Computational Approaches to Bridge Experimental and Simulation Viewpoints in Neuroscience

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

Yves Bernaerts

aus Antwerpen, Belgien

Tübingen

2023

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	20.12.2023
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter/-in:	Prof. Dr. Philipp Berens
2. Berichterstatter/-in:	Prof. Dr. Philipp Hennig

Abstract

The nervous system constitutes the highway of communication in living animals. Information propagates in both bottom-up fashion, that is from the periphery of the body to the brain, and top-down, from the brain back to the body. It is the nervous system's foundational units, the neurons, that serve as computational nodes in this bidirectional sophisticated design.

Neurons reside in biological organisms, yet the nature of their electrical responses are reminiscent of systems in engineering and physics, their communication through neurotransmitters of diffusion models in chemistry, and their interconnectivity patterns of neural networks in machine learning. The field of neuroscience develops increasingly interdisciplinary, yet we still lack methods that meaningfully bridge them.

To appreciate the importance of a bridge between different viewpoints in neuroscience, we can be inspired by the 'cortical tree of neural types'. Even though the notion of neural types in neuroscience is fairly established, neurons have been observed to vary both discretely and continuously in both genetic and phenotypic landscapes, making their classification in 'types', as discrete entities conforming to both modalities, difficult.

Statistical models that bridge the genetic landscape to phenotypic modalities including electrophysiology and morphology could help us build a multimodal neuronal taxonomy. Yet, even though many exist for the unidimensional ('one-view') analysis of cells in the nervous system, we currently still lack 'joint-view' statistical tools that can for instance predict one modality from another or produce joint-view two-dimensional embeddings for the exploratory analysis of neuronal data sets.

Biophysical models move one step further. They do not merely describe the quantitative relationship between modalities, but allow for a peek into the black-box transformation from one to the other. They can describe *how* the electrophysiology comes to be in neurons based on ion channel abundance in the cell's membrane, the latter determined by the expression of specific genes. Reliably inferring probability distributions over parameters of biophysical models that reproduce real-world observations in electrophysiology, has however also proved challenging. While recent advances in inference methods have demonstratively shown useful for synthetic simulated data, they remain difficult to generalize to experimental data.

In this thesis, we show how recent advances in machine learning can bring both statistical as well as biophysical models, to the next level. We present sparse bottleneck neural networks that select specific genes to nonlinearly predict electrophysiological measurements and outperform linear methods on the prediction task. They furthermore produce two-dimensional joint-view embeddings that are directly interpretable in biology.

We furthermore introduce a slight manipulation in training strategy for neural density estimators, causing the inference of biophysical model parameters to be much more reliable when it is applied to real-world electrophysiological data.

Finally, as a first attempt to bridge the genetic landscape with electrophysiological behavior in neurons, we show how sparse reduced-rank regression intuitively selects specific genes to predict fitted parameter values of biophysical models that replicate real-world neuronal electrophysiology.

Zusammenfassung

Das Nervensystem bildet die Grundlage für einen der wichtigsten Kommunikationskanäle der Lebewesen. Informationen werden sowohl 'top-down', d. h. von der Peripherie des Körpers zum Gehirn, als auch 'bottom-up', d. h. vom Gehirn zurück zum Körper, weitergegeben. Spezifische Zellen, Neuronen genannt, bilden die Grundeinheiten des Nervensystems, die die Rechenknoten dieses bidirektionalen raffinierten Designs darstellen.

Obwohl Neuronen traditionell nur mit biologischen Organismen in Verbindung gebracht werden, ähnelt die Art ihrer elektrischen Reaktionen Systemen in Technik und Physik, ihre Kommunikation über Neurotransmitter Diffusionsmodellen in der Chemie und ihre Vernetzungsmuster neuronalen Netzen im maschinellen Lernen. Das Gebiet der Neurowissenschaften entwickelt sich zunehmend interdisziplinär, doch fehlt es noch immer an Methoden, die eine sinnvolle Brücke zwischen ihnen bilden.

Um zu verstehen, wie wichtig eine Brücke zwischen den unterschiedlichen Sichtweisen in den Neurowissenschaften ist, können wir uns vom "kortikalen Baum der neuronalen Typen" inspirieren lassen. Der Begriff der neuronalen Typen in den Neurowissenschaften hat sich zwar relativ weitgehend etabliert, doch wurde beobachtet dass Neuronen sowohl diskret als auch kontinuierlich in der genetischen und phänotypischen Landschaft variieren. Dies erschwert ihre Klassifizierung in 'Typen' als diskrete Einheiten.

Statistische Modelle könnten eine Brücke zwischen der genetischen Landschaft und den phänotypischen Modalitäten einschließlich Elektrophysiologie und Morphologie schlagen, und somit eine multimodale neuronale Taxonomie erstellen. Obwohl es bereits viele Modelle für die eindimensionale Analyse von Zellen im Nervensystem gibt, fehlt es derzeit noch an statistischen Werkzeugen für eine umfassende mehrdimensionale Analyse. Hierzu gehören z.b. die Möglichkeit eine Modalität aus einer Anderen vorherzusagen oder zweidimensionale Darstellungen für die explorative Analyse neuronaler Datensätze erstellen zu können.

Biophysikalische Modelle gehen noch einen Schritt weiter. Sie beschreiben nicht nur die quantitative Beziehung zwischen den Modalitäten, sondern erlauben auch einen Blick in die 'Black-box'-Transformation von einer Modalität zu einer Anderen. Sie sind in der Lage, die Elektrophysiologie in Neuronen anhand der Dichte der Ionenkanäle in der Zellmembran zu beschreiben. Die Dichte der Ionenkanäle wird von der Genenexpression bestimmt. Es hat sich jedoch als schwierig erwiesen, zuverlässige Wahrscheinlichkeitsverteilungen aus biophysikalischen Modellparametern abzuleiten, die reale Beobachtungen in der Elektrophysiologie nachbilden. Die neuesten Erkenntnisse auf dem Gebiet der Inferenzmethoden haben sich für synthetische und simulierte Daten als hilfreich erwiesen, sie lassen sich jedoch nur schwer auf experimentelle Daten verallgemeinern.

In dieser Arbeit zeigen wir, wie die neuesten Erkenntnisse im Bereich des maschinellen Lernens genutzt werden können, um sowohl statistische als auch biophysikalische Modelle weiterzuentwickeln. Wir stellen neuronale Netze mit 'Bottlenecks' vor, die in der Lage sind, spezifische Gene zu selektieren und damit elektrophysiologische Messungen nichtlinear vorherzusagen. Wir zeigen auch, dass diese Netze lineare Methoden in der Vorhersagegenauigkeit übertreffen. Darüber hinaus erzeugen sie zweidimensionale, umfassende Darstellungen, die direkt in der Biologie interpretiert werden können.

Darüber hinaus haben wir eine leichte Manipulation der Trainingsstrategie für neuronale 'Density Networks' eingeführt. Dies führte zu einer wesentlich höheren Zuverlässigkeit der Ableitung von biophysikalischen Modellparametern in Anwendung auf reale elektrophysiologische Daten.

Schließlich zeigen wir in einem ersten Versuch, eine Brücke zwischen der genetischen Landschaft und dem elektrophysiologischen Verhalten von Neuronen zu schlagen, wie 'sparse reduced-rank regression' intuitiv spezifische Gene selektiert und somit die Parameterwerte biophysikalischer Modelle vorhersagt, die die reale neuronale Elektrophysiologie nachbilden.

Acknowledgements

This thesis would not have been possible if it were not for much needed support, provided knowingly or unknowingly, directly or indirectly.

