time integration, we get

$$u = \begin{cases} \frac{\Delta t}{2} \left[\omega(\boldsymbol{y}, t - \Delta t) + \omega(\boldsymbol{y}, t)\right] & \|\mathbf{q}\| = 0 \\ \frac{1 - e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t} (4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t + 1)}{(4\pi^2 D_0 \|\mathbf{q}\|^2)^2 \Delta t} \omega(\boldsymbol{y}, t - \Delta t) & \\ + \frac{e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t} + 4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t - 1}{(4\pi^2 D_0 \|\mathbf{q}\|^2)^2 \Delta t} \omega(\boldsymbol{y}, t) & \|\mathbf{q}\| \neq 0 \end{cases} \tag{C.27}$$

We obtain the diffusion MRI signal (eq. (C.20)) by solving eqs. (C.21) to (C.27). Their numerical implementations are shown in algorithms 11 to 14.

Algorithm 11: Definition of `omega_bar_temp`.

Function `omega_bar_temp`(t, $\{\boldsymbol{x}_i\}$, $\{\Delta l_i\}$, $\{\boldsymbol{n}_i\}$, D_0, $\mathbf{q}$, Δt, S_single)**:**

"""*Numerical implementation of eq. (C.25).*"""

$result = 0$

 # spatial integration

forall $i \in [1:1:\#\{\boldsymbol{x}_i\}]$ **do** *# can be parallelized*

$res\_u = \mathtt{u}(D_0, \mathbf{q}, \Delta t, S\_single(\boldsymbol{x}_i, t - \Delta t), S\_single(\boldsymbol{x}_i, t))$

$result = result + 2\pi\imath\mathbf{q} \cdot \boldsymbol{n}_i e^{2\pi\imath\mathbf{q}\cdot\boldsymbol{x}_i}\ res\_u\ \Delta l_i$

end

return *result*

Algorithm 12: Definition of `u`.

Function `u`(D_0, $\mathbf{q}$, Δt, s_1, s_2)**:**

"""*Numerical implementation of eq. (C.27).*"""

if $\|\mathbf{q}\| = 0$ **then**

$$result = \frac{\Delta t}{2}[s_1 + s_2]$$

else

$$weight1 = [1 - e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t}(4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t + 1)]/[(4\pi^2 D_0 \|\mathbf{q}\|^2)^2 \Delta t]$$

$$weight2 = [e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t} + 4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t - 1]/[(4\pi^2 D_0 \|\mathbf{q}\|^2)^2 \Delta t]$$

$$result = weight1 \cdot s_1 + weight2 \cdot s_2$$

end

return *result*

Algorithm 13: Computation of the single layer potential $S[\mu]$.

Data: *mu*, *fhat*, D_0, η, $\{\boldsymbol{x}_i\}$, $\mathcal{V} = [-\nu_{max} : \Delta\nu : \nu_{max}]$, $\mathcal{T} = [0 : \Delta t : \Delta - \delta]$
Result: an array *S_single* of size $\#\{\boldsymbol{x}_i\} \times \#\mathcal{T}$
begin
  """*Numerical computation of eqs. (C.21) to (C.23).*"""
  forall $t_m \in \mathcal{T}$ **do**
    forall $i \in [1 : 1 : \#\{\boldsymbol{x}_i\}]$ **do** *# can be parallelized*
      # compute S_short
      if $t_m \leq \eta$ **then**
        $S\_short(\boldsymbol{x}_i, t_m) = \sqrt{\frac{D_0 t_m}{\pi}} mu(\boldsymbol{x}_i, t_m)$
      else
        $S\_short(\boldsymbol{x}_i, t_m) = \sqrt{\frac{D_0 \eta}{\pi}} mu(\boldsymbol{x}_i, t_m)$
      end
      # compute S_long
      $S\_long(\boldsymbol{x}_i, t_m) = D_0 \sum_{\boldsymbol{\nu} \in \mathcal{V}^2} fhat(\boldsymbol{\nu}, t_m) e^{2\pi\imath\boldsymbol{\nu}\cdot\boldsymbol{x}_i} \Delta\nu^2$
      # S_single = S_short + S_long
      $S\_single(\boldsymbol{x}_i, t_m) = S\_short(\boldsymbol{x}_i, t_m) + S\_long(\boldsymbol{x}_i, t_m)$
    end
  end
end

Algorithm 14: Computation of the diffusion MRI signal.

Data: *area*, *S_single*, $\{\boldsymbol{x}_i\}$, $\{\Delta l_i\}$, $\{\boldsymbol{n}_i\}$, D_0, $\mathbf{q}$, Δt, $\mathcal{T} = [0 : \Delta t : \Delta - \delta]$
Result: a number *E*
begin
  """*Numerical computation of eq. (C.20).*"""
  foreach $t_m \in \mathcal{T}$ **do**
    if $t_m = 0$ **then**
      $omega\_bar(t_m) = 0$
    else
      $z = \texttt{omega\_bar\_temp}(t_m, \{\boldsymbol{x}_i\}, \{\Delta l_i\}, \{\boldsymbol{n}_i\}, D_0, \mathbf{q}, \Delta t, S\_single)$
      $omega\_bar(t_m) = e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 \Delta t} omega\_bar(t_m - \Delta t) - D_0 \cdot z$
    end
  end
  $E = \rho e^{-4\pi^2 D_0 \|\mathbf{q}\|^2 (\Delta - \delta)} + omega\_bar(\Delta - \delta)/area$
end

Appendix D

Supplementary Material to Chapter 5

D.1 Microstructure parameters

We compute the microstructure parameters for an artificial brain voxel based on the neuroanatomical parameters defined in section A.1. Suppose a voxel consists of M neuron meshes and an ECS compartment whose volume fraction is f_{ecs}. We utilize the superscript m to indicate the m-th neuron. Some microstructure parameters are

1. total neuron volume, $V_{\text{neuron}}^{\text{all}} = \sum_{m=1}^{M} V_{\text{neuron}}^{m}$;
2. total neurite volume, $V_{\text{neurite}}^{\text{all}} = \sum_{m=1}^{M} V_{\text{neurite}}^{m}$;
3. total soma volume, $V_{\text{soma}}^{\text{all}} = \sum_{m=1}^{M} V_{\text{soma}}^{m}$;
4. brain voxel volume, $V_{\text{voxel}} = V_{\text{neuron}}^{\text{all}}/(1 - f_{\text{ecs}})$;
5. ECS volume, $V_{\text{ecs}} = f_{\text{ecs}} V_{\text{voxel}}$;
6. soma volume fraction, $f_{\text{soma}} = V_{\text{soma}}^{\text{all}}/V_{\text{voxel}}$;
7. neurite volume fraction, $f_{\text{neurite}} = V_{\text{neurite}}^{\text{all}}/V_{\text{voxel}}$;
8. total neuron area, $A_{\text{neuron}}^{\text{all}} = \sum_{m=1}^{M} A_{\text{neuron}}^{m}$;
9. total neurite area, $A_{\text{neurite}}^{\text{all}} = \sum_{m=1}^{M} A_{\text{neurite}}^{m}$;
10. total soma area, $A_{\text{soma}}^{\text{all}} = \sum_{m=1}^{M} A_{\text{soma}}^{m}$;
11. soma area fraction, $a_{\text{soma}} = A_{\text{soma}}^{\text{all}}/A_{\text{neuron}}^{\text{all}}$;

12. neurite area fraction, $a_{\text{neurite}} = A^{\text{all}}_{\text{neurite}}/A^{\text{all}}_{\text{neuron}}$;
13. neuron volume-area ratio, $V^{\text{all}}_{\text{neuron}}/A^{\text{all}}_{\text{neuron}}$;
14. neurite volume-area ratio, $V^{\text{all}}_{\text{neurite}}/A^{\text{all}}_{\text{neurite}}$;
15. soma volume-area ratio, $V^{\text{all}}_{\text{soma}}/A^{\text{all}}_{\text{soma}}$;
16. total neurite length, $L^{\text{all}}_{\text{neurite}} = \sum_{m=1}^{M} L^{m}_{\text{neurite}}$;
17. average neurite length, $\overline{L}^{\text{all}}_{\text{neurite}} = L^{\text{all}}_{\text{neurite}}/M$;
18. average neuron volume, $\overline{V}^{\text{all}}_{\text{neuron}} = V^{\text{all}}_{\text{neuron}}/M$;
19. average neurite volume, $\overline{V}^{\text{all}}_{\text{neurite}} = V^{\text{all}}_{\text{neurite}}/M$;
20. average soma volume, $\overline{V}^{\text{all}}_{\text{soma}} = V^{\text{all}}_{\text{soma}}/M$;
21. average neuron area, $\overline{A}^{\text{all}}_{\text{neuron}} = A^{\text{all}}_{\text{neuron}}/M$;
22. average neurite area, $\overline{A}^{\text{all}}_{\text{neurite}} = A^{\text{all}}_{\text{neurite}}/M$;
23. average soma area, $\overline{A}^{\text{all}}_{\text{soma}} = A^{\text{all}}_{\text{soma}}/M$;
24. total number of stems, $N^{\text{all}}_{stems} = \sum_{m=1}^{M} N^{m}_{\text{stems}}$;
25. average number of stems, $\overline{N}^{\text{all}}_{stems} = N^{\text{all}}_{stems}/M$;
26. average stem length, $\overline{L}_{\text{stem}} = L^{\text{all}}_{\text{neurite}}/N^{\text{all}}_{stems}$;
27. average neurite radius (based on total neurite volume), $\overline{r}^{v}_{\text{neurite}} = \sqrt{\frac{V^{\text{all}}_{\text{neurite}}}{\pi L^{\text{all}}_{\text{neurite}}}}$;
28. average neurite radius (based on total neurite area), $\overline{r}^{a}_{\text{neurite}} = \frac{A^{\text{all}}_{\text{neurite}}}{2\pi L^{\text{all}}_{\text{neurite}}}$;
29. neurite irregularity, $\frac{\overline{r}^{v}_{\text{neurite}} - \overline{r}^{a}_{\text{neurite}}}{\overline{r}^{v}_{\text{neurite}} + \overline{r}^{a}_{\text{neurite}}}$;
30. average soma radius (first order average), $\overline{r}^{(1)}_{\text{soma}} = \frac{\sum_{m=1}^{M} r^{m}_{\text{soma}}}{M}$;
31. average soma radius (second order), $\overline{r}^{(2)}_{\text{soma}} = \sqrt{\frac{\overline{A}^{\text{all}}_{\text{soma}}}{4\pi}} = \sqrt{\frac{\sum_{m=1}^{M} (r^{m}_{\text{soma}})^{2}}{M}}$;

32. average soma radius (third order), $\overline{r}^{(3)}_{\text{soma}} = \sqrt[3]{\dfrac{\overline{V}^{\text{all}}_{\text{soma}}}{4\pi/3}} = \sqrt[3]{\dfrac{\sum_{m=1}^{M}(r^{m}_{\text{soma}})^{3}}{M}}$;

33. average soma radius (kth order), $\overline{r}^{(k)}_{\text{soma}} = \sqrt[k]{\dfrac{\sum_{m=1}^{M}(r^{m}_{\text{soma}})^{k}}{M}}$;

34. volume weighted average soma radius, $\overline{r}^{vw}_{\text{soma}} = \dfrac{\sum_{m=1}^{M} V^{m}_{\text{soma}} r^{m}_{\text{soma}}}{\sum_{m=1}^{M} V^{m}_{\text{soma}}} = \dfrac{(\overline{r}^{(4)}_{\text{soma}})^{4}}{(\overline{r}^{(3)}_{\text{soma}})^{3}}$;

35. other possible mean soma radius, $\overline{r}^{(ab,cd)}_{\text{soma}} = \sqrt[c-d]{\dfrac{(\overline{r}^{(a)}_{\text{soma}})^{c}}{(\overline{r}^{(b)}_{\text{soma}})^{d}}}$ with appropriate integers a, b, c, d;

36. total stem Euclidean distance, $D^{\text{all}}_{\text{euc}} = \sum_{m=1}^{M} D^{m}_{\text{euc}}$;

37. total stem path distance, $D^{\text{all}}_{\text{path}} = \sum_{m=1}^{M} D^{m}_{\text{path}}$;

38. average stem Euclidean distance, $\overline{D}^{\text{all}}_{\text{euc}} = D^{\text{all}}_{\text{euc}}/M$;

39. average stem path distance, $\overline{D}^{\text{all}}_{\text{path}} = D^{\text{all}}_{\text{path}}/M$;

40. contraction, $D^{\text{all}}_{\text{euc}}/D^{\text{all}}_{\text{path}}$;

41. ratio between soma and neuron volume-area ratios, $\dfrac{V^{\text{all}}_{\text{soma}}/A^{\text{all}}_{\text{soma}}}{V^{\text{all}}_{\text{neuron}}/A^{\text{all}}_{\text{neuron}}}$;

42. ratio between neurite and neuron volume-area ratios, $\dfrac{V^{\text{all}}_{\text{neurite}}/A^{\text{all}}_{\text{neurite}}}{V^{\text{all}}_{\text{neuron}}/A^{\text{all}}_{\text{neuron}}}$;

43. ratio between soma and neurite volume-area ratios, $\dfrac{V^{\text{all}}_{\text{soma}}/A^{\text{all}}_{\text{soma}}}{V^{\text{all}}_{\text{neurite}}/A^{\text{all}}_{\text{neurite}}}$.