First, I would like to express my sincere gratitude to Prof. Dr. Philipp Berens and Dr. Dmitry Kobak. You directly advised, mentored and guided me in the world of science. You helped me in rigorous data analysis, data visualization and henceforth helped me grow in scientific independence in order to supervise my own projects. Importantly, I would like to thank you for your patience and trust. Like in any PhD, the problems were not easily solved, took time, energy and a lot of back-and-forth communication.

I thank Prof. Dr. Philipp Hennig and Prof. Dr. Martin Giese for agreeing to be part of my thesis advisory committee. The sessions together provided for much scientific insight and needed support. I also thank external collaborators Prof. Dr. Andreas S. Tolias and Prof. Dr. Jakob Macke for guidance at crucial moments in the PhD.

This PhD was of a very interdisciplinary nature. My gratitude therefore goes to many collaborators including (but not limited to) Dr. Federico Scala, Dr. Pedro J. Gonçalvez, Michael Deistler, Dr. Sophie Laturnus and Dr. Ziwei Huang. I learned a lot from you, thank you!

I deeply appreciated the nurturing environment in the Berenslab, for both scientific and social settings. To whom I have therefore not already mentioned, Dr. Murat Seckin Ayhan, Lisa Smors, Dr. Cornelius Schröder, Sarah Strauß as well as all past and current members, I say cheers. Cheers to allowing me to pester you from time to time, and cheers to lovely moments spent together including lunch and coffee breaks together in Tübingen's magnificent old town, sporty running events, and winter 'cold dips' in the Neckarriver.

I also thank all the people involved in the International Max Planck Research Schools for Intelligent Systems (IMPRS-IS), a school without which I would have not been exposed to so many fields in Machine learning. Thank you for your coordination, Leila Masri.

Finally, I would like to express my sincerest and deepest gratitude to many wonderful friends I have met during my time in Tübingen. Naray, Louise, Naila, Chiara, Arijit, Mehrdad, Dania, Dmitri, Anna-Lena, Lena, Barçin, Albachiara, Gianluca, Giacomo, Ernest, Chris, Väinö, Oskari, Noah, Andreas, Pablo, Dominique, Michael, Julius, Ahmed, Valentyn, Marek, Joseph, Vjosa, Valeria, Vincent, Noémie, Andreas, Cleo, Ina, Julia, Joana, Cristina, Carmen, Annalena, Emma, Blair and many others, you have been my bedrocks throughout the PhD, thank you.

I also thank the 'fysicaboys' Gregory, Siemen, Robin, Nard and Ward. We embarked on a Bachelor in physics together, and jointly got to the point of finishing a PhD in fields ranging from cryptography, quantum computing, medical proton radiation therapy, astrophysics and neural science. Cheers!

Finally, I thank my family whose warm support was vital to conduct the PhD abroad, as well as friends from Belgium including (but not limited to) Isabeau, Jana, Andrey, Seppe, Ynke, Jordi and Thomas. You would welcome me back home in Belgium for a needed break, at any time.

I thank you all from the bottom of my heart!

Contents

Notation

This section provides the reader with a reference for the notation used throughout this thesis. Placeholder symbols are introduced in Numbers and Arrays to denote types including scalars, vectors and matrices. We then introduce some useful operators from Linear Algebra, Calculus and Probability Theory.

Numbers and Arrays

a	A scalar.
$\mathbf{a}$	A vector.
$\mathbf{A}$	A matrix.
a_i	The $i-$th element in vector $\mathbf{a}$.
A_{ij}	Matrix element of $\mathbf{A}$ at the $i-$th row and $j-$th column.
$A_{i:}$	The $i-$th row of $\mathbf{A}$. Analogous for a column.
$\mathbf{a}^{(n)}$	The $n-$th vector element of some set, for instance a training data set.
$\mathbf{A}^{(n)}$	The $n-$th matrix element of some set, for instance a set of weights in a deep neural network.

Linear Algebra

$\mathbf{A}^\top$	Transpose of matrix $\mathbf{A}$.
$\mathbf{A}^{-1}$	Inverse of matrix $\mathbf{A}$.
$\det(\mathbf{A})$	Determinant of matrix $\mathbf{A}$.
$\|\mathbf{a}\|_p$	$\mathscr{L}^p$-norm of vector $\mathbf{a}$. When $p = 2$, we will often omit p in the expression.
$\|\mathbf{A}\|_p$	$\mathscr{L}^p$-norm of matrix $\mathbf{A}$.

Calculus

$f(\mathbf{a}; \boldsymbol{\theta})$	A scalar-valued function f of $\mathbf{a}$ parameterized by $\boldsymbol{\theta}$.
$\frac{df}{da}$	Derivative of f with respect to a.
$\frac{\partial f}{\partial a}$	Partial derivative of f with respect to a.
$\mid a \mid$	Absolute value of a.
$\sum_{n=1}^{N}(\mathbf{a}^{(n)})$	Sum of n (data) samples a_n.
$g \circ f$	The composition of functions g and f, i.e. $g(f(\cdot))$.

Probability Theory

$p(\mathbf{a})$	Probability distribution over random variable $\mathbf{a}$.
$\mathbf{a} \sim p(\mathbf{a})$	$\mathbf{a}$ is distributed according to p.
$p(\mathbf{a} \mid \mathbf{b})$	Conditional probability distribution p of $\mathbf{a}$ given $\mathbf{b}$.
$\mathbb{E}_{p(\mathbf{a})}(f(\mathbf{a}))$	Expectation of $f(\mathbf{a})$ when $\mathbf{a}$ is distributed according to p.
$\mathcal{KL}(p \mid\mid q)$	The Kullback-Leibler divergence of distributions p and q.

List of Figures

List of Tables

Prologue

1 Introduction

The nervous system constitutes the foundation for communication inside living organisms. In a bottom-up fashion, the body uses the nervous system to communicate what is happening in our inside and outside perceived worlds to the brain. In a top-down fashion, the brain uses the same nervous system to communicate to the body and respond to what has been perceived, possibly with action [1, 2]. For instance, if you sit in an ice bath, cold gets translated into an electrical signal that propagates to the somatosensory cortex in the brain allowing you to perceive temperature. It also propagates to the periaqueductal gray or pain center of the brain, however, as the cold might not merely be felt as cold, but painful too. Depending on your internal state, you might take action: you immediately get out of the ice bath. Alternatively, you might respond with non-action, and be accepting of the situation, at least for a while [3].

Feeling, perceiving, responding and taking action is only possible because of the nervous system's intricate design. In the case of humans, the brain's makeup consists of over 86 billion neurons [4] that span a variety of neuronal families with distinct phenomenology and genetic makeup. Within families, both discrete and continuous variation in gene expression (genotype) as well as electrophysiology and morphology (phenotype) has been observed [5–11]. Such complexities in multi-modal correspondence of cells on a finer 'cell type' level has kept the debate on neuronal classification ongoing for decades [12–14]. The study of the nervous system's foundational level, therefore needs multimodal experimental and statistical tools, embracing the complex variation in both genetic and phenotypic neuronal landscapes.

1.1 The BRAIN Initiative for Cell Census Network

In the spirit of understanding emerging behavior starting at the neuronal level, the Obama administration launched the Brain Research Through Advancing Innovative Neurotechnologies or BRAIN initiative through the National Institutes of Health or NIH. With this initiative, researchers aim to further accelerate the development of innovative technologies in order to revolutionize our understanding of the brain. From describing the brain on a single-cell level, to deriving computations in neuronal circuits, to understanding emerging behavior, this initiative tries to tackle long-standing questions about ourselves: what we are made up of and how that pertains to how we perceive, store information, communicate and act. Moreover, scientists hope that new discoveries in understanding our own brain's computations in this way could lead to new hypotheses in disease: by unraveling how single cells and complex neural circuits compute in healthy individuals, we can possibly understand when and how disease emerges from uncommon single-cell profiles and disharmonious circuit computations.

[1]: Taylor et al. (2010), 'Top-Down and Bottom-Up Mechanisms in Mind-Body Medicine: Development of an Integrative Framework for Psychophysiological Research'
[2]: Rauss et al. (2013), 'What is Bottom-Up and What is Top-Down in Predictive Coding?'
[3]: Muzik et al. (2018), '"Brain over body"–A study on the willful regulation of autonomic function during cold exposure'
[4]: Herculano-Houzel (2009), 'The human brain in numbers: a linearly scaled-up primate brain'
[5]: Tremblay et al. (2016), 'GABAergic interneurons in the neocortex: from cellular properties to circuits'
[6]: Harris et al. (2015), 'The neocortical circuit: themes and variations'
[7]: Tasic et al. (2016), 'Adult mouse cortical cell taxonomy revealed by single cell transcriptomics'
[8]: Tasic et al. (2018), 'Shared and distinct transcriptomic cell types across neocortical areas'
[9]: Markram et al. (2015), 'Reconstruction and simulation of neocortical microcircuitry'
[10]: Jiang et al. (2015), 'Principles of connectivity among morphologically defined cell types in adult neocortex'
[11]: Scala et al. (2021), 'Phenotypic variation of transcriptomic cell types in mouse motor cortex'
[12]: Ascoli et al. (2008), 'Petilla terminology: nomenclature of features of GABAergic interneurons of the cerebral cortex'
[13]: Callaway et al. (2021), 'A multimodal cell census and atlas of the mammalian primary motor cortex'
[14]: Zeng et al. (2017), 'Neuronal cell-type classification: challenges, opportunities and the path forward'

Incentivized by such aspirations, within the BRAIN initiative, the BRAIN Initiative for Cell Census Network or BICCN aims to provide a comprehensive cell atlas with descriptions of each neuronal cell type based not only on their transcriptomic makeup but on their morphology, connectivity and electrophysiology as well. Indeed, a comprehensive understanding of cell types in the brain is thought to enhance our understanding on how neuronal circuits generate perception and complex behavior.

1.2 Machine learning in Neural Science

Under the BICCN's umbrella, this thesis sets out to understand the computational units, the building blocks of our nervous system better. We will see how both statistical models as well as biophysical models prove beneficial to tackle some of these long-standing questions. Statistical models will be introduced to understand how gene expression levels in a neuron can be used to predict electrophysiological measurements, thereby bridging the genetic makeup of a neuron with its phenotype. With biophysical models, we attempt to move one step further. Beyond learning a descriptive relationship, we use determinism in a biophysical model to have a peek into *how* ion channels in the cell membrane, coded for by specific genes, give rise to rapid fluctuations in the cell's membrane voltage, i.e. its electrophysiology, by opening and closing their pores in response to certain stimuli. Biophysical models thus allow us to understand a neuron's computational framework, beyond its phenotypical and genetic description.

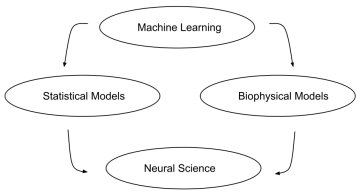

Figure 1.1: Machine learning in neural science, sketch. Machine learning can bring both statistical models and biophysical models in neural science to the next level.

Machine learning can be used to take both statistical and biophysical modeling approaches to the next level (Figure 1.1). Machine learning has recently seen a rapid rise in usage for various disciplines in science, including neural science. With the availability of increasingly bigger and publicly available data sets, as well as improving computing infrastructure and flexible software frameworks, machine learning has gained widespread adoption in for instance (but not limited to) image production [15], speech recognition [16] and the prediction of protein structure [17]. Here too, we will capitalize on recent advances in deep neural networks, to support both statistical and biophysical modeling approaches in neural science.

[15]: Karras et al. (2018), 'Progressive Growing of GANs for Improved Quality, Stability, and Variation'

[16]: Amodei et al. (2016), 'Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin'

[17]: Jumper et al. (2021), 'Highly accurate protein structure prediction with AlphaFold'

1.3 Outline

The **PRELIMINARIES** part consists of the bulk of methods and key concepts necessary to understand the thesis.

- Chapter 2 introduces the main data set studied throughout the thesis, obtained with a recently developed experimental techniques called Patch-seq. It further discusses a pipeline that automatically computes expert-defined features from raw electrophysiological recordings.

- Chapter 3 introduces *sparse reduced-rank regression*, a linear and intuitive statistical modeling approach to predict electrophysiological measurements from the gene expression levels in a cell, as well as produce biologically interpretable joint-view two-dimensional embeddings.
- Chapter 4 will expand on Chapter 3 with a nonlinear natural extension: *sparse bottleneck neural networks*, a machine learning framework that outperforms its linear counterpart on both the prediction task and interpretability of joint-view embeddings.
- Chapter 5 talks about the Hodgkin-Huxley model and how it served neuroscientists for decades to understand neuronal computations in the brain. We will introduce a minimal adaption of the model, sufficiently flexible to capture the electrophysiological variation observed in this Patch-seq data set.
- Chapter 6 introduces the field of *simulation-based inference*, another machine learning framework for inferring probability distributions over model parameters, consistent with electrophysiological data.

In **BRIDGING SIMULATION AND EXPERIMENTAL VIEWPOINTS IN NEUROSCIENCE**, we discuss the main contributions of this thesis, largely based on recent work (see Section 1.4). It consists of:

- Chapter 7 that introduces *neural posterior estimation with noise*, an adapted framework for the inference of posterior distributions over model parameters. It introduces noise to the summarizing statistics of simulations, with which the deep neural network is trained, in order for it to generalize to real-world experimental data such as neuronal electrophysiology. We will obtain reliable Hodgkin-Huxley model parameter combinations capable of producing simulations matching electrophysiology obtained with Patch-seq.
- Chapter 8 will use *sparse reduced-rank regression*, introduced in Chapter 3 to linearly predict Hodgkin-Huxley-based model parameters based on gene expression levels, effectively bridging the genetic makeup with parameters of a biophysical model that deterministically explains experimentally observed electrophysiology in the cell.

Chapter 9 in **CONCLUSION AND OUTLOOK** summarizes the main contributions of this thesis, and discusses promising avenues for the future understanding of our own nervous system with advances in machine learning.

1.4 List of Publications

This thesis, in smaller and greater extents, refers to both peer-reviewed and unpublished articles outlined here:

1. **Yves Bernaerts**, Michael Deistler, Pedro J. Gonçalves, Jonas Beck, Marcel Stimberg, Federico Scala, Andreas S. Tolias, Jakob Macke, Dmitry Kobak, and Philipp Berens. 'Combined statistical-mechanistic modeling links ion channel genes to physiology of cortical neuron types'. In: *bioRxiv* (2023)

2. **Yves Bernaerts**, Philipp Berens, and Dmitry Kobak. 'Sparse bottleneck neural networks for exploratory non-linear visualization of Patch-seq data'. In: *ArXiv* (2022)
3. Jonas Beck, Michael Deistler, **Yves Bernaerts**, Jakob Macke, and Philipp Berens. 'Efficient identification of informative features in simulation-based inference'. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2022
4. Dmitry Kobak, **Yves Bernaerts**, Marissa A. Weis, Federico Scala, Andreas Tolias, and Philipp Berens. 'Sparse reduced-rank regression for exploratory visualization of paired multivariate data'. In: *Journal of the Royal Statistical Society, Series C* (2021)
5. Federico Scala, Dmitry Kobak, Matteo Bernabucci, **Yves Bernaerts**, Cathryn René Cadwell, Jesus Ramon Castro, Leonard Hartmanis, Xiaolong Jiang, Sophie Laturnus, Elanine Miranda, et al. 'Phenotypic variation of transcriptomic cell types in mouse motor cortex'. In: *Nature* 598.7879 (2021)
6. Federico Scala, Dmitry Kobak, Shen Shan, **Yves Bernaerts**, Sophie Laturnus, Cathryn Rene Cadwell, Leonard Hartmanis, Emmanouil Froudarakis, Jesus Ramon Castro, Zheng Huan Tan, et al. 'Layer 4 of mouse neocortex differs in cell types and circuit organization between sensory areas'. In: *Nature Communications* 10 (2019)
7. Edward M. Callaway et al. 'A multimodal cell census and atlas of the mammalian primary motor cortex'. In: *Nature* 598.7879 (2021)