We can also compute other microstructure parameters like the number of bifurcations, the angle between two bifurcation points, fractal dimension, partition asymmetry, etc., using measurements from L-measure.

D.2 Box plot

A box plot is a method for graphically representing data distribution by their quartiles. The three important percentiles are:

- Median (Q_2 or 50th percentile): the median value in the dataset;
- First quartile (Q_1 or 25th percentile): the median of the left half of the dataset;
- Third quartiles (Q_3 or 75th percentile): the median of the left half of the dataset.

In fig. D.1, the box's left and right sides (hinges) denote Q_1 and Q_3. Q_2 is represented by the bar in the middle of the box. The range between the two hinges is called the interquartile range, which contains the middle 50% of the dataset. The whiskers extend to the range of 1.5 times the interquartile range, as illustrated in fig. D.1. All other data points outside the boundary of the whiskers are treated as outliers. The box and whiskers represent 99.3% of the data points for a normal distribution.

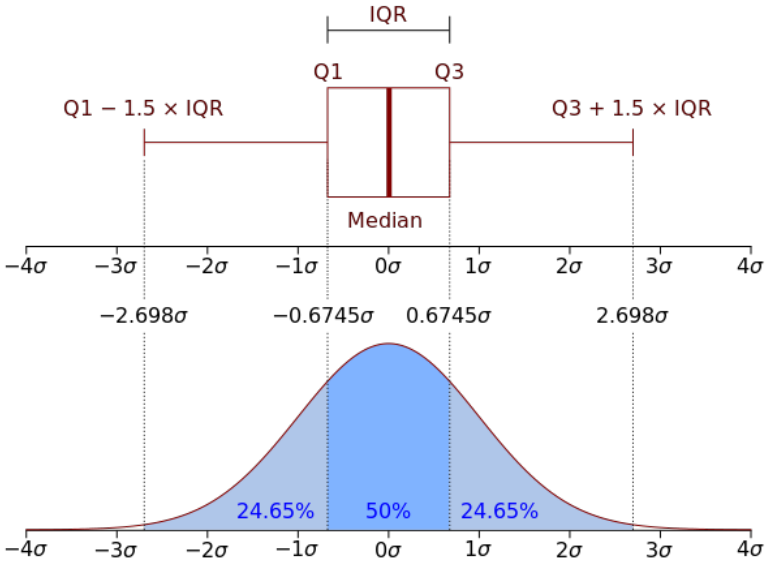


Figure D.1: An illustration of a box plot for a normal distribution $\mathcal{N}(0, \sigma^2)$. By Jhguch at en.wikipedia, CC BY-SA 2.5, https://commons.wikimedia.org/w/index.php?curid=14524285

D.3 Signal comparison between SANDI and simulation at 19 ms

We present the comparison between the simulated signals and the signals recomputed by the SANDI model in logarithmic (fig. D.2) and linear (fig. D.3) scales. The diffusion time is $\delta/\Delta = 8/19\ ms$.

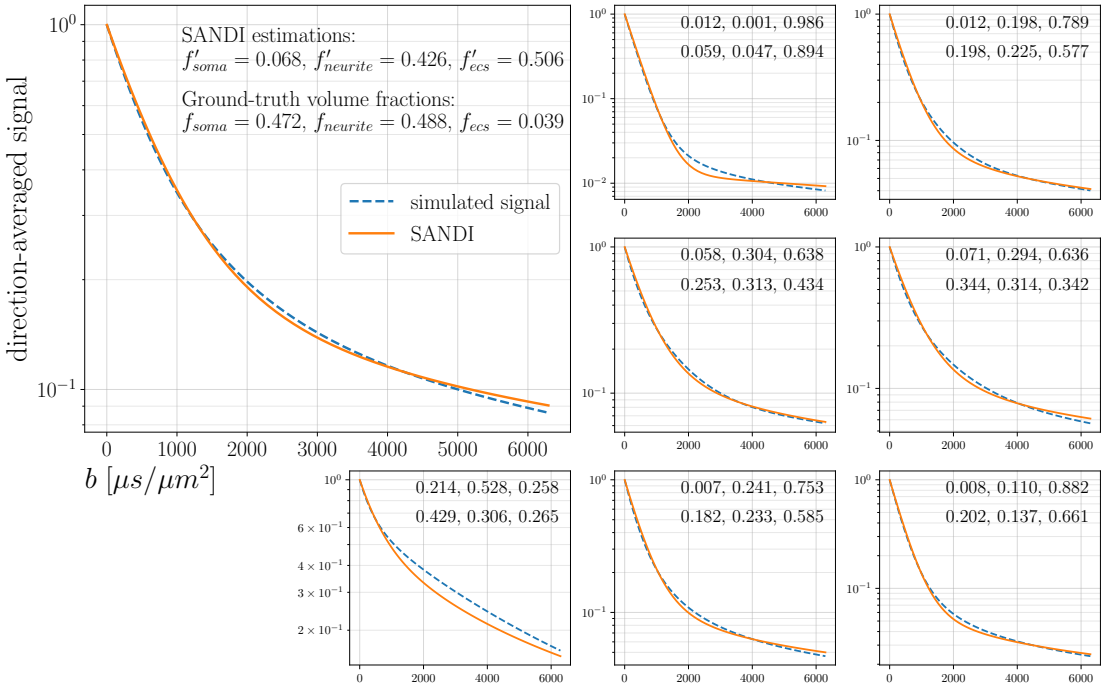


Figure D.2: The comparison between the simulated (blue dashed line) and recomputed signals (solid line) of eight randomly picked artificial brain voxels. The recomputed signals are obtained by substituting SANDI's estimations into eq. (37) in the paper. The diffusion time is $\delta/\Delta = 8/19\ ms$. The first subplot represents the meaning of the six numbers annotated in the upper right corner of each subplot. The numbers in the first row are soma, neurite, and ECS signal fractions. The numbers below are the ground-truth volume fractions. The y-axes are in logarithmic scales.

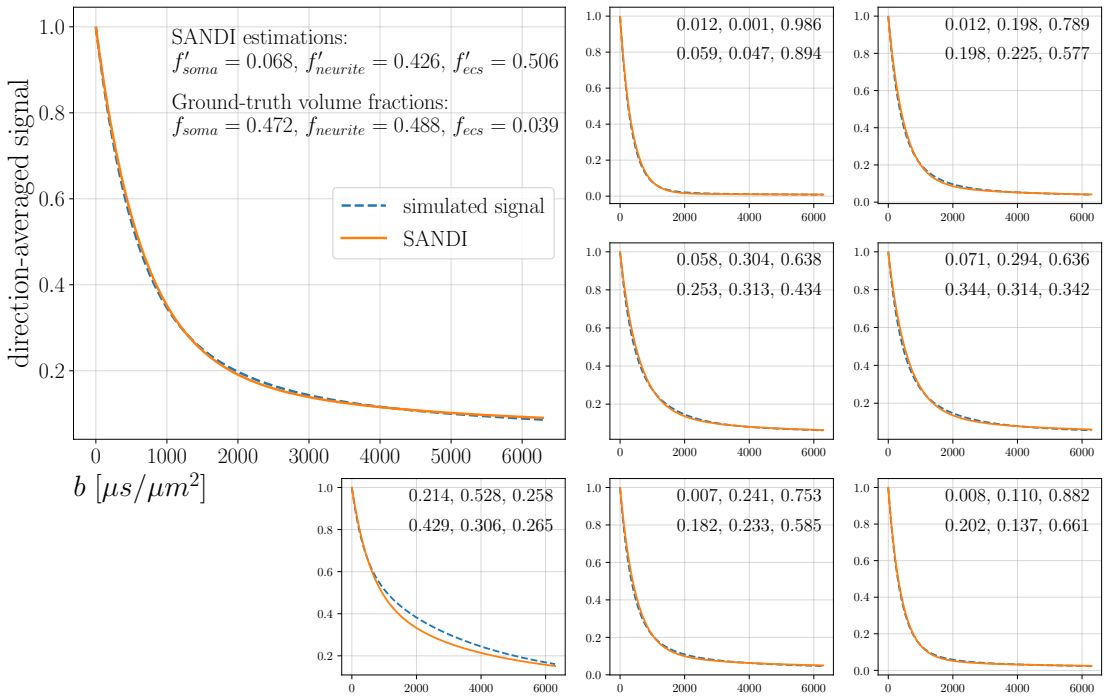


Figure D.3: The comparison between the simulated (blue dashed line) and recomputed signals (solid line) of eight randomly picked artificial brain voxels. The recomputed signals are obtained by substituting SANDI's estimations into eq. (37) in the paper. The diffusion time is $\delta/\Delta = 8/19\ ms$. The first subplot represents the meaning of the six numbers annotated in the upper right corner of each subplot. The numbers in the first row are soma, neurite, and ECS signal fractions. The numbers below are the ground-truth volume fractions. The y-axes are in linear scales.

D.4 Signal comparison between SANDI and simulation at 49 ms

We present the comparison between the simulated signals and the signals recomputed by the SANDI model in logarithmic (fig. D.4) and linear (fig. D.5) scales. The diffusion time is $\delta/\Delta = 8/49\ ms$.

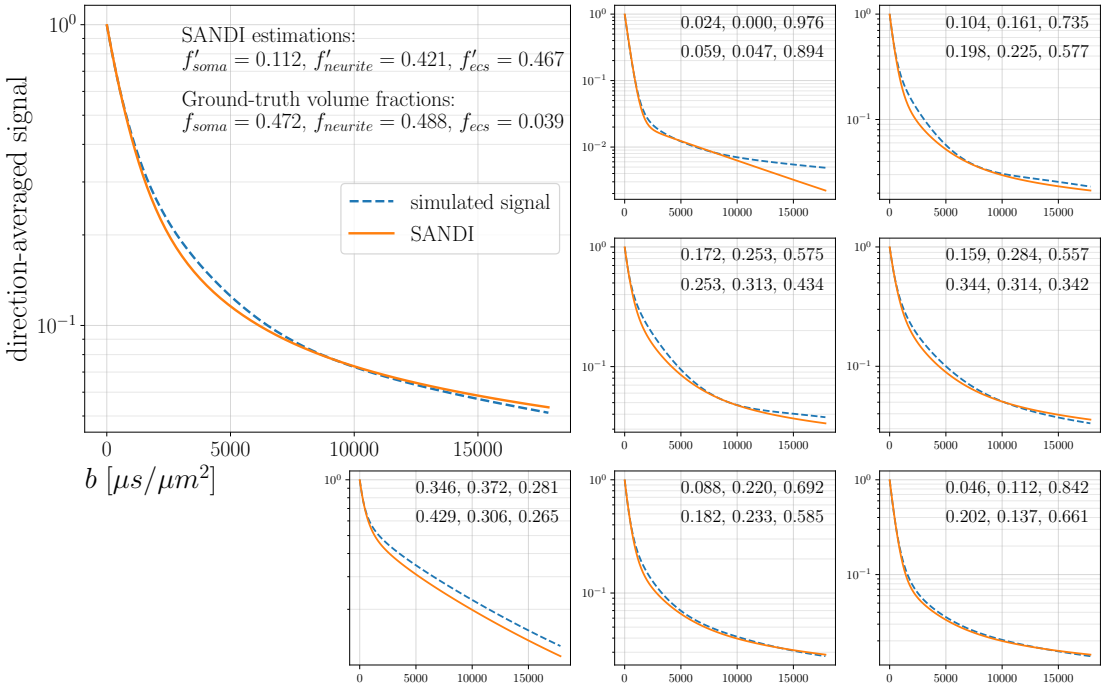


Figure D.4: The comparison between the simulated (blue dashed line) and recomputed signals (solid line) of eight randomly picked artificial brain voxels. The recomputed signals are obtained by substituting SANDI's estimations into eq. (37) in the paper. The diffusion time is $\delta/\Delta = 8/49\ ms$. The first subplot represents the meaning of the six numbers annotated in the upper right corner of each subplot. The numbers in the first row are soma, neurite, and ECS signal fractions. The numbers below are the ground-truth volume fractions. The y-axes are in logarithmic scales.

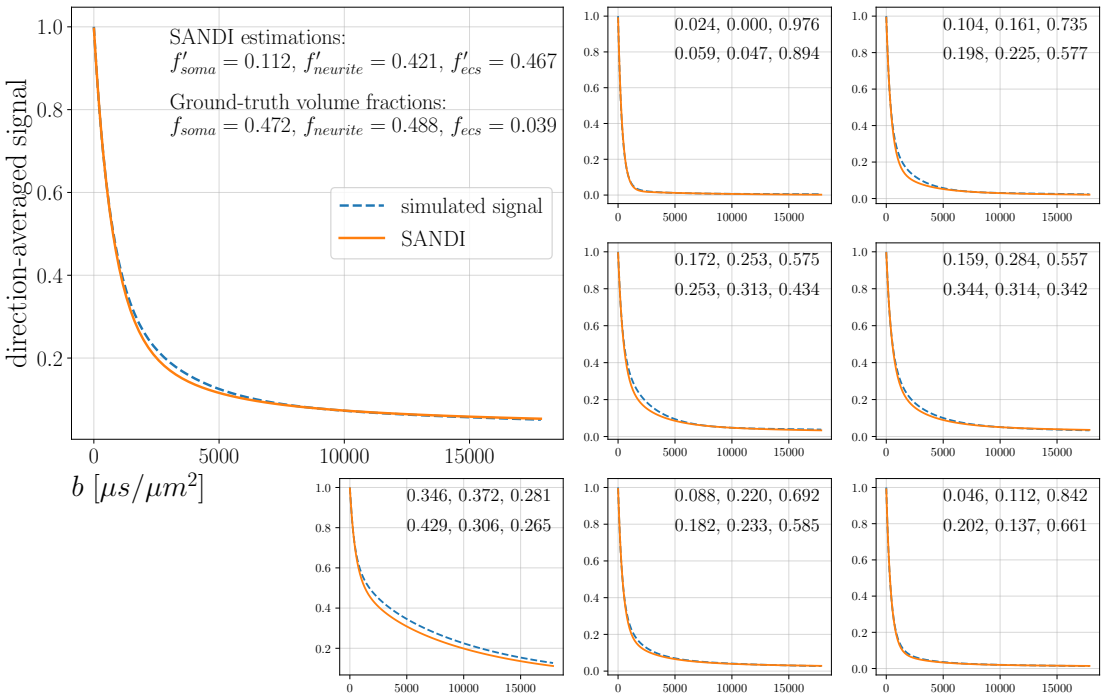


Figure D.5: The comparison between the simulated (blue dashed line) and recomputed signals (solid line) of eight randomly picked artificial brain voxels. The recomputed signals are obtained by substituting SANDI's estimations into eq. (37) in the paper. The diffusion time is $\delta/\Delta = 8/49\ ms$. The first subplot represents the meaning of the six numbers annotated in the upper right corner of each subplot. The numbers in the first row are soma, neurite, and ECS signal fractions. The numbers below are the ground-truth volume fractions. The y-axes are in linear scales.

D.5 SANDI's parameter maps for `sub_002` with dense parameter sampling

We present SANDI's parameter maps of estimated signal fractions for `sub_002` in MGH CDMD (fig. D.6). The parameter distributions for fitting SANDI to experimental signals are: 50 values of D_{in} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 50 values of D_{ecs} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 50 values of r_s linearly spaced in $[1, 12]\ \mu m$; L1 and L2 regularization terms are 0 and 5×10^{-3}, respectively.

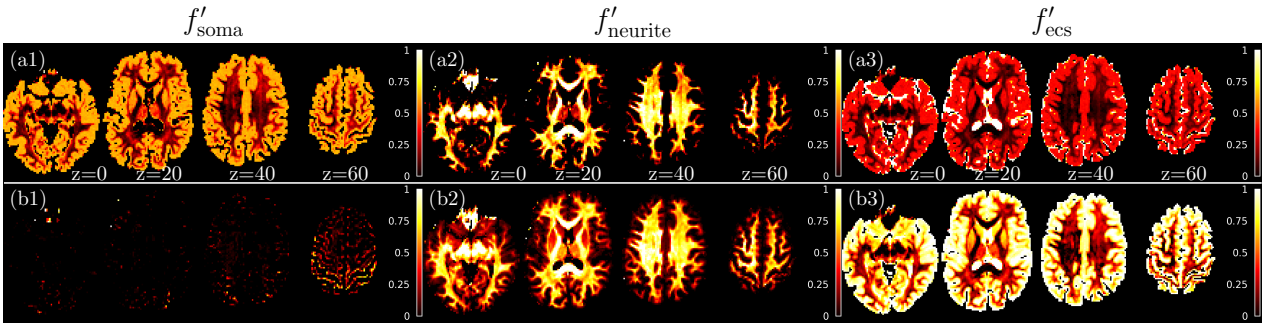


Figure D.6: The parameter maps of signal fractions. The first column is for soma signal fraction f'_{soma}, the second for neurite, and the third for ECS. The first row is for the short diffusion time ($\delta/\Delta = 8/19\ ms$). The second row is for the long diffusion time ($\delta/\Delta = 8/49\ ms$). The parameter maps show the signal fractions obtained by fitting the SANDI model to the direction-averaged signals from `sub_002`. The parameter distributions for fitting SANDI to experimental signals are: 50 values of D_{in} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 50 values of D_{ecs} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 50 values of r_s linearly spaced in $[1, 12]\ \mu m$; L1 and L2 regularization terms are 0 and 5×10^{-3}, respectively.

D.6 Parameter maps and joint distributions for sub_011

We present the parameter maps of volume and signal fractions for sub_011 in MGH CDMD (fig. D.7) and the joint distributions at the two diffusion times (fig. D.8).

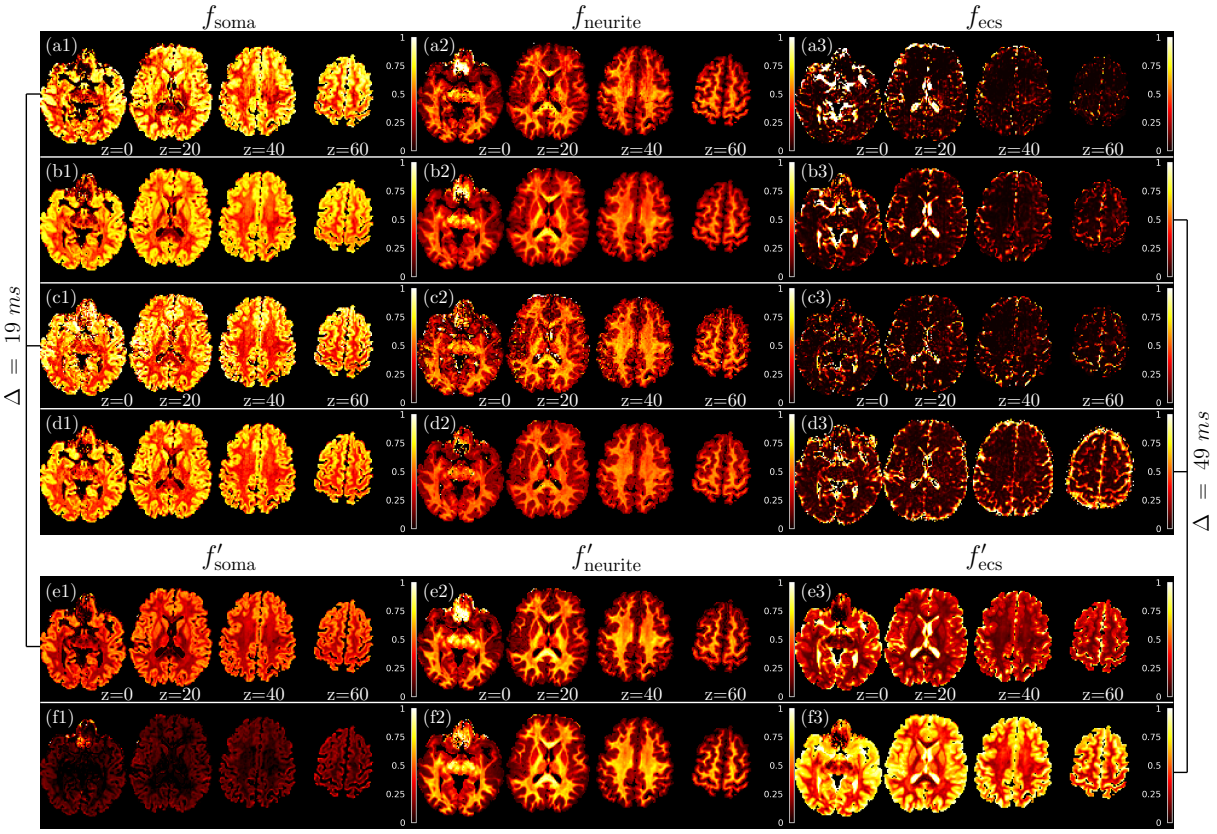


Figure D.7: The comparison of volume and signal fractions. The first column is for soma volume fraction f_{soma} or soma signal fraction f'_{soma}, the second for neurite, and the third for ECS. Three rows, (a), (c), and (e), are for the short diffusion time ($\delta/\Delta = 8/19\ ms$). The remaining rows are for the long diffusion time ($\delta/\Delta = 8/49\ ms$). The first four rows, (a) (b), (c), and (d), are obtained by respectively applying `mlp_sig_vol_19`, `mlp_sig_vol_49`, `mlp_mk_vol_19`, and `mlp_mk_vol_49`, to the experimental data from `sub_011`. The last two rows, (e) and (f), show the signal fractions obtained by fitting the SANDI model to the direction-averaged signals from `sub_011`. The parameter distributions for fitting SANDI to experimental signals are: 10 values of D_{in} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 10 values of D_{ecs} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 10 values of r_s linearly spaced in $[1, 12]\ \mu m$; L1 and L2 regularization terms are 0 and 5×10^{-3}, respectively.

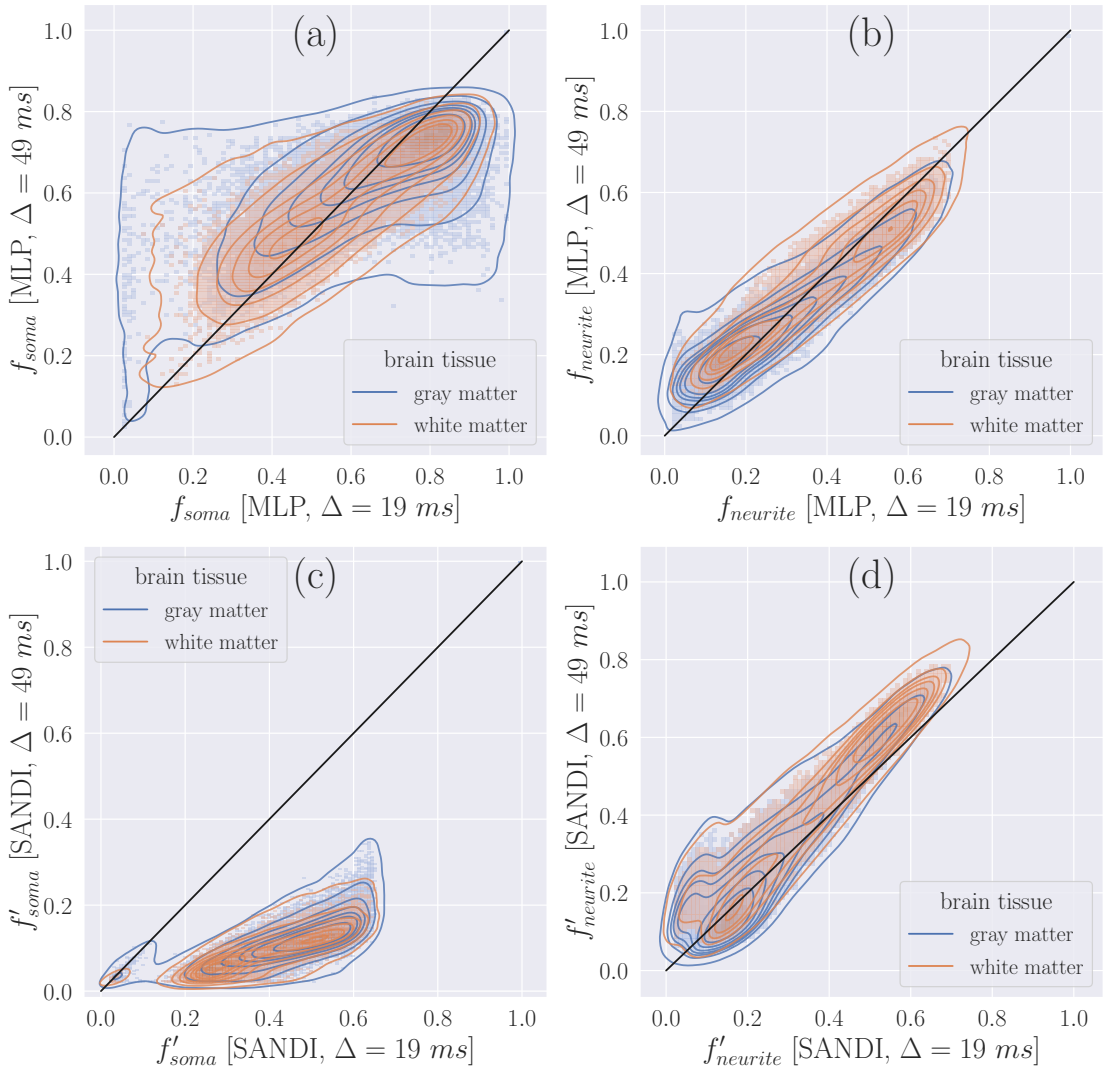


Figure D.8: The voxelwise joint distribution of estimated fractions at two diffusion times. All brain white and gray matter voxels of `sub_011` are included. The x- and y-axes represent the estimated fractions at $\delta/\Delta = 8/19\ ms$ and $\delta/\Delta = 8/49\ ms$, respectively. The black lines are the identity lines. (a) the distribution of the soma volume fractions estimated by `mlp_sig_vol_19` and `mlp_sig_vol_49`. (b) the distribution of the neurite volume fractions estimated by `mlp_sig_vol_19` and `mlp_sig_vol_49`. (c) the distribution of the soma signal fractions estimated by the SANDI model at the two diffusion times. (d) the distribution of the neurite signal fractions estimated by the SANDI model at the two diffusion times.