Preliminaries

Automated Extraction of Electrophysiological Features

2

The electrophysiology of neurons constitutes a data modality of high interest to neuroscientists because it describes one of the most important characteristics of the nervous system: the propagation of electrical signals. Neurons receive electrical signals in the form of excitatory and inhibitory synaptic potentials from neighboring cells at their dendrites, cellular protrusions that extend outwards form their center, the soma. At the soma, the cell sums up the dendritic input reaching a possible threshold that manifests as a membrane voltage, for which if the summed input goes beyond it, the cell will generate rapid nonlinear membrane voltage fluctuations, called action potentials (APs). These potentials continue to propagate through the cell in protrusions morphologically similar to dendrites called the axons of the neuron. Finally, axons synapse onto other neighboring cells thereby further conducting the electrical signal throughout the nervous system.

As electrophysiology therefore characterizes the important communication aspects in the nervous systems, we need carefully crafted features that describe the summarizing statistics of electrical signals. In Chapter 2, we therefore turn to the automatic extraction of expert-defined features summarizing the membrane voltage responses to current injection at the soma. Before we do so, however, we discuss how the data was obtained with a recently developed experimental technique called Patch-seq.

2.1 Patch-seq

Patch-seq combines electrophysiological recordings, single-cell RNA sequencing and morphological reconstruction in individual neurons [23–26]. It therefore embraces the multimodal nature of cells in the nervous systems and allows for their joint-view study. We will first turn to electrophysiology, however, obtained with Patch-seq, and come back to genetic features in later Chapters.

[23]: Cadwell et al. (2016), 'Electrophysiological, transcriptomic and morphologic profiling of single neurons using Patch-seq'
[24]: Cadwell et al. (2017), 'Multimodal profiling of single-cell morphology, electrophysiology, and gene expression using Patch-seq'
[25]: Fuzik et al. (2016), 'Integration of electrophysiological recordings with single-cell RNA-seq data identifies neuronal subtypes'
[26]: Lipovsek et al. (2021), 'Patch-seq: Past, present, and future'

In the Patch-seq study of Scala et al. [11], researchers investigated $n \sim 1300$ cells from adult mouse motor cortex (MoP). The experimenter first injects strong negative currents (starting for instance around -200 pA) followed by steps of 20 pA, each time during 600 ms time windows until too strong positive currents (such as 400 pA) are applied and the cell stops eliciting APs. The experimenter records hyperpolarizing membrane voltage responses (increasing the negative polarity between inner and outer environment of the cell) and depolarizing responses (decreasing the polarity) in response to negative and positive current injections, respectively Figure 2.1. When the cell's membrane voltage depolarizes and reaches a certain threshold, the cell generates APs Figure 2.1. We thus end up with multiple recordings — membrane voltage traces — that depict the cell's response over time to current injections of different size. These responses vary across neuronal families and cell types, and

therefore warrant an expert definition of electrophysiological features reflecting their variability.

2.2 A Python-based Pipeline to Extract Them All

Electrophysiologists choose a set of features that captures the characteristics of membrane voltage responses observed in neurons. Firstly, of great interest, are the shape of individual APs including their width, amplitude and upstroke. Beyond individual AP properties, passive properties in traces showing no APs include for instance the input resistance, that describes how strongly the membrane voltage deviates from its resting state after applying currents. Moreover, active properties (belonging to traces with APs) include for instance the *latency* that describes how long after minimal positive current injection one needs to wait for an AP to be elicited.

We want to establish a pipeline that not only automatically extracts the electrophysiological features from the membrane voltage traces, but does so consistently for ~ 1300 MoP neurons. We can establish such a pipeline in `Python` that is publicly available on GitHub. The code builds upon a previously established toolkit from the Allen Institute, but tailors to our Patch-seq data set. In addition to quickly and consistently extracting expert-defined features for all membrane voltage traces and each cell in a sequential fashion, the user can produce 'sanity check' figures in order to verify the correctness of feature values Figure 2.1.

We discuss consecutive steps of the pipeline next, where we apply it to experimentally observed firing patterns from a cell obtained with Patch-seq. We will see that the pipeline outputs intermediate explicit values of features describing individual action potentials (Section 2.2.1, Table 2.1), features summarizing traces of membrane voltage responses to current injections of various magnitude (Section 2.2.2, Table 2.2), and eventually features summarizing the statistics of the whole cell's electrophysiology (Section 2.2.3, Table 2.3).

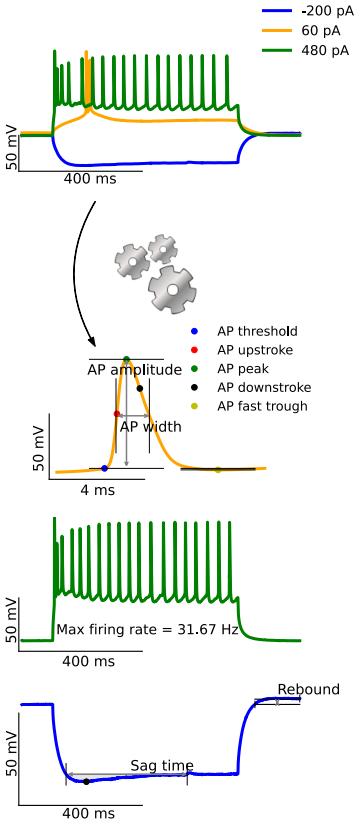

Figure 2.1: Automatic electrophysiological feature extraction. Top plot. Raw recordings of membrane voltage responses to current injections of three different magnitudes (legend). **Lower three plots.** Sanity check of automatically derived electrophysiological features. Some computed values can be directly represented by horizontal or vertical lines that can be directly visualized on top of the voltage traces.

2.2.1 Action Potential Features

A Patch-seq data set, besides transcriptomic read counts, and possibly morphological reconstructions of cells, contains raw electrophysiological recordings. More specifically, the raw data contains time points, membrane voltage values for each time point, and current magnitudes used at each experiment. They are saved in '.nwb' format and are publicly available on DANDI.

First our pipeline detects APs: when the first derivative of the membrane voltage w.r.t. to time crosses a certain threshold, called the *AP threshold*, an AP is being generated. The pipeline then continues to describe the AP's characteristics including the *AP upstroke*, which gives the maximum change in voltage (divided by time) taking place during AP generation, and the *AP amplitude*, which is the difference between the maximum voltage reached during AP generation and the voltage at AP threshold. Other features are calculated after the AP reaches its peak, including the *AP downstroke*, which denotes the most negative change of membrane

voltage w.r.t. time, and the *trough*, the lowest point the membrane voltage reaches, usually during the AP's afterhyperpolarization. Feature values for individual action potentials can be found in Table 2.1. Importantly, the pipeline can give the data analyst a 'sanity check', by plotting calculated intermediate values from which said features are derived, onto the AP itself, i.e. the membrane voltage in function of time (Figure 2.1, orange curves).

AP index	Trough time (ms)	Trough voltage (mV)	Upstroke time (ms)	Upstroke voltage (mV)	Upstroke-to-downstroke ratio	Width (ms)
0	212	-42.8	208	0.125	3.41	1.40
1	398	-55.5	220	-5.23	4.26	1.96
0	172	-42.3	167	-5.40	3.51	1.40
1	284	-49.7	178	-4.99	4.09	1.88
2	443	-52.8	372	-4.52	3.78	1.52
3	700	-53.0	600	-1.24	3.74	1.48

Table 2.1: Electrophysiological feature values for individual APs. Example of derived feature values belonging to individual APs. *AP index*: index of the AP in a sequence of APs belonging to a single membrane voltage trace. When it starts again from 0, it belongs to a sequence in the next membrane voltage trace, a response to injecting $20pA$ more. *Trough time, voltage*: time and voltage at trough (explained in main text). *Upstroke time, voltage*: time and voltage at the upstroke. *Upstroke-to-downstroke ratio*: ratio of the upstroke to the downstroke. *AP width*: width of the AP at half the height.

2.2.2 Trace Features

The pipeline continues to calculate feature values that summarize the statistics of a full trace or one experiment of current injection. When the APs have been correctly identified and described, the pipeline continues with features including the *interspike intervals* (ISIs), which measure the elapsed time between two consecutively generated APs, *spike count*, and the *AP coefficient of variation* (AP CV) which is the standard deviation of AP amplitudes in the trace divided by their mean (Table 2.2). The latter measures how much the AP amplitudes vary throughout the experiment. Most cells indeed respond with high amplitude for the first AP generated close to stimulus onset, and thereafter continue to respond with smaller AP amplitudes. Yet, some cells, including this example cell, eventually fires consistent high amplitude APs till the end of the experiment (Figure 2.1, green curves).

In Scala et al. [11], cells have not been stimulated with only positive currents, but with negative currents as well, giving rise to hyperpolarizing responses described above. During hyperpolarization, some cells rapidly reach a new steady state, others can take longer. This can be measured with the *membrane time constant*, the time constant of an exponential function fitted to the membrane voltage response during the initial phase of negative current stimulus (Figure 2.1, blue curves). Some cells additionally reach a trough, a lowest membrane voltage point, before reaching the steady state. The latter would be described by an important feature called the *sag ratio*. It is computed as the difference between the trough voltage and resting membrane potential, divided by the steady state membrane voltage difference from the resting membrane potential. Here, the steady state membrane voltage is calculated as the average voltage before stimulus offset. Some feature values can be found in Table 2.3. The data analyst can build trust in the pipeline's procedure by looking at intermediately calculated values, plotted on the membrane voltage trace itself (Figure 2.1, blue curves).

Table 2.2: Electrophysiological feature values for traces. Here we queried feature values belonging to the membrane voltage traces with highest firing rate (19 APs for this example cell). *Trace index*: index in sequence of traces with 19 APs. *ISI CV*: CV of ISIs in the trace. *AP CV*: CV of AP amplitudes. *Current*: injected current. *Burstiness*: when a cell fires rapid successive APs, it bursts. Its computation is involved and described in Scala et al. [11]. *Spike count*: number of APs.

Trace index	ISI CV	AP CV	Current (pA)	Burstiness	Spike count
0	0.25	0.13	500	0.56	19
1	0.24	0.13	520	0.50	19
3	0.33	0.15	560	0.47	19
4	0.34	0.17	620	0.54	19
5	0.42	0.21	660	0.51	19

2.2.3 Cell Features

The pipeline finalizes by computing feature values describing the electrophysiology of the cell. We have computed features describing the characteristics of individual action potentials (Section 2.2.1), as well as features summarizing the statistics of hyperpolarization and depolarization traces (Section 2.2.2). We can now think of features best describing the cell's membrane voltage response to current injections of multiple magnitudes. For electrophysiologists, especially the 1st AP fired by the cell during the experiment is of interest, and thus its characteristics like the threshold, amplitude and width is of immediate value. Other interesting features include the highest firing rate and sag of the hyperpolarizing response to the most negative current stimulus. Computed values of some of the latter features as well as other descriptive features can be found in Table 2.3.

Table 2.3: Electrophysiological feature values for a cell. *Resting membrane potential*: average voltage between experiment and stimulus onset *AP threshold*, *width*: described in main text. *AHP*: afterhyperpolarization; depth of the membrane voltage drop after the 1st AP reaches its peak, measured from AP threshold. *ISI CV*: calculated as in Table 2.2, i.e. for the trace with highest firing rate. *Sag ratio*: explained in main text.

Resting potential (mV)	AP threshold (mV)	AP width (ms)	AHP	ISI CV	Sag ratio
-71.7	-41.6	1.40	-1.12	0.25	1.11

2.3 Application

Electrophysiologists craft features such as the ones in Table 2.3 to answer specific questions in biology. For instance, do layer-4 cells in mouse somatosensory cortex respond similarly to current-injection experiments as layer-4 cells in mouse motor cortex? In Scala et al. [22], we found interesting differences in AP width, input resistance and membrane time constant between the two, as well as morphological and transcriptomic differences. Here, a methodology called sparse-reduced rank regression, which we will discuss extensively in Chapter 3, has been employed to analyze both transcriptomic and electrophysiological data.

For the Patch-seq data set introduced in Section 2.1 — serving as main data set discussed in this thesis — we wondered how well the phenotypical landscape of neurons aligns with the genotypical landscape. In Scala et al. [11], we propose in fact a 'banana tree' structure for the 'tree of cortical cell types'. In this structure, we have few large leaves

representing families of neurons, including fast-spiking *Pvalb*-expressing interneurons, *Vip*-expressing interneurons and excitatory *Pyramidal* cells, to name a few, with both distinct transcriptomic and electrophysiological characteristics. Within those leaves, however, cells rather form a continuous and correlated transcriptomic and morpho-electrical[1] landscape. Neighboring transcriptomic types largely show similar electrophysiological and morphological characteristics, and should not be thought of as purely discrete entities, as often implicitly assumed.

1: Referring here to both the morphology and electrophysiology of the neuron or cell.

We can support the 'banana tree' of cortical cell types, with a preliminary analysis, namely by running the pipeline and computing all electrophysiological features (Section 2.2.3) for each MoP neuron in the data set. We plot the computed values of some electrophysiological features across transcriptomic types in Figure 2.2. Features including the *AP count*, *AP width*, *ISI CV* and *latency* show greater variability across families than across transcriptomic types within a family (or leaf). All *Pvalb*-expressing neurons are indeed known to be fast-spiking interneurons, reflected in high *AP count*, and *Pyramidal* excitatory cells have distinct AP features from interneurons, reflected here for instance in high *AP width* and *latency*.

In Chapter 3 and Chapter 4, we corroborate this analysis by introducing statistical frameworks that can predict one modality from another or in this case, electrophysiology from transcriptome.