D.7 Parameter maps and joint distributions for sub_023

We present the parameter maps of volume and signal fractions for sub_023 in MGH CDMD (fig. D.9) and the joint distributions at the two diffusion times (fig. D.10).

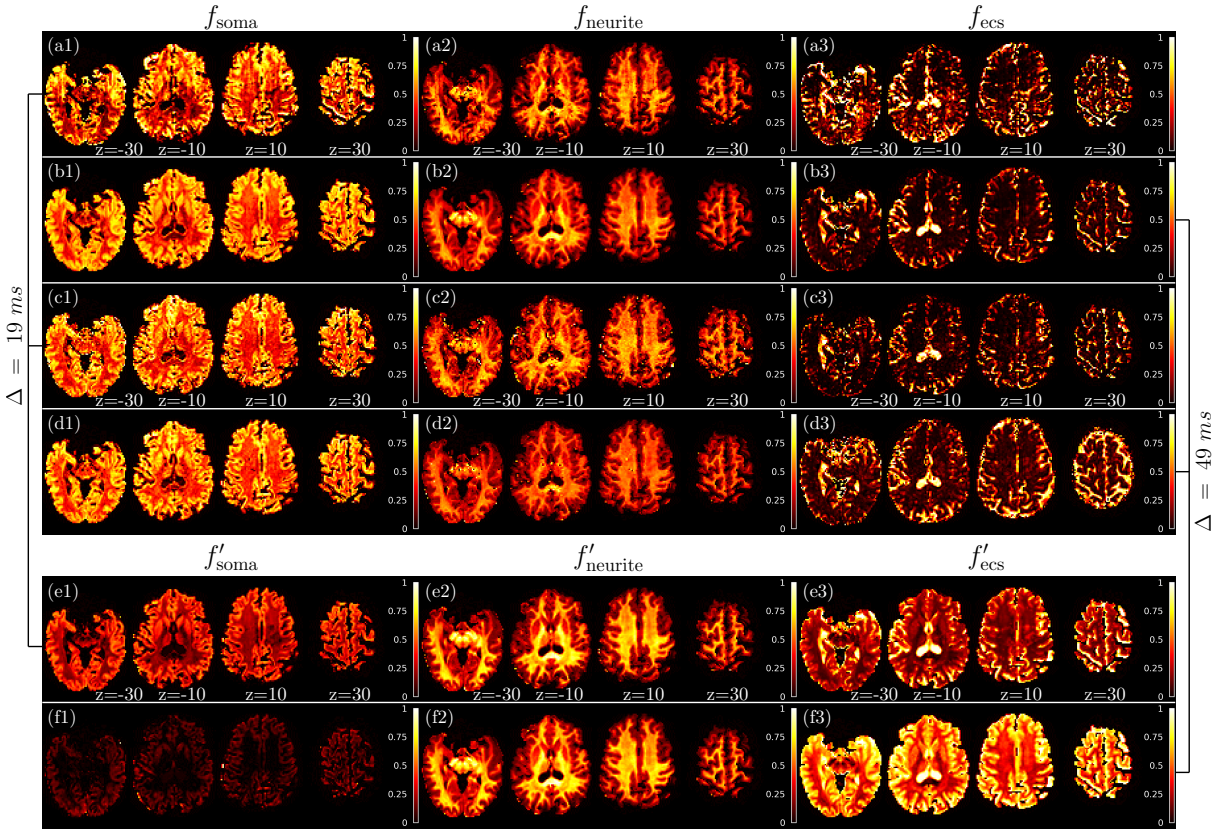


Figure D.9: The comparison of volume and signal fractions. The first column is for soma volume fraction f_{soma} or soma signal fraction f'_{soma}, the second for neurite, and the third for ECS. Three rows, (a), (c), and (e), are for the short diffusion time ($\delta/\Delta = 8/19\ ms$). The remaining rows are for the long diffusion time ($\delta/\Delta = 8/49\ ms$). The first four rows, (a) (b), (c), and (d), are obtained by respectively applying `mlp_sig_vol_19`, `mlp_sig_vol_49`, `mlp_mk_vol_19`, and `mlp_mk_vol_49`, to the experimental data from sub_023. The last two rows, (e) and (f), show the signal fractions obtained by fitting the SANDI model to the direction-averaged signals from sub_023. The parameter distributions for fitting SANDI to experimental signals are: 10 values of D_{in} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 10 values of D_{ecs} linearly spaced in $[0.1, 3] \times 10^{-3}\ \mu m^2/\mu s$; 10 values of r_s linearly spaced in $[1, 12]\ \mu m$; L1 and L2 regularization terms are 0 and 5×10^{-3}, respectively.

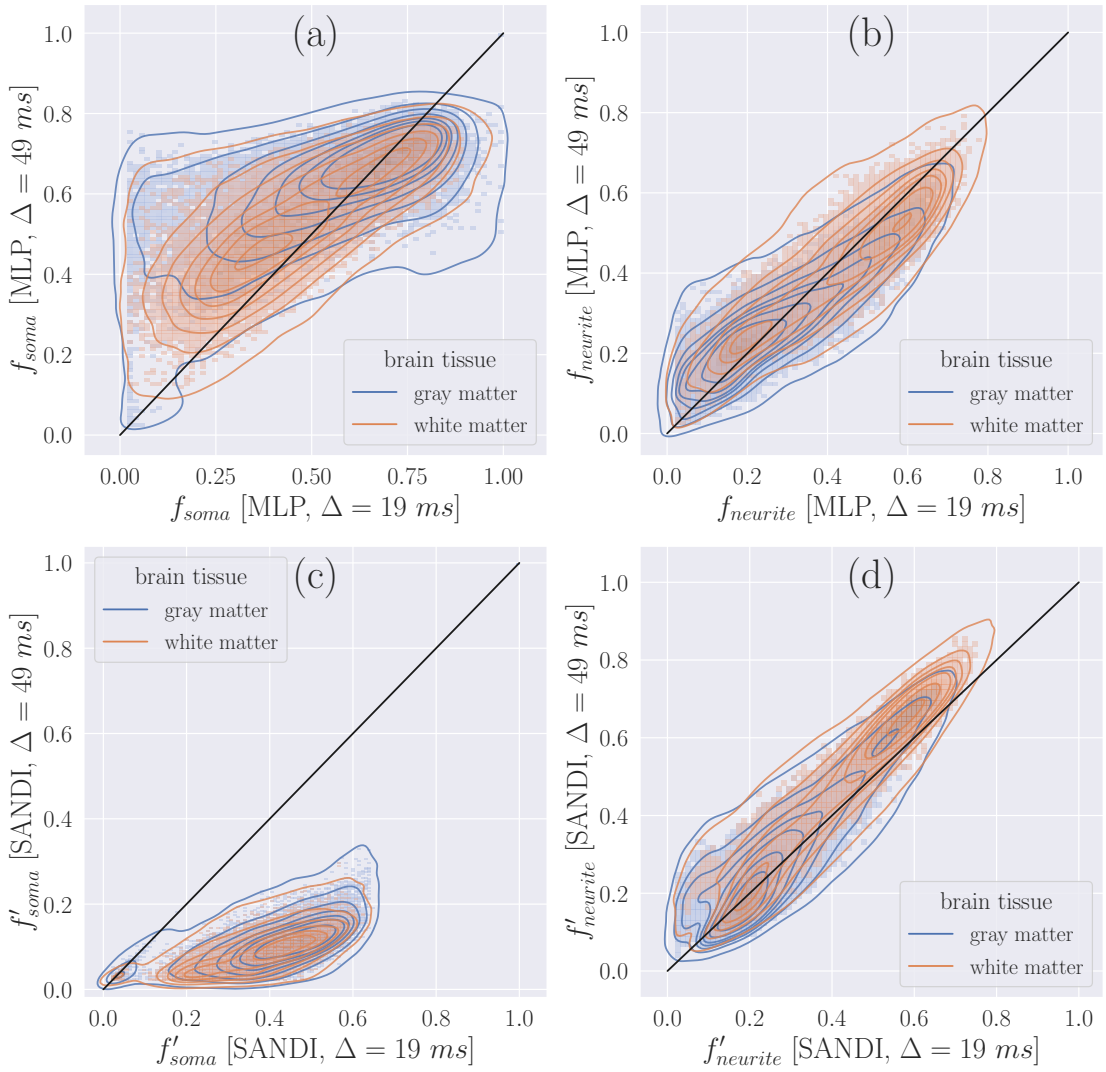


Figure D.10: The voxelwise joint distribution of estimated fractions at two diffusion times. All brain white and gray matter voxels of `sub_023` are included. The x- and y-axes represent the estimated fractions at $\delta/\Delta = 8/19\ ms$ and $\delta/\Delta = 8/49\ ms$, respectively. The black lines are the identity lines. (a) the distribution of the soma volume fractions estimated by `mlp_sig_vol_19` and `mlp_sig_vol_49`. (b) the distribution of the neurite volume fractions estimated by `mlp_sig_vol_19` and `mlp_sig_vol_49`. (c) the distribution of the soma signal fractions estimated by the SANDI model at the two diffusion times. (d) the distribution of the neurite signal fractions estimated by the SANDI model at the two diffusion times.

Bibliography

[1] C. Henley, *Foundations of Neuroscience* (Open textbook library). Michigan State University, 2021. [Online]. Available: https://openbooks.lib.msu.edu/neuroscience/.

[2] C. Chayer and M. Freedman, "Frontal lobe functions," *Current neurology and neuroscience reports*, vol. 1, no. 6, pp. 547–552, 2001.

[3] J. N. Sanes, J. P. Donoghue, *et al.*, "Plasticity and primary motor cortex," *Annual review of neuroscience*, vol. 23, no. 1, pp. 393–415, 2000.

[4] M. Borich, S. Brodie, W. Gray, S Ionta, and L. Boyd, "Understanding the role of the primary somatosensory cortex: Opportunities for rehabilitation," *Neuropsychologia*, vol. 79, pp. 246–255, 2015.

[5] D. Purves, *Brains as engines of association: an operating principle for nervous systems*. Oxford University Press, 2019.

[6] D. Debanne, E. Campanac, A. Bialowas, E. Carlier, and G. Alcaraz, "Axon physiology," *Physiological reviews*, vol. 91, no. 2, pp. 555–602, 2011.

[7] N. Baumann and D. Pham-Dinh, "Biology of oligodendrocyte and myelin in the mammalian central nervous system," *Physiological reviews*, 2001.

[8] C. Koch and A. Jones, "Big science, team science, and open science for neuroscience," *Neuron*, vol. 92, no. 3, pp. 612–616, 2016.

[9] S. Herculano-Houzel, "The remarkable, yet not extraordinary, human brain as a scaled-up primate brain and its associated cost," *Proceedings of the National Academy of Sciences*, vol. 109, no. supplement_1, pp. 10 661–10 668, 2012.

[10] W. G. Webb, "4 - neuronal function in the nervous system," in *Neurology for the Speech-Language Pathologist (Sixth Edition)*, W. G. Webb, Ed., Sixth Edition, Mosby, 2017, pp. 74–92, isbn: 978-0-323-10027-4. doi: https://doi.org/10.1016/B978-0-323-10027-4.00004-X. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B978032310027400004X.

[11] H. Johansen-Berg and T. E. Behrens, *Diffusion MRI: from quantitative measurement to in vivo neuroanatomy*. Academic Press, 2013.

[12] A. Shapson-Coe *et al.*, "A connectomic study of a petascale fragment of human cerebral cortex," *BioRxiv*, 2021.

[13] F. D. Lublin, "Chapter 29 - multiple sclerosis classification and overview," in *Myelin Biology and Disorders*, R. A. Lazzarini, J. W. Griffin, H. Lassman, K.-A. Nave, R. Miller, and B. D. Trapp, Eds., San Diego: Academic Press, 2004, pp. 691–699, isbn: 978-0-12-439510-7. doi: https://doi.org/10.1016/B978-012439510-7/50082-6. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780124395107500826.

[14] M. G. Erkkinen, M.-O. Kim, and M. D. Geschwind, "Clinical neurology and epidemiology of the major neurodegenerative diseases," *Cold Spring Harbor perspectives in biology*, vol. 10, no. 4, a033118, 2018.

[15] N. S. Ward, "Restoring brain function after stroke—bridging the gap between animals and humans," *Nature Reviews Neurology*, vol. 13, no. 4, pp. 244–255, 2017.

[16] E. Panagiotaki *et al.*, "Noninvasive quantification of solid tumor microstructure using verdict mri," *Cancer research*, vol. 74, no. 7, pp. 1902–1912, 2014.

[17] E. L. Hahn, "Spin echoes," *Phys. Rev.*, vol. 80, pp. 580–594, 4 1950.

[18] E. O. Stejskal and J. E. Tanner, "Spin diffusion measurements: Spin echoes in the presence of a time-dependent field gradient," *The Journal of Chemical Physics*, vol. 42, no. 1, pp. 288–292, 1965. doi: 10.1063/1.1695690.

[19] D. L. Bihan, E Breton, D Lallemand, P Grenier, E Cabanis, and M Laval-Jeantet, "MR imaging of intravoxel incoherent motions: Application to diffusion and perfusion in neurologic disorders.," *Radiology*, vol. 161, no. 2, pp. 401–407, 1986, PMID: 3763909.

[20] C. Beaulieu, "The basis of anisotropic water diffusion in the nervous system–a technical review," *NMR in Biomedicine: An International Journal Devoted to the Development and Application of Magnetic Resonance In Vivo*, vol. 15, no. 7-8, pp. 435–455, 2002.

[21] A. T. Van, C. Granziera, and R. Bammer, "An introduction to model-independent diffusion mri," *Topics in magnetic resonance imaging: TMRI*, vol. 21, no. 6, p. 339, 2010.

[22] Q. Tian *et al.*, "Comprehensive diffusion mri dataset for in vivo human brain microstructure mapping using 300 mt/m gradients," *Scientific Data*, vol. 9, no. 1, pp. 1–11, 2022.

[23] M. H. Levitt, *Spin dynamics: basics of nuclear magnetic resonance*. John Wiley & Sons, 2013.

[24] L. G. Hanson, "Is quantum mechanics necessary for understanding magnetic resonance?" *Concepts in Magnetic Resonance Part A: An Educational Journal*, vol. 32, no. 5, pp. 329–340, 2008.

[25] P. T. Callaghan, *Principles of nuclear magnetic resonance microscopy* (Oxford science publications). Oxford University Press, 1993, isbn: 9780198539971.

[26] M. Goldman, "Formal theory of spin–lattice relaxation," *Journal of Magnetic Resonance*, vol. 149, no. 2, pp. 160–187, 2001.

[27] F. Bloch, "Nuclear induction," *Phys. Rev.*, vol. 70, no. 7-8, pp. 460–474, Oct. 1946. [Online]. Available: http://link.aps.org/doi/10.1103/PhysRev.70.460.

[28] N. D. DiProspero, S. Kim, and M. A. Yassa, "Magnetic resonance imaging biomarkers for cognitive decline in down syndrome," in *The Neurobiology of Aging and Alzheimer Disease in Down Syndrome*, Elsevier, 2022, pp. 149–172.

[29] A. E. Derome, *Modern NMR techniques for chemistry research*. Elsevier, 2013.

[30] N. Moutal, "Study of the bloch-torrey equation associated to diffusion magnetic resonance imaging," Ph.D. dissertation, Institut polytechnique de Paris, 2020.

[31] D. Van Nguyen, "A finite elements method to solve the bloch-torrey equation applied to diffusion magnetic resonance imaging of biological tissues," Ph.D. dissertation, Ecole Polytechnique X, 2014.

[32] R. Feynman, R. Leighton, and M. Sands, *The Feynman Lectures on Physics, Vol. I: The New Millennium Edition: Mainly Mechanics, Radiation, and Heat*. Basic Books, 2015, isbn: 9780465040858. [Online]. Available: https://books.google.fr/books?id=d76DBQAAQBAJ.

[33] H. Torrey, "Bloch equations with diffusion terms," *Physical Review Online Archive (Prola)*, vol. 104, no. 3, pp. 563–565, 1956. doi: 10.1103/PhysRev.104.563.

[34] M. Holz, S. R. Heil, and A. Sacco, "Temperature-dependent self-diffusion coefficients of water and six selected molecular liquids for calibration in accurate 1h nmr pfg measurements," *Physical Chemistry Chemical Physics*, vol. 2, no. 20, pp. 4740–4742, 2000.

[35] V. Kenkre, E. Fukushima, and D Sheltraw, "Simple solutions of the torrey–bloch equations in the nmr study of molecular diffusion," *Journal of Magnetic Resonance*, vol. 128, no. 1, pp. 62–69, 1997.

[36] B. D. Hughes, *Random walks and random environments*, English. Clarendon Press Oxford ; New York, 1995, <v. 1–2 > : isbn: 0198537891 0198537883.

[37] C.-H. Yeh, B. Schmitt, D. Le Bihan, J.-R. Li-Schlittgen, C.-P. Lin, and C. Poupon, "Diffusion microscopist simulator: A general monte carlo simulation system for diffusion magnetic resonance imaging," *PLoS ONE*, vol. 8, no. 10, e76626, Oct. 2013. doi: 10.1371/journal.pone.0076626.

[38] G. T. Balls and L. R. Frank, "A simulation environment for diffusion weighted mr experiments in complex media," *Magn. Reson. Med.*, vol. 62, no. 3, pp. 771–778, 2009, issn: 1522-2594. [Online]. Available: http://dx.doi.org/10.1002/mrm.22033.

[39] E. Fieremans, D. S. Novikov, J. H. Jensen, and J. A. Helpern, "Monte carlo study of a two-compartment exchange model of diffusion," *NMR in Biomedicine*, vol. 23, no. 7, pp. 711–724, 2010.

[40] D. S. Novikov and V. G. Kiselev, "Effective medium theory of a diffusion-weighted signal," *NMR in Biomedicine*, vol. 23, no. 7, pp. 682–697, 2010.

[41] H.-H. Lee, E. Fieremans, and D. S. Novikov, "Realistic microstructure simulator (rms): Monte carlo simulations of diffusion in three-dimensional cell segmentations of microscopy images," *Journal of neuroscience methods*, vol. 350, p. 109 018, 2021.

[42] E. Fieremans and H.-H. Lee, "Physical and numerical phantoms for the validation of brain microstructural mri: A cookbook," *NeuroImage*, vol. 182, pp. 39 –61, 2018, Microstructural Imaging, issn: 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2018.06.046. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1053811918305536.

[43] J.-R. Li, D. Calhoun, C. Poupon, and D. L. Bihan, "Numerical simulation of diffusion mri signals using an adaptive time-stepping method," *Physics in Medicine and Biology*, vol. 59, no. 2, p. 441, 2014. [Online]. Available: http://stacks.iop.org/0031-9155/59/i=2/a=441.

[44] J.-R. Li *et al.*, "Spindoctor: A matlab toolbox for diffusion mri simulation," *NeuroImage*, vol. 202, p. 116 120, 2019, issn: 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2019.116120. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1053811919307116.

[45] D. V. Nguyen, J.-R. Li, D. Grebenkov, and D. Le Bihan, "A finite elements method to solve the Bloch-Torrey equation applied to diffusion magnetic resonance imaging," *Journal of Computational Physics*, vol. 263, no. 0, pp. 283–302, Apr. 2014, issn: 0021-9991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999114000308.

[46] L. Beltrachini, Z. A. Taylor, and A. F. Frangi, "A parametric finite element solution of the generalised bloch–torrey equation for arbitrary domains," *Journal of Magnetic Resonance*, vol. 259, pp. 126 –134, 2015, issn: 1090-7807. doi: https://doi.org/10.1016/j.jmr.2015.08.008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1090780715001743.

[47] D. S. Grebenkov, "Laplacian eigenfunctions in nmr. i. a numerical tool," *Concepts in Magnetic Resonance Part A: An Educational Journal*, vol. 32, no. 4, pp. 277–301, 2008.

[48] D. S. Grebenkov, "Laplacian eigenfunctions in nmr. ii. theoretical advances," *Concepts Magn. Reson.*, vol. 34A, no. 5, pp. 264–296, 2009, issn: 1552-5023. doi: 10.1002/cmr.a.20145.

[49] J.-R. Li, T. N. Tran, and V.-D. Nguyen, "Practical computation of the diffusion mri signal of realistic neurons based on laplace eigenfunctions," *NMR in Biomedicine*, vol. 33, no. 10, e4353, 2020.

[50] S. D. Agdestein, T. N. Tran, and J.-r. Li, "Practical computation of the diffusion mri signal based on laplace eigenfunctions: Permeable interfaces," *NMR in Biomedicine*, vol. 35, no. 3, e4646, 2022.

[51] J Xu, M. Does, and J. Gore, "Numerical study of water diffusion in biological tissues using an improved finite difference method," *Physics in medicine and biology*, vol. 52, no. 7, Apr. 2007, issn: 0031-9155. [Online]. Available: http://view.ncbi.nlm.nih.gov/pubmed/17374905.

[52] G. Russell, K. D. Harkins, T. W. Secomb, J.-P. Galons, and T. P. Trouard, "A finite difference method with periodic boundary conditions for simulations of diffusion-weighted magnetic resonance experiments in tissue," *Physics in Medicine and Biology*, vol. 57, no. 4, N35, 2012. [Online]. Available: http://stacks.iop.org/0031-9155/57/i=4/a=N35.

[53] H. Hagslatt, B. Jonsson, M. Nyden, and O. Soderman, "Predictions of pulsed field gradient NMR echo-decays for molecules diffusing in various restrictive geometries. simulations of diffusion propagators based on a finite element method," *Journal of Magnetic Resonance*, vol. 161, no. 2, pp. 138–147, Apr. 2003, issn: 1090-7807. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1090780702000393.

[54] N. Loren, H. Hagslatt, M. Nyden, and A.-M. Hermansson, "Water mobility in heterogeneous emulsions determined by a new combination of confocal laser scanning microscopy, image analysis, nuclear magnetic resonance diffusometry, and finite element method simulation," *The Journal*

of Chemical Physics, vol. 122, no. 2, 024716, pp. –, 2005. doi: 10.1063/1.1830432.

[55] B. F. Moroney, T. Stait-Gardner, B. Ghadirian, N. N. Yadav, and W. S. Price, "Numerical analysis of NMR diffusion measurements in the short gradient pulse limit," *Journal of Magnetic Resonance*, vol. 234, no. 0, pp. 165–175, Sep. 2013, issn: 1090-7807. doi: https://doi.org/10.1016/j.jmr.2013.06.019.

[56] V. D. Nguyen, "A fenics-hpc framework for multi-compartment bloch-torrey models," in *ECCOMAS Congress 2016*, vol. 1, 2016, pp. 105–119.

[57] V.-D. Nguyen, J. Jansson, J. Hoffman, and J.-R. Li, "A partition of unity finite element method for computational diffusion mri," *Journal of Computational Physics*, 2018, issn: 0021-9991. doi: 10.1016/j.jcp.2018.08.039. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999118305709.

[58] V.-D. Nguyen, J. Jansson, H. T. A. Tran, J. Hoffman, and J.-R. Li, "Diffusion mri simulation in thin-layer and thin-tube media using a discretization on manifolds," *Journal of Magnetic Resonance*, vol. 299, pp. 176 –187, 2019, issn: 1090-7807. doi: https://doi.org/10.1016/j.jmr.2019.01.002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1090780719300023.

[59] V.-D. Nguyen *et al.*, "Portable simulation framework for diffusion mri," *Journal of Magnetic Resonance*, vol. 309, p. 106 611, 2019, issn: 1090-7807. doi: https://doi.org/10.1016/j.jmr.2019.106611.

[60] M. Hall and D. Alexander, "Convergence and parameter choice for monte-carlo simulations of diffusion mri," *Medical Imaging, IEEE Transactions on*, vol. 28, no. 9, pp. 1354 –1364, 2009, issn: 0278-0062. doi: 10.1109/TMI.2009.2015756.

[61] M. Palombo *et al.*, "New paradigm to assess brain cell morphology by diffusion-weighted MR spectroscopy in vivo," *Proceedings of the National Academy of Sciences*, vol. 113, no. 24, pp. 6671–6676, 2016, issn: 0027-8424. doi: 10.1073/pnas.1504327113. eprint: http://www.pnas.org/content/113/24/6671.full.pdf.

[62] K. V. Nguyen, E. H. Garzon, and J. Valette, "Efficient gpu-based monte-carlo simulation of diffusion in real astrocytes reconstructed from confocal microscopy," *Journal of Magnetic Resonance*, 2018, issn: 1090-7807. doi: 10.1016/j.jmr.2018.09.013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1090780718302386.

[63] C. A. Waudby and J. Christodoulou, "Gpu accelerated monte carlo simulation of pulsed-field gradient nmr experiments," *Journal of Magnetic Resonance*, vol. 211, no. 1, pp. 67 –73, 2011, issn: 1090-7807. doi: https://doi.org/10.1016/j.jmr.2011.04.004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1090780711001376.

[64] P. A. Cook, Y. Bai, M. G. Hall, S. Nedjati-Gilani, K. K. Seunarine, and D. C. Alexander, "Camino: Diffusion mri reconstruction and processing," 2005.

[65] L. Kerkelä, F. Nery, M. G. Hall, and C. A. Clark, "Disimpy: A massively parallel monte carlo simulator for generating diffusion-weighted mri data in python," *Journal of Open Source Software*, vol. 5, no. 52, p. 2527, 2020.

[66] J. H. Burdette, A. D. Elster, and P. E. Ricci, "Acute cerebral infarction: Quantification of spin-density and t2 shine-through phenomena on diffusion-weighted mr images," *Radiology*, vol. 212, no. 2, pp. 333–339, 1999.

[67] D. Le Bihan, *Apparent diffusion coefficient and beyond: What diffusion mr imaging can tell us about tissue structure*, 2013.

[68] D. S. Grebenkov, "Use, misuse, and abuse of apparent diffusion coefficients," *Concepts Magn. Reson.*, vol. 36A, no. 1, pp. 24–35, 2010, issn: 1552-5023. doi: 10.1002/cmr.a.20152.