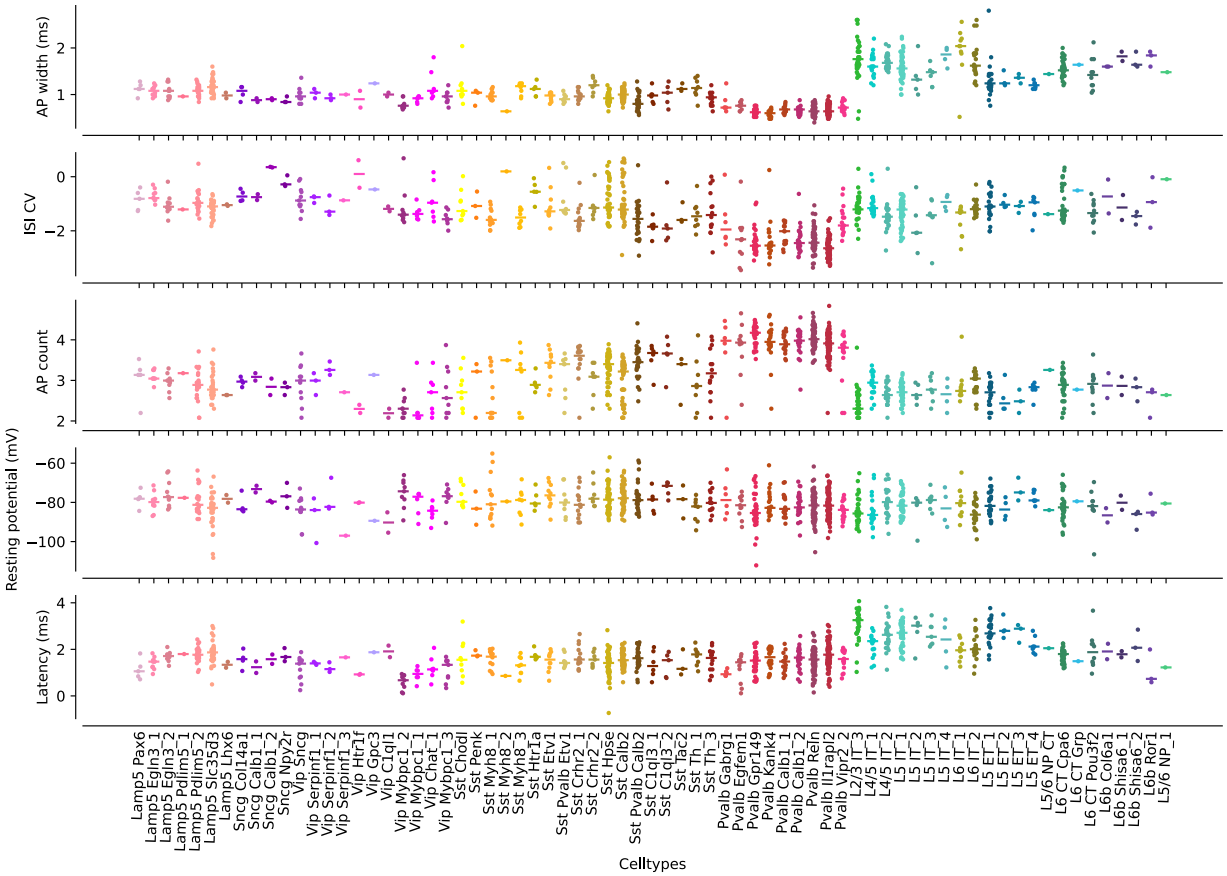

Figure 2.2: Electrophysiology across transcriptomic types. Electrophysiological feature values, computed for all MoP neurons and shown across transcriptomic types, together with the median (small horizontal lines). Colors correspond To transcriptomic types. Red: *Pvalb*; yellow: *Sst*; purple: *Vip*; salmon: *Lamp5* interneurons; green/blue: excitatory *Pyramidal* neurons. See Scala et al. [11].

2.4 Conclusion

In order to describe the firing patterns of experimentally observed neurons, faithfully, quickly and consistently, it is paramount that we build and can trust a pipeline that does so automatically, in a streaming fashion, and in the same way each time for each cell. In Section 2.2.1, Section 2.2.2 and Section 2.2.3, we discussed how our pipeline computes electrophysiological features that summarize statistics of individual action potentials, depolarizing and hyperpolarizing membrane voltage traces and the full electrophysiology of the cell, respectively. Moreover, sanity check figures such as shown in Figure 2.1 allow the user to obtain trust in the pipeline: one can visually verify that the algorithm correctly obtains values of intermediate calculations such as the trough in a hyperpolarization trace in order to calculate the sag, or the timings of AP threshold and maximum voltage to calculate the AP amplitude as the difference between the two.

[23]: Cadwell et al. (2016), 'Electrophysiological, transcriptomic and morphologic profiling of single neurons using Patch-seq'
[24]: Cadwell et al. (2017), 'Multimodal profiling of single-cell morphology, electrophysiology, and gene expression using Patch-seq'

With Patch-seq, we can not only obtain raw electrophysiological recordings, but by aspiring the cell's nucleus content, RNA contents as well [23, 24]. The latter can be sequenced, so that expression levels of tens of thousands of genes can be obtained for each cell. This allows us to deploy paired-data or 'two-view' analysis techniques to Patch-seq data sets and to go beyond mere 'one-view' perspectives. We will discuss a linear and nonlinear technique in Chapter 3 and Chapter 4, respectively.

Sparse Reduced-Rank Regression

3

We touched upon the importance of building a multimodal-conforming taxonomy of cortical cell types in Chapter 1. In order to do so, we need multimodal experimental techniques, like Patch-seq introduced in Chapter 2. Albeit a low-throughput technique, Patch-seq in particular paved the way for scientists to study the relationship between genetic identity in the form of gene counts and behavioral identities in the form of electrophysiology and morphology in the *same* set of cells (see Chapter 2).

As the nature of new emerging experimental techniques including Patch-seq becomes increasingly multimodal, so should our statistical approaches in order to extract as much knowledge possible from the joint-view data they generate. In Chapter 3, we therefore introduce *sparse reduced-rank regression*: a regression technique that can be used to predict the response variables in one modality from predictors in another, select the most important predictors for the task and produce joint-view embeddings for visualization with direct (biological) interpretability.

The discussion of sparse reduced-rank regression in Chapter 3 is largely based on Kobak et al. [21]. In Chapter 4, we will see a nonlinear extension to this linear framework, and capitalize on recent advances in machine learning to see how neural networks can do better at the prediction task, and aid the interpretability of two-view embeddings.

3.1 Sparse Reduced-rank Regression

3.1.1 Intuition

We could in principle take any response variable such as the *AP width* and see if ~ 1000 most variable detected genes could predict the whole spectrum of possible *action potential width* values. Focusing on one response variable would be termed *simple regression* (Figure 3.1 b). If besides the *action potential width* we were to also be interested in predicting e.g. the *latency* and *interspike-interval*, we perform *multivariate regression* as we try to regress our predictors onto multiple (response) variables (Figure 3.1 c). In the case of Patch-seq data, certain genes probably (negatively) correlate in their expression in order for the cell to have a certain electrophysiology or morphology. A linear combination of predictors could therefore be sufficient at the regression task. Now we move into the field of *Reduced-rank regression* (Figure 3.1 d). A final step in our reasoning process could be to exclude some genes entirely as we do not expect every gene to be relevant in determining a neuron's firing pattern. We can introduce a *group lasso* penalty to the cost function (see Section 3.1.2) so that the adapted regression model can select the most important predictors (genes). The latter is termed *sparse reduced-rank regression* (sRRR, Figure 3.1 e).

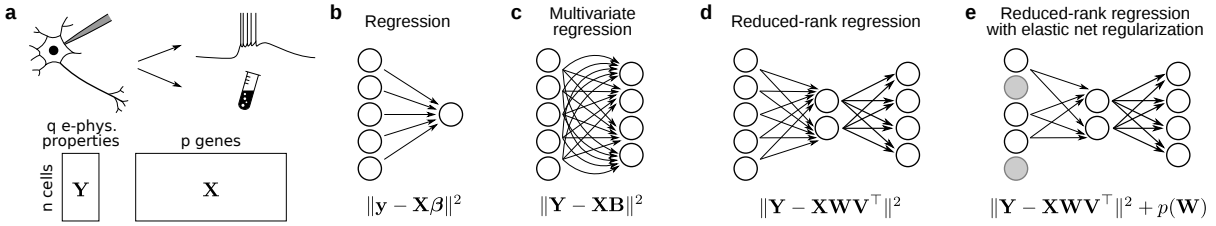

Figure 3.1: Sparse reduced-rank regression, sketch. (a) Patch-seq experiment, a schematic illustration: patch-clamp experiment leads to electrophysiological recording and is then followed by RNA extraction and sequencing. Below: data matrices after computational characterization of electrophysiological features ($\mathbf{Y}$) and estimation of gene counts ($\mathbf{X}$) (as discussed in Chapter 2). **(b)** Simple regression. **(c)** Multivariate regression. **(d)** Reduced-rank regression. **(e)** Sparse reduced-rank regression. Gray circles denote predictors that are left out of the sparse model. **(b-e)** Below, the regression task or the cost function to be minimized. Taken from Kobak et al. [21].

3.1.2 Mathematics

Patch-seq provides for two paired data matrices (Figure 3.1 a): an $n \times p$ matrix $\mathbf{X}$ containing expression levels of p genes for each of the n cells, and an $n \times q$ matrix $\mathbf{Y}$ containing q electrophysiological properties of the same n cells. We assume that both matrices are centered, i.e. column means have been subtracted. Note that we could include morphological features in Y too, or predict only morphology but we will focus on transcriptome and electrophysiology in the rest of this Chapter.

We will jump straight to discussing the mathematics behind multivariate regression [1] and build from there.

1: Simple regression can be interpreted as a special case of multivariate regression

Multivariate Regression

If we are interested in predicting all computed electrophysiological feature values in matrix $\mathbf{Y}$ based on gene count matrix $\mathbf{X}$, and want to do so linearly, we optimize the elements in a matrix $\mathbf{B}$, of size $p \times q$ so that the product $\mathbf{XB}$ is as close as possible to $\mathbf{Y}$. A popular choice for measuring 'closeness' involves the mean-squared error, so that

$$\mathcal{L} = \|\mathbf{Y} - \mathbf{XB}\|^2, \tag{3.1}$$

can be interpreted as a *cost function*, an adept name, for it is the cost or the error that we want to minimize. The smaller the cost, the smaller the mean-squared error in this case, and the better our prediction. Note that the norm used in Equation 3.1 and below is the Frobenius norm.

Its solution is well known and can be mathematically stated as

$$\mathbf{B} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{Y}. \tag{3.2}$$

Reduced-rank Regression

2: Or another gene could be similarly depressed.

In Section 3.1.1, we discussed how the expression of multiple genes could be enhanced [2] in a coherent fashion in order to produce a type of electrophysiology (or morphology) observed in neurons. With the language of mathematics, in linear regression this amounts to creating a linear combinations $\mathbf{XW}$, where $\mathbf{W}$ is of size $p \times r$, which we multiply with another matrix $\mathbf{V}^\top$ of size $r \times q$ to produce $\mathbf{XWV}^\top$, which we hope to

be as close as possible again to $\mathbf{Y}$. Again, we would use the mean-squared error loss term:

$$\mathscr{L} = \|\mathbf{Y} - \mathbf{X}\mathbf{W}\mathbf{V}^\top\|^2. \tag{3.3}$$

In reduced-rank regression [27, 28], r is called the rank, a scalar that effectively sets the dimensionality of the linear latent space, as well as the ranks of matrices $\mathbf{W}$ and $\mathbf{V}$ and could be set to as small as two for instance in order to produce direct two-dimensional latent visualizations (see Section 3.2). Furthermore $\mathbf{XW}$ could be interpreted as latent gene variability that is predictive of electrophysiological variability, and $\mathbf{YV}$ as latent electrophysiological variability that is predictive of gene variability (see Section 3.2). The solution to minimizing $\mathscr{L}$ in Equation 3.3 requires a tidy bit more mathematical detail, for which we refer to our work in Kobak et al. [21].

[27]: Izenman (1975), 'Reduced-rank regression for the multivariate linear model'
[28]: Velu et al. (2013), *Multivariate reduced-rank regression: theory and applications*

Sparse Reduced-rank Regression

In Section 3.1.1, we further postulated that we most likely only need few genes in order to understand electrophysiological variability observed in neurons. Mathematically, we can introduce what is called a *group lasso penalty* to $\mathscr{L}$ in Equation 3.3:

$$\mathscr{L} = \|\mathbf{Y} - \mathbf{X}\mathbf{W}\mathbf{V}^\top\|^2 + \lambda \sum_{i=1}^{p} \|\mathbf{W}_{i\cdot}\|_2. \tag{3.4}$$

Here, λ is again a scalar we can set to our choosing and determines the penalty strength, whereas $\sum_{i=1}^{p} \|\mathbf{W}_{i\cdot}\|_2$ formulates a sum over all $\mathbf{W}$ row ℓ_2 norms. To reduce the latter term in the cost amounts to making certain ℓ_2 norms in the sum insignificant or close to zero, which is exactly what a lasso penalty is known for. This is equivalent to neglecting certain genes in the prediction task. The cost or loss function $\mathscr{L}$ in Equation 3.4 therefore establishes a tradeoff: the second term tries to significantly reduce every $\mathbf{W}$ row's *weight* or contribution to the prediction task, measured by the ℓ_2 norm, to zero, yet some row norms will prevail, as at least some rows (corresponding to some genes) are needed to perform well at the prediction task at hand (first term in Equation 3.4).

It is important to note that the group lasso penalty is a form of *regularization* that is necessary in regression problems whenever the amount of features in the model denoted by p and q exceeds the amount of data n to our disposal. The mouse genome comes with 20 thousand genes, but usually Patch-seq data sets are reduced to $p \sim 1000$ in $\mathbf{X}$, i.e. ~ 1000 most variable genes are selected for downstream analyses. Patch-seq data sets with sufficiently small n (as most are) therefore need to be regularized.

Besides lasso, employing a *ridge penalty* is another common choice in overparameterized linear regression problems. *Ridge* regularization can be introduced to reduce every parameter's *weight* (every element in matrices $\mathbf{W}$) or contribution to the prediction task, in addition to lasso. In fact, within the penalty term, we can introduce a hyperparameter α, a scalar up to our choosing and set between $[0, 1]$, that determines the relative strength of lasso to ridge:

$$\mathcal{L} = \frac{1}{2n}\|\mathbf{Y} - \mathbf{X}\mathbf{W}\mathbf{V}^\top\|^2 + \lambda\Big(\alpha \sum_{i=1}^{p}\|\mathbf{W}_{i\cdot}\|_2 + (1-\alpha)\|\mathbf{W}\|^2/2\Big), \tag{3.5}$$

which is also termed *reduced-rank regression with elastic net regularization*. We will usually refer to sparse reduced-rank regression however, as we deploy the ℓ_1 group lasso penalty, to *select* genes in order to predict electrophysiological variability [21, 29]. The minimization problem in Equation 3.5 is a biconvex problem: when $\mathbf{W}$ is fixed, we can find the optimal $\mathbf{V}$ and vice versa. Usually Equation 3.5 is also subject to $\mathbf{V}^\top\mathbf{V} = \mathbf{I}$, i.e. $\mathbf{V}$ has fixed ℓ_2 norm.

[21]: Kobak et al. (2021), 'Sparse reduced-rank regression for exploratory visualization of paired multivariate data'
[29]: Chen et al. (2012), 'Sparse reduced-rank regression for simultaneous dimension reduction and variable selection'

Solutions to the biconvex problem involve again a bit more mathematical detail which can be found in Kobak et al. [21]. Following Friedman, Hastie, and Tibshirani [30], we can use the `glmnet` library to find optimal solutions for the minimization problem in Equation 3.5.

3.1.3 Training

$\mathbf{R^2}$ Coefficient of Determination

In regression we can use the R^2 coefficient of determination to see how well the regression predicts the data:

$$R^2 = 1 - \frac{\|\mathbf{Y} - \mathbf{X}\mathbf{W}\mathbf{V}^\top\|^2}{\|\mathbf{Y}\|^2}. \tag{3.6}$$

The higher R^2, the better our prediction works. In fact, a perfect prediction for all data points would mean $\|\mathbf{Y} - \mathbf{X}\mathbf{W}\mathbf{V}^\top\|^2 = 0$, so that $R^2 = 1$. Otherwise R^2 will be lower than 1.

If we were to evaluate the R^2 score on the same data with which we minimized $\mathcal{L}$ in Equation 3.5, called *training* data in machine learning jargon, we however risk *overfitting*. Overfitting occurs when the model performs well on training data but poorly on data the minimization procedure did not see, termed *test* data. Put differently, your model fails to generalize to data points collected after you have already trained your model. One way of preventing overfitting is to evaluate on a *test* set rather than the *training* set with:

$$R^2_{test} = 1 - \frac{\|\mathbf{Y}_{\text{test}} - \mathbf{X}_{\text{test}}\mathbf{W}^*\mathbf{V}^{*\top}\|^2}{\|\mathbf{Y}_{\text{test}}\|^2}, \tag{3.7}$$

that is, we have minimized $\mathcal{L}$ with respect to $\mathbf{W}$ and $\mathbf{V}$ in Equation 3.5 with training data $\mathbf{X}_{train}$ and $\mathbf{Y}_{train}$ and obtained parameters $\mathbf{W}^*$ and $\mathbf{V}^*$. We then evaluate R^2 on $\mathbf{X}_{test}$ and $\mathbf{Y}_{test}$. On which data points we train and on which we test depends on the statistical modeler's choice. The entire data set can for instance be split in 80% training data and 20% test data.

Cross-validation

When we set hyperparameters α, λ and r, we can find the optimal minimum for $\mathcal{L}$ in Equation 3.5. But how do we choose those parameters? What are the best ones? We can perform what is called a grid search, that is we try out all combinations of α, λ and r and see which one performs best according to a metric such as R^2 introduced in Equation 3.6. Moreover, it is good practice to find the best combination before one evaluates on the test set introduced in Equation 3.7. An interesting strategy is to split the training data set in 10 folds or parts, where 9 folds make up the (new) training data set, and 1 fold an *evaluation* set, so that after training, we can evaluate:

$$R^2_{val} = 1 - \frac{\|\mathbf{Y}_{\text{val}} - \mathbf{X}_{\text{val}}\mathbf{W}^*\mathbf{V}^{*\top}\|^2}{\|\mathbf{Y}_{\text{val}}\|^2}. \tag{3.8}$$

Here again, we obtained optimal parameters $\mathbf{W}^*$ and $\mathbf{V}^*$ for $\mathcal{L}$ on the training data set $\mathbf{X}_{train}$ and $\mathbf{Y}_{train}$, but evaluate R^2 on $\mathbf{X}_{val}$ and $\mathbf{Y}_{val}$.

In *cross-validation*, every fold has served as both training and validation data, so that for each combination of α, λ and r we can compute the average of R^2_{val} over 10 validation folds. This will give us an optimal combination (α, λ, r) corresponding to highest average R^2_{val}, with which we can train the entire regression model again, this time on all 10 folds and consequently compute R^2 on the true test data set, as formulated in Equation 3.7. This also leads to a final combination of optimal $\mathbf{W}^*$ and $\mathbf{V}^*$ matrices.

Relaxed Elastic Net

Elastic net or even the lasso penalty on its own can lead to non-zero weights (elements in matrices $\mathbf{W}$ or $\mathbf{V}$) shrinking too much [31] during optimization. There have been several suggestions in the literature on how to mitigate this effect [31–33], including setting a combination of penalties in α and λ in a first optimization round, followed by a combination of (softer) penalties in α and λ in a second optimization round. Similarly, we find that one can see a strong improvement in predictive performance if, after a first round, we take the predictors or genes with non-zero ℓ_2 row norms in $\mathbf{W}$ and run the optimization again using $\alpha = 0$ (i.e. pure ridge) with the same value of λ as in the first round.

[31]: Zou et al. (2005), 'Regularization and variable selection via the elastic net'

[31]: Zou et al. (2005), 'Regularization and variable selection via the elastic net'
[32]: Efron et al. (2004), 'Least angle regression'
[33]: Meinshausen (2007), 'Relaxed lasso'

Comparison to Canonical Correlation Analysis

We have seen throughout Section 3.1.2 that *sparse reduced-rank regression* embeds the transcriptomic space $\mathbf{X}$ in a lower-dimensional linear space $\mathbf{XW}$ from which it further linearly projects $\mathbf{XWV}^\top$ in order to predict electrophysiology $\mathbf{XWV}^\top \sim \mathbf{Y}$. Its aim is therefore one of prediction, mirrored in the minimization of $\mathcal{L}$ in Equation 3.5. Yet, $\mathbf{XW}$ and $\mathbf{YV}$ allow for direct linear lower-dimensional embeddings in both modalities that are biologically interpretable (see Section 3.2).

Instead of optimizing a prediction task, one could seek for maximal correlation in the two linear embeddings $\mathbf{XW}$ and $\mathbf{YV}$. This methodology is termed *canonical correlation analysis* (CCA), of which sparse variants also

exist, arguably the most popular being sparse CCA developed by Witten, Tibshirani, and Hastie [34]. Interestingly, however, one can compare correlations between $\mathbf{XW}^*$ and $\mathbf{YV}^*$ (that is, $\mathbf{W}^*$ and $\mathbf{V}^*$ obtained with sRRR) with correlations between $\mathbf{XW}_{sCCA}$ and $\mathbf{YV}_{sCCA}$ and find that sRRR can outperform sparse CCA, even though its aim is not to maximize correlations. As both sRRR and sparse CCA involve regularization, the explanation could be due to 'over-regularization' using sparse CCA. Details and possible explanations can be found in Kobak et al. [21].

Conclusion

Sparse reduced-rank regression is an intuitive linear framework, employed by data scientists to predict variability in one modality based on another. It uses the group lasso penalty to select predictors and linearly combines them into a latent lower-dimensional representation to predict variability in the response variable. It therefore lends itself beautifully to Patch-seq: it selects the most important genes that, linearly combined, predict electrophysiological variability.

3.2 Latent Visualizations

In Section 3.1.2 we discussed the significance of the rank r in sparse reduced-*rank* regression: a scalar that sets the dimensionality of the lower-dimensional latent space. We saw that it could be set to $r = 2$ so that $\mathbf{XW}$ constructs a two-dimensional latent space with gene variability that is predictive of electrophysiological variability, and $\mathbf{YV}$ constructs a two-dimensional latent space with electrophysiological variability that is predictive of gene variability. Both two-dimensional representations can be visualized next to one another creating what we term a *bibiplot* [21] (Figure 3.2).

[21]: Kobak et al. (2021), 'Sparse reduced-rank regression for exploratory visualization of paired multivariate data'

Data Sets

Here, we apply sRRR to four Patch-seq data sets in order to demonstrate the intuitive biological interpretability that comes with bibiplots in Figure 3.2. The fist data set involves $n = 51$ interneurons from *layer 1* in the mouse neocortex taken from Fuzik et al. [25] and visualized in Figure 3.2 a. The second data set involves 110 cells from layer 4 in both the mouse visual cortex and somatosensory cortex discussed by Scala et al. [22] and displayed in Figure 3.2 b. The third data set is discussed throughout this entire thesis, $n = 1213$ excitatory and inhibitory neurons