[69] D. S. Novikov, E. Fieremans, S. N. Jespersen, and V. G. Kiselev, "Quantifying brain microstructure with diffusion MRI: Theory and parameter estimation," *NMR in Biomedicine*, vol. 32, no. 4, e3998, 2019. doi: 10.1002/nbm.3998. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/nbm.3998.

[70] A. F. Frøhlich, L. Østergaard, and V. G. Kiselev, "Effect of impermeable boundaries on diffusion-attenuated mr signal," *Journal of Magnetic Resonance*, vol. 179, no. 2, pp. 223–233, 2006.

[71] P. Kahlert *et al.*, "Silent and apparent cerebral ischemia after percutaneous transfemoral aortic valve implantation: A diffusion-weighted magnetic resonance imaging study," *Circulation*, vol. 121, no. 7, pp. 870–878, 2010.

[72] A. M. Herneth, S. Guccione, and M. Bednarski, "Apparent diffusion coefficient: A quantitative parameter for in vivo tumor characterization," *European journal of radiology*, vol. 45, no. 3, pp. 208–213, 2003.

[73] M. Horger, K. Weisel, W. Horger, A. Mroue, M. Fenchel, and M. Lichy, "Whole-body diffusion-weighted mri with apparent diffusion coefficient mapping for early response monitoring in multiple myeloma: Preliminary results," *American Journal of Roentgenology*, vol. 196, no. 6, W790–W795, 2011.

[74] P. Basser, J. Mattiello, and D. LeBihan, "MR diffusion tensor spectroscopy and imaging," *Biophysical Journal*, vol. 66, no. 1, pp. 259–267, Jan. 1994, issn: 0006-3495. doi: https://doi.org/10.1016/s0006-3495(94)80775-1.

[75] L. J. O'Donnell and C.-F. Westin, "An introduction to diffusion tensor image analysis," *Neurosurgery Clinics*, vol. 22, no. 2, pp. 185–196, 2011.

[76] L. R. Ranzenberger and T. Snyder, "Diffusion tensor imaging," 2019.

[77] J. Dubois, L. Hertz-Pannier, G. Dehaene-Lambertz, Y. Cointepas, and D. Le Bihan, "Assessment of the early organization and maturation of infants' cerebral white matter fiber bundles: A feasibility study using quantitative diffusion tensor imaging and tractography," *Neuroimage*, vol. 30, no. 4, pp. 1121–1132, 2006.

[78] T. Klingberg *et al.*, "Microstructure of temporo-parietal white matter as a basis for reading ability: Evidence from diffusion tensor magnetic resonance imaging," *Neuron*, vol. 25, no. 2, pp. 493–500, 2000.

[79] M Inglese and M. Bester, "Diffusion imaging in multiple sclerosis: Research and clinical implications," *NMR in Biomedicine*, vol. 23, no. 7, pp. 865–872, 2010.

[80] T. C. Chua, W. Wen, M. J. Slavin, and P. S. Sachdev, "Diffusion tensor imaging in mild cognitive impairment and alzheimer's disease: A review," *Current opinion in neurology*, vol. 21, no. 1, pp. 83–92, 2008.

[81] H. Zhang, T. Schneider, C. A. Wheeler-Kingshott, and D. C. Alexander, "Noddi: Practical in vivo neurite orientation dispersion and density imaging of the human brain," *NeuroImage*, vol. 61, no. 4, pp. 1000–1016, Jul. 2012, issn: 1053-8119. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1053811912003539.

[82] E. Kaden, F. Kruggel, and D. C. Alexander, "Quantitative mapping of the per-axon diffusion coefficients in brain white matter," *Magnetic resonance in medicine*, vol. 75, no. 4, pp. 1752–1763, 2016.

[83] B. Lampinen *et al.*, "Searching for the neurite density with diffusion mri: Challenges for biophysical modeling," *Human Brain Mapping*, vol. 40, no. 8, pp. 2529–2545, 2019. doi: 10.1002/hbm.24542. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/hbm.24542.

[84] I. O. Jelescu, J. Veraart, E. Fieremans, and D. S. Novikov, "Degeneracy in model parameter estimation for multi-compartmental diffusion in neuronal tissue," *NMR in Biomedicine*, vol. 29, no. 1, pp. 33–47, 2016.

[85] J. Veraart, E. Fieremans, and D. S. Novikov, "On the scaling behavior of water diffusion in human brain white matter," *NeuroImage*, vol. 185, pp. 379–387, 2019.

[86] J. Veraart *et al.*, "Noninvasive quantification of axon radii using diffusion mri," *Elife*, vol. 9, e49855, 2020.

[87] S. N. Jespersen, J. L. Olesen, A. Ianuş, and N. Shemesh, "Effects of non-gaussian diffusion on "isotropic diffusion" measurements: An ex-vivo microimaging and simulation study," *Journal of Magnetic Resonance*, vol. 300, pp. 84–94, 2019.

[88] M. Palombo *et al.*, "Sandi: A compartment-based model for non-invasive apparent soma and neurite imaging by diffusion mri," *NeuroImage*, p. 116 835, 2020.

[89] C. D. Kroenke, J. J. Ackerman, and D. A. Yablonskiy, "On the nature of the naa diffusion attenuated mr signal in the central nervous system," *Magnetic resonance in medicine*, vol. 52, no. 5, pp. 1052–1059, 2004. doi: 10.1002/mrm.20260/full.

[90] J. Murday and R. M. Cotts, "Self-diffusion coefficient of liquid lithium," *The Journal of Chemical Physics*, vol. 48, no. 11, pp. 4938–4945, 1968.

[91] B. Balinov, B Jonsson, P. Linse, and O Soderman, "The nmr self-diffusion method applied to restricted diffusion. simulation of echo attenuation from molecules in spheres and between planes," *Journal of Magnetic Resonance, Series A*, vol. 104, no. 1, pp. 17–25, 1993.

[92] M. Palombo *et al.*, "Corrigendum to "sandi: A compartment-based model for non-invasive apparent soma and neurite imaging by diffusion mri"[neuroimage 215 (2020), 116835]," *Neuroimage*, vol. 226, 2021.

[93] A. Daducci, E. J. Canales-Rodríguez, H. Zhang, T. B. Dyrby, D. C. Alexander, and J.-P. Thiran, "Accelerated microstructure imaging via convex optimization (amico) from diffusion mri data," *Neuroimage*, vol. 105, pp. 32–44, 2015.

[94] N. Moutal and D. S. Grebenkov, "The localization regime in a nutshell," *Journal of Magnetic Resonance*, vol. 320, p. 106 836, 2020.

[95] D. Miyamoto, *Swc2vtk*, version 1.1.1, Jan. 30, 2017. [Online]. Available: https://github.com/DaisukeMiyamoto/swc2vtk.

[96] J. R. Glaser and E. M. Glaser, "Neuron imaging with neurolucida—a pc-based system for image combining microscopy," *Computerized Medical Imaging and Graphics*, vol. 14, no. 5, pp. 307–317, 1990.

[97] H. Peng, Z. Ruan, F. Long, J. H. Simpson, and E. W. Myers, "V3d enables real-time 3d visualization and quantitative analysis of large-scale biological image data sets," *Nature biotechnology*, vol. 28, no. 4, pp. 348–353, 2010.

[98] M. Palombo, D. C. Alexander, and H. Zhang, "A generative model of realistic brain cells with application to numerical simulation of the diffusion-weighted mr signal," *NeuroImage*, vol. 188, pp. 391 –402, 2019, issn: 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2018.12.025. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1053811918321694.

[99] J. L. Olesen, L. Østergaard, N. Shemesh, and S. N. Jespersen, "Diffusion time dependence, power-law scaling, and exchange in gray matter," *NeuroImage*, vol. 251, p. 118 976, 2022.

[100] A. Ianuş, D. C. Alexander, and I. Drobnjak, "Microstructure imaging sequence simulation toolbox," in *Simulation and Synthesis in Medical Imaging*, S. A. Tsaftaris, A. Gooya, A. F. Frangi, and J. L. Prince, Eds., Cham: Springer International Publishing, 2016, pp. 34–44, isbn: 978-3-319-46630-9.

[101] I. Drobnjak, H. Zhang, M. G. Hall, and D. C. Alexander, "The matrix formalism for generalised gradients with time-varying orientation in diffusion nmr," *Journal of Magnetic Resonance*, vol. 210, no. 1, pp. 151 –157, 2011, issn: 1090-7807. doi: 10.1016/j.jmr.2011.02.022. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1090780711000838.

[102] M. Mercredi and M. Martin, "Toward faster inference of micron-scale axon diameters using monte carlo simulations," *Magnetic Resonance Materials in Physics, Biology and Medicine*, vol. 31, no. 4, pp. 511–530, 2018, issn: 1352-8661. doi: 10.1007/s10334-018-0680-1. [Online]. Available: https://doi.org/10.1007/s10334-018-0680-1.

[103] G. Rensonnet, B. Scherrer, S. K. Warfield, B. Macq, and M. Taquet, "Assessing the validity of the approximation of diffusion-weighted-mri signals from crossing fascicles by sums of signals from single fascicles," *Magnetic Resonance in Medicine*, vol. 79, no. 4, pp. 2332–2345, 2018. doi: 10.1002/mrm.26832. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.26832. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.26832.

[104] G. Rensonnet *et al.*, "Towards microstructure fingerprinting: Estimation of tissue properties from a dictionary of monte carlo diffusion mri simulations," *NeuroImage*, vol. 184, pp. 964–980, Jan. 2019, issn: 1053-8119. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1053811918319487.

[105] H. Peng, B. Roysam, and G. A. Ascoli, "Automated image computing reshapes computational neuroscience," *BMC bioinformatics*, vol. 14, no. 1, pp. 1–5, 2013.

[106] D. Z. Jin, T. Zhao, D. L. Hunt, R. P. Tillage, C.-L. Hsu, and N. Spruston, "Shutu: Open-source software for efficient and accurate reconstruction of dendritic morphology," *Frontiers in neuroinformatics*, vol. 13, p. 68, 2019.

[107] M. Halavi, K. A. Hamilton, R. Parekh, and G. A. Ascoli, "Digital reconstructions of neuronal morphology: Three decades of research trends," *Frontiers in neuroscience*, vol. 6, p. 49, 2012.

[108] E. Stockley, H. Cole, A. Brown, and H. Wheal, "A system for quantitative morphological measurement and electrotonic modelling of neurons: Three-dimensional reconstruction," *Journal of neuroscience methods*, vol. 47, no. 1-2, pp. 39–51, 1993.

[109] R. C. Cannon, D. Turner, G. Pyapali, and H. Wheal, "An on-line archive of reconstructed hippocampal neurons," *Journal of neuroscience methods*, vol. 84, no. 1-2, pp. 49–54, 1998.

[110] G. A. Ascoli, D. E. Donohue, and M. Halavi, "Neuromorpho.org: A central resource for neuronal morphologies," *Journal of Neuroscience*, vol. 27, no. 35, pp. 9247–9251, 2007, issn: 0270-6474. doi: 10.1523/JNEUROSCI.2055-07.2007. eprint: http://www.jneurosci.org/content/27/35/9247.full.pdf. [Online]. Available: http://www.jneurosci.org/content/27/35/9247.

[111] H. Si, "Tetgen, a delaunay-based quality tetrahedral mesh generator," *ACM Trans. Math. Softw.*, vol. 41, no. 2, 11:1–11:36, Feb. 2015, issn: 0098-3500. doi: 10.1145/2629697. [Online]. Available: http://doi.acm.org/10.1145/2629697.

[112] C. Geuzaine and J. F. Remacle, "Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities," *International Journal for Numerical Methods in Engineering*, 2009.

[113] The CGAL Project, *CGAL User and Reference Manual*, 5.5. CGAL Editorial Board, 2022. [Online]. Available: https://doc.cgal.org/5.5/Manual/packages.html.

[114] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy, *Polygon mesh processing*. CRC press, 2010.

[115] J. P. Brito, S. Mata, S. Bayona, L. Pastor, J. DeFelipe, and R. Benavides-Piccione, "Neuronize: A tool for building realistic neuronal cell morphologies," *Frontiers in neuroanatomy*, vol. 7, p. 15, 2013.

[116] C. Fang, V.-D. Nguyen, D. Wassermann, and J.-R. Li, "Diffusion mri simulation of realistic neurons with spindoctor and the neuron module," *NeuroImage*, vol. 222, p. 117 198, 2020.

[117] *BETA CAE Systems, ANSA pre-processor: The advanced CAE pre-processing software for complete model build up.* `https://www.beta-cae.com`.

[118] J. Shewchuk, "What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures (preprint)," *University of California at Berkeley*, vol. 2002, 2002.

[119] K. K. Watson, T. K. Jones, and J. M. Allman, "Dendritic architecture of the von economo neurons," *Neuroscience*, vol. 141, no. 3, pp. 1107–1112, 2006.

[120] Matthias, *Vtk2stl*, version 1.0.0.0, Jun. 10, 2015. [Online]. Available: `https://www.mathworks.com/matlabcentral/fileexchange/51127-vtk2stl`.

[121] E. A. Nimchinsky, E. Gilissen, J. M. Allman, D. P. Perl, J. M. Erwin, and P. R. Hof, "A neuronal morphologic type unique to humans and great apes," *Proceedings of the National Academy of Sciences*, vol. 96, no. 9, pp. 5268–5273, 1999.

[122] H. C. Evrard, T. Forro, and N. K. Logothetis, "Von economo neurons in the anterior insula of the macaque monkey," *Neuron*, vol. 74, no. 3, pp. 482–489, 2012.

[123] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, vol. 7, 2006, pp. 61–70.

[124] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Transactions on Graphics (ToG)*, vol. 32, no. 3, pp. 1–13, 2013.

[125] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997, pp. 209–216.

[126] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh optimization," in *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, 1993, pp. 19–26.

[127] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, G. Ranzuglia, *et al.*, "Meshlab: An open-source mesh processing tool.," in *Eurographics Italian chapter conference*, Salerno, Italy, vol. 2008, 2008, pp. 129–136.

[128] N. Pietroni, M. Tarini, and P. Cignoni, "Almost isometric mesh parameterization through abstract domains," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 4, pp. 621–635, 2009.

[129] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009, isbn: 1441412697.

[130] Á. González, "Measurement of areas on a sphere using fibonacci and latitude–longitude lattices," *Mathematical Geosciences*, vol. 42, no. 1, pp. 49–64, 2010.

[131] T. L. Hayes and D. A. Lewis, "Magnopyramidal neurons in the anterior motor speech region: Dendritic features and interhemispheric comparisons," *Archives of neurology*, vol. 53, no. 12, pp. 1277–1283, 1996.

[132] B. Jacobs, L. Driscoll, and M. Schall, "Life-span dendritic and spine changes in areas 10 and 18 of human cortex: A quantitative golgi study," *Journal of comparative neurology*, vol. 386, no. 4, pp. 661–680, 1997.

[133] R. Benavides-Piccione, J. I. Arellano, and J. DeFelipe, "Catecholaminergic innervation of pyramidal neurons in the human temporal cortex," *Cerebral Cortex*, vol. 15, no. 10, pp. 1584–1591, 2005.

[134] L. Madisen *et al.*, "Transgenic mice for intersectional targeting of neural sensors and effectors with high specificity and performance," *Neuron*, vol. 85, no. 5, pp. 942–958, 2015.

[135] B. Hrvoj-Mihic *et al.*, "Basal dendritic morphology of cortical pyramidal neurons in williams syndrome: Prefrontal cortex and beyond," *Frontiers in neuroscience*, vol. 11, p. 419, 2017.

[136] C. Falcone *et al.*, "Cortical interlaminar astrocytes across the therian mammal radiation," *Journal of Comparative Neurology*, vol. 527, no. 10, pp. 1654–1674, 2019.

[137] J. Berg *et al.*, "Human cortical expansion involves diversification and specialization of supragranular intratelencephalic-projecting neurons," *BioRxiv*, 2020.

[138] R. Benavides-Piccione *et al.*, "Differential structure of hippocampal ca1 pyramidal neurons in the human and mouse," *Cerebral Cortex*, vol. 30, no. 2, pp. 730–752, 2020.

[139] C. Falcone *et al.*, "Cortical interlaminar astrocytes are generated prenatally, mature postnatally, and express unique markers in human and nonhuman primates," *Cerebral Cortex*, vol. 31, no. 1, pp. 379–395, 2021.

[140] B. Jacobs *et al.*, "Comparative morphology of gigantopyramidal neurons in primary motor cortex across mammals," *Journal of Comparative Neurology*, vol. 526, no. 3, pp. 496–536, 2018.

[141] A. Warling *et al.*, "Putative dendritic correlates of chronic traumatic encephalopathy: A preliminary quantitative golgi exploration," *Journal of Comparative Neurology*, vol. 529, no. 7, pp. 1308–1326, 2021.

[142] K. Anderson, E. Yamamoto, J. Kaplan, M. Hannan, and B. Jacobs, "Neurolucida lucivid versus neurolucida camera: A quantitative and qualitative comparison of three-dimensional neuronal reconstructions," *Journal of neuroscience methods*, vol. 186, no. 2, pp. 209–214, 2010.

[143] B. Jacobs *et al.*, "Regional dendritic and spine variation in human cerebral cortex: A quantitative golgi study," *Cerebral cortex*, vol. 11, no. 6, pp. 558–571, 2001.

[144] K. Anderson *et al.*, "The morphology of supragranular pyramidal neurons in the human insular cortex: A quantitative golgi study," *Cerebral Cortex*, vol. 19, no. 9, pp. 2131–2144, 2009.

[145] K. Travis, K. Ford, and B. Jacobs, "Regional dendritic variation in neonatal human cortex: A quantitative golgi study," *Developmental neuroscience*, vol. 27, no. 5, pp. 277–287, 2005.

[146] B. Jacobs *et al.*, "Comparative neuronal morphology of the cerebellar cortex in afrotherians, carnivores, cetartiodactyls, and primates," *Frontiers in neuroanatomy*, vol. 8, p. 24, 2014.

[147] D. Linaro *et al.*, "Xenotransplanted human cortical neurons reveal species-specific development and functional integration into mouse visual circuits," *Neuron*, vol. 104, no. 5, pp. 972–986, 2019.

[148] G. Eyal *et al.*, "Unique membrane properties and enhanced signal processing in human neocortical neurons," *Elife*, vol. 5, e16553, 2016.

[149] T. Chailangkarn *et al.*, "A human neurodevelopmental model for williams syndrome," *Nature*, vol. 536, no. 7616, pp. 338–343, 2016.

[150] M. Vukšić, Z. Petanjek, M. R. Rašin, and I. Kostović, "Perinatal growth of prefrontal layer iii pyramids in down syndrome," *Pediatric neurology*, vol. 27, no. 1, pp. 36–38, 2002.

[151] L. E. Westrum and T. W. Blackstad, "An electron microscopic study of the stratum radiatum of the rat hippocampus (regio superior, ca 1) with particular emphasis on synaptology," *Journal of Comparative Neurology*, vol. 119, no. 3, pp. 281–309, 1962.

[152] P Berbel and G. Innocenti, "The development of the corpus callosum in cats: A light-and electron-microscopic study," *Journal of Comparative Neurology*, vol. 276, no. 1, pp. 132–156, 1988.

[153] N. Ishizuka, J. Weber, and D. G. Amaral, "Organization of intrahippocampal projections originating from ca3 pyramidal cells in the rat," *Journal of comparative neurology*, vol. 295, no. 4, pp. 580–623, 1990.

[154] X.-G. Li, P Somogyi, A Ylinen, and G Buzsáki, "The hippocampal ca3 network: An in vivo intracellular labeling study," *Journal of comparative neurology*, vol. 339, no. 2, pp. 181–208, 1994.

[155] G. Major, A. U. Larkman, P. Jonas, B. Sakmann, and J. Jack, "Detailed passive cable models of whole-cell recorded ca3 pyramidal neurons in rat hippocampal slices," *Journal of Neuroscience*, vol. 14, no. 8, pp. 4613–4638, 1994.

[156] M. Palombo *et al.*, "New paradigm to assess brain cell morphology by diffusion-weighted mr spectroscopy in vivo," *Proceedings of the National Academy of Sciences*, vol. 113, no. 24, pp. 6671–6676, 2016.

[157] R. Scorcioni, S. Polavaram, and G. A. Ascoli, "L-measure: A web-accessible tool for the analysis, comparison and search of digital reconstructions of neuronal morphologies," *Nature protocols*, vol. 3, no. 5, pp. 866–876, 2008.

[158] C. Zhang and T. Chen, "Efficient feature extraction for 2d/3d objects in mesh representation," in *Proceedings 2001 International Conference on Image Processing (Cat. No. 01CH37205)*, IEEE, vol. 3, 2001, pp. 935–938.

[159] M Palombo, D. Alexander, and H Zhang, "Large-scale analysis of brain cell morphometry informs microstructure modelling of gray matter," in *Proc. Intl. Soc. Mag. Reson. Med*, vol. 29, 2021.

[160] K. Ginsburger, F. Matuschke, F. Poupon, J.-F. Mangin, M. Axer, and C. Poupon, "Medusa: A gpu-based tool to create realistic phantoms of the brain microstructure using tiny spheres," *NeuroImage*, vol. 193, pp. 10 – 24, 2019, issn: 1053-8119. doi: https://doi.org/10.1016/j.neuroimage.2019.02.055. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S105381191930151X.

[161] R. G. Thorne and C. Nicholson, "In vivo diffusion analysis with quantum dots and dextrans predicts the width of brain extracellular space," *Proceedings of the National Academy of Sciences*, vol. 103, no. 14, pp. 5567–5572, 2006.

[162] E. Syková and C. Nicholson, "Diffusion in brain extracellular space," *Physiological reviews*, vol. 88, no. 4, pp. 1277–1340, 2008.

[163] D Le Bihan, E Breton, D Lallemand, P Grenier, E Cabanis, and M Laval-Jeantet, "MR imaging of intravoxel incoherent motions: Application to diffusion and perfusion in neurologic disorders.," *Radiology*, vol. 161, no. 2, pp. 401–407, 1986. [Online]. Available: http://radiology.rsna.org/content/161/2/401.abstract.

[164] V. Y. Shifrin, P. G. Park, V. N. Khorev, C. H. Choi, and C. S. Kim, "A new low-field determination of the proton gyromagnetic ratio in water," *IEEE Transactions on Instrumentation and Measurement*, vol. 47, no. 3, pp. 638–643, 1998.

[165] K. R. Brownstein and C. Tarr, "Importance of classical diffusion in nmr studies of water in biological cells," *Physical review A*, vol. 19, no. 6, p. 2446, 1979.

[166] D. S. Grebenkov, "Nmr survey of reflected brownian motion," *Reviews of Modern Physics*, vol. 79, no. 3, p. 1077, 2007.

[167] D. S. Grebenkov, "Pulsed-gradient spin-echo monitoring of restricted diffusion in multilayered structures," *Journal of Magnetic Resonance*, vol. 205, no. 2, pp. 181–195, Aug. 2010, issn: 1090-7807. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1090780710001199.

[168] G. J. Stanisz, G. A. Wright, R. M. Henkelman, and A. Szafer, "An analytical model of restricted diffusion in bovine optic nerve," *Magn. Reson. Med.*, vol. 37, no. 1, pp. 103–111, 1997, issn: 1522-2594. doi: 10.1002/mrm.1910370115.

[169] G. J. Stanisz, "Diffusion mr in biological systems: Tissue compartments and exchange," *Israel journal of chemistry*, vol. 43, no. 1-2, pp. 33–44, 2003.

[170] S. Michaeli *et al.*, "Proton t2 relaxation study of water, n-acetylaspartate, and creatine in human brain using hahn and carr-purcell spin echoes at 4t and 7t," *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 47, no. 4, pp. 629–633, 2002.

[171] N. Shemesh *et al.*, "Conventions and nomenclature for double diffusion encoding nmr and mri," *Magnetic Resonance in Medicine*, vol. 75, no. 1, pp. 82–87, 2016. doi: 10.1002/mrm.25901. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.25901.

[172] N. Shemesh, E. Özarslan, P. J. Basser, and Y. Cohen, "Detecting diffusion-diffraction patterns in size distribution phantoms using double-pulsed field gradient nmr: Theory and experiments," *The Journal of chemical physics*, vol. 132, no. 3, p. 034 703, 2010.

[173] P. T. Callaghan and J. Stepianik, "Frequency-domain analysis of spin motion using modulated-gradient NMR," *Journal of Magnetic Resonance, Series A*, vol. 117, no. 1, pp. 118–122, Nov. 1995, issn: 1064-1858. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1064185885799597.

[174] M. D. Does, E. C. Parsons, and J. C. Gore, "Oscillating gradient measurements of water diffusion in normal and globally ischemic rat brain," *Magn. Reson. Med.*, vol. 49, no. 2, pp. 206–215, 2003, issn: 1522-2594. doi: 10.1002/mrm.10385.

[175] D.-M. Koh and D. J. Collins, "Diffusion-weighted mri in the body: Applications and challenges in oncology," *American Journal of Roentgenology*, vol. 188, no. 6, pp. 1622–1635, 2007.

[176] P. J. Basser, S. Pajevic, C. Pierpaoli, J. Duda, and A. Aldroubi, "In vivo fiber tractography using dt-mri data," *Magnetic Resonance in Medicine*, vol. 44, no. 4, pp. 625–632, 2000. [Online]. Available: http://dx.doi.org/10.1002/1522-2594(200010)44:4<625::AID-MRM17>3.0.CO;2-O.

[177] R. G. González, "Clinical mri of acute ischemic stroke," *Journal of Magnetic Resonance Imaging*, vol. 36, no. 2, pp. 259–271, 2012.

[178] N. Shemesh, C.-F. Westin, and Y. Cohen, "Magnetic resonance imaging by synergistic diffusion-diffraction patterns," *Physical review letters*, vol. 108, no. 5, p. 058 103, 2012.

[179] M. H. Blees, "The effect of finite duration of gradient pulses on the pulsed-field-gradient nmr method for studying restricted diffusion," *Journal of Magnetic Resonance, Series A*, vol. 109, no. 2, pp. 203–209, 1994. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1064185884711569.

[180] J. Kärger, "Nmr self-diffusion studies in heterogeneous systems," *Advances in Colloid and Interface Science*, vol. 23, pp. 129–148, 1985. doi: https://doi.org/10.1016/0001-8686(85)80018-X.

[181] N. F. Lori, T. E. Conturo, and D. Le Bihan, "Definition of displacement probability and diffusion time in q-space magnetic resonance measurements that use finite-duration diffusion-encoding gradients," *Journal of Magnetic Resonance*, vol. 165, no. 2, pp. 185–195, 2003.

[182] V. G. Kiselev, "Fundamentals of diffusion mri physics," *NMR in Biomedicine*, vol. 30, no. 3, e3602, 2017.

[183] P. Tofts *et al.*, "Test liquids for quantitative mri measurements of self-diffusion coefficient in vivo," *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, vol. 43, no. 3, pp. 368–374, 2000.

[184] M. Moseley *et al.*, "Diffusion-weighted mr imaging of acute stroke: Correlation with t2-weighted and magnetic susceptibility-enhanced mr imaging in cats.," *American Journal of Neuroradiology*, vol. 11, no. 3, pp. 423–429, 1990.

[185] D. V. Nguyen, J.-R. Li, D. Grebenkov, and D. L. Bihan, "A finite elements method to solve the bloch–torrey equation applied to diffusion magnetic resonance imaging," *Journal of Computational Physics*, vol. 263, no. 0, pp. 283 –302, 2014, issn: 0021-9991. doi: 10.1016/j.jcp.2014.01.009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0021999114000308.

[186] C. S. Landis *et al.*, "Determination of the mri contrast agent concentration time course in vivo following bolus injection: Effect of equilibrium transcytolemmal water exchange," *Magnetic resonance in medicine*, vol. 44, no. 4, pp. 563–574, 2000.

[187] D. M. Yang, J. E. Huettner, G. L. Bretthorst, J. J. Neil, J. R. Garbow, and J. J. Ackerman, "Intracellular water preexchange lifetime in neurons and astrocytes," *Magnetic resonance in medicine*, vol. 79, no. 3, pp. 1616–1627, 2018.

[188] I. O. Jelescu, A. de Skowronski, F. Geffroy, M. Palombo, and D. S. Novikov, "Neurite exchange imaging (nexi): A minimal model of diffusion in gray matter with inter-compartment water exchange," *NeuroImage*, vol. 256, p. 119 277, 2022.

[189] T. Rahman and J. Valdman, "Fast matlab assembly of fem matrices in 2d and 3d: Nodal elements," *Applied Mathematics and Computation*, vol. 219, no. 13, pp. 7151–7158, 2013.

[190] M. G. Larson and F. Bengzon, *The finite element method: theory, implementation, and applications*. Springer Science & Business Media, 2013, vol. 10.

[191] L. F. Shampine and M. W. Reichelt, "The matlab ode suite," *SIAM journal on scientific computing*, vol. 18, no. 1, pp. 1–22, 1997.

[192] W. A. Strauss, *Partial differential equations: An introduction*. John Wiley & Sons, 2007.

[193] D. S. Grebenkov and B.-T. Nguyen, "Geometrical structure of laplacian eigenfunctions," *siam REVIEW*, vol. 55, no. 4, pp. 601–667, 2013.

[194] B. N. Parlett, *The symmetric eigenvalue problem*. SIAM, 1998.

[195] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted Arnoldi methods*. SIAM, 1998.

[196] G. W. Stewart, "A krylov–schur algorithm for large eigenproblems," *SIAM Journal on Matrix Analysis and Applications*, vol. 23, no. 3, pp. 601–614, 2002.

[197] N. J. Higham, "The scaling and squaring method for the matrix exponential revisited," *SIAM Journal on Matrix Analysis and Applications*, vol. 26, no. 4, pp. 1179–1193, 2005.

[198] N. J. Higham, "The scaling and squaring method for the matrix exponential revisited," *SIAM review*, vol. 51, no. 4, pp. 747–764, 2009.

[199] A. H. Al-Mohy and N. J. Higham, "Computing the action of the matrix exponential, with an application to exponential integrators," *SIAM journal on scientific computing*, vol. 33, no. 2, pp. 488–511, 2011.

[200] A. H. Al-Mohy and N. J. Higham, "A new scaling and squaring algorithm for the matrix exponential," *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 3, pp. 970–989, 2010.

[201] C. Fang, D. Wassermann, and J.-R. Li, "Fourier representation of the diffusion mri signal using layer potentials," *SIAM Journal on Applied Mathematics*, vol. 83, no. 1, pp. 99–121, 2023.

[202] A. Vretblad, *Fourier analysis and its applications*. Springer Science & Business Media, 2003, vol. 223.

[203] L. Greengard and J. Strain, "A fast algorithm for the evaluation of heat potentials," *Comm. Pure Appl. Math.*, vol. 43, no. 8, pp. 949–963, 1990, issn: 0010-3640.

[204] J.-R. Li and L. Greengard, "On the numerical solution of the heat equation i: Fast solvers in free space," *Journal of Computational Physics*, vol. 226, no. 2, pp. 1891–1901, Oct. 2007. doi: https://doi.org/10.1016/j.jcp.2007.06.021.

[205] J.-R. Li and L. Greengard, "High order accurate methods for the evaluation of layer heat potentials," *SIAM J. Sci. Comput.*, vol. 31, no. 5, pp. 3847–3860, 2009, issn: 1064-8275. doi: 10.1137/080732389.

[206] H. Haddar, J.-R. Li, and S. Schiavi, "Understanding the time-dependent effective diffusion coefficient measured by diffusion mri: The intracellular case," *SIAM Journal on Applied Mathematics*, vol. 78, no. 2, pp. 774–800, 2018.

[207] S. G. Johnson, *Notes on the convergence of trapezoidal-rule quadrature*, 2010.

[208] A. Zaimi, M. Wabartha, V. Herman, P.-L. Antonsanti, C. S. Perone, and J. Cohen-Adad, "AxonDeepSeg: Automatic axon and myelin segmentation from microscopy data using convolutional neural networks," en, *Scientific Reports*, vol. 8, no. 1, p. 3816, Feb. 2018, Number: 1 Publisher: Nature Publishing Group, issn: 2045-2322. doi: 10.1038/s41598-018-22181-4. [Online]. Available: https://www.nature.com/articles/s41598-018-22181-4 (visited on 04/22/2022).

[209] M. Nordin, M. Nilsson-Jacobi, and M. Nydén, "A mixed basis approach in the sgp-limit," *Journal of magnetic resonance*, vol. 212, no. 2, pp. 274–279, 2011.

[210] M. Nordin, D. Grebenkov, M. N. Jacobi, and M. Nydén, "An efficient eigenfunction approach to calculate spin-echo signals in heterogeneous porous media," *Microporous and mesoporous materials*, vol. 178, pp. 7–10, 2013.

[211] M. Zucchelli, S. Deslauriers-Gauthier, and R. Deriche, "A computational framework for generating rotation invariant features and its application in diffusion mri," *Medical image analysis*, vol. 60, p. 101 597, 2020.

[212] D. S. Novikov, J. Veraart, I. O. Jelescu, and E. Fieremans, "Rotationally-invariant mapping of scalar and orientational metrics of neuronal microstructure with diffusion mri," *NeuroImage*, vol. 174, pp. 518–538, 2018.

[213] P. W. Kuchel and C. J. Durrant, "Permeability coefficients from nmr q-space data: Models with unevenly spaced semi-permeable parallel membranes," *Journal of Magnetic Resonance*, vol. 139, no. 2, pp. 258–272, 1999.

[214] S. Merlet, E. Caruyer, A. Ghosh, and R. Deriche, "A computational diffusion mri and parametric dictionary learning framework for modeling the diffusion signal and its features," *Medical Image Analysis*, vol. 17, no. 7, pp. 830–843, 2013.

[215] D. Karimi *et al.*, "A machine learning-based method for estimating the number and orientations of major fascicles in diffusion-weighted magnetic resonance imaging," *Medical Image Analysis*, vol. 72, p. 102 129, 2021.

[216] I. Hill *et al.*, "Machine learning based white matter models with permeability: An experimental study in cuprizone treated in-vivo mouse model of axonal demyelination," *NeuroImage*, vol. 224, p. 117 425, 2021.

[217] M. Jallais, P. L. Rodrigues, A. Gramfort, and D. Wassermann, "Cytoarchitecture measurements in brain gray matter using likelihood-free inference," in *International Conference on Information Processing in Medical Imaging*, Springer, 2021, pp. 191–202.

[218] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[219] A. R. Barron, "Approximation and estimation bounds for artificial neural networks," *Machine learning*, vol. 14, no. 1, pp. 115–133, 1994.

[220] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, "The expressive power of neural networks: A view from the width," *Advances in neural information processing systems*, vol. 30, 2017.

[221] T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, and Q. Liao, "Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review," *International Journal of Automation and Computing*, vol. 14, no. 5, pp. 503–519, 2017.

[222] D. L. Donoho *et al.*, "High-dimensional data analysis: The curses and blessings of dimensionality," *AMS math challenges lecture*, vol. 1, no. 2000, p. 32, 2000.

[223] B. Keil *et al.*, "A 64-channel 3t array coil for accelerated brain mri," *Magnetic resonance in medicine*, vol. 70, no. 1, pp. 248–258, 2013.

[224] A. F. Howard *et al.*, "Estimating axial diffusivity in the noddi model," *bioRxiv*, pp. 2020–10, 2022.

[225] V. A. Shepherd, "The cytomatrix as a cooperative system of macromolecular and water networks," *Current topics in developmental biology*, vol. 75, pp. 171–223, 2006.

[226] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[227] E. T. McKinnon, J. H. Jensen, G. R. Glenn, and J. A. Helpern, "Dependence on b-value of the direction-averaged diffusion-weighted imaging signal in brain," *Magnetic resonance imaging*, vol. 36, pp. 121–127, 2017.

[228] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[229] G. E. Hinton, "Connectionist learning procedures," in *Machine learning*, Elsevier, 1990, pp. 555–610.

[230] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[231] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[232] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[233] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[234] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[235] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," *arXiv preprint arXiv:1904.09237*, 2019.

[236] J. Bridle, "Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters," *Advances in neural information processing systems*, vol. 2, 1989.

[237] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[238] P. Virtanen *et al.*, "Scipy 1.0: Fundamental algorithms for scientific computing in python," *Nature methods*, vol. 17, no. 3, pp. 261–272, 2020.

[239] C. De Boor and C. De Boor, *A practical guide to splines*. springer-verlag New York, 1978, vol. 27.

[240] D. H. Wolpert, "The Lack of A Priori Distinctions Between Learning Algorithms," *Neural Computation*, vol. 8, no. 7, pp. 1341–1390, Oct. 1996, issn: 0899-7667. doi: 10.1162/neco.1996.8.7.1341. eprint: https://direct.mit.edu/neco/article-pdf/8/7/1341/813495/neco.1996.8.7.1341.pdf. [Online]. Available: https://doi.org/10.1162/neco.1996.8.7.1341.

[241] N. G. Gyori, M. Palombo, C. A. Clark, H. Zhang, and D. C. Alexander, "Training data distribution significantly impacts the estimation of tissue microstructure with machine learning," *Magnetic resonance in medicine*, vol. 87, no. 2, pp. 932–947, 2022.

[242] U. Ferizi *et al.*, "Diffusion mri microstructure models with in vivo human brain connectome data: Results from a multi-group comparison," *NMR in Biomedicine*, vol. 30, no. 9, e3734, 2017.

[243] A. De Luca *et al.*, "On the generalizability of diffusion mri signal representations across acquisition parameters, sequences and tissue types: Chronicles of the memento challenge," *NeuroImage*, vol. 240, p. 118 367, 2021.

[244] S. Coelho *et al.*, "Reproducibility of the standard model of diffusion in white matter on clinical mri systems," *NeuroImage*, vol. 257, p. 119 290, 2022.

[245] D. C. Alexander *et al.*, "Orientationally invariant indices of axon diameter and density from diffusion mri," *Neuroimage*, vol. 52, no. 4, pp. 1374–1389, 2010.

[246] H.-H. Lee *et al.*, "Along-axon diameter variation and axonal orientation dispersion revealed with 3d electron microscopy: Implications for quantifying brain white matter microstructure with histology and diffusion mri," *Brain Structure and Function*, vol. 224, pp. 1469–1488, 2019.

[247] H.-H. Lee, S. N. Jespersen, E. Fieremans, and D. S. Novikov, "The impact of realistic axonal shape on axon diameter estimation using diffusion mri," *Neuroimage*, vol. 223, p. 117 228, 2020.

[248] C. S. Von Bartheld, J. Bahney, and S. Herculano-Houzel, "The search for true numbers of neurons and glial cells in the human brain: A review of 150 years of cell counting," *Journal of Comparative Neurology*, vol. 524, no. 18, pp. 3865–3895, 2016.

[249] B. Fischl and A. M. Dale, "Measuring the thickness of the human cerebral cortex from magnetic resonance images," *Proceedings of the National Academy of Sciences*, vol. 97, no. 20, pp. 11 050–11 055, 2000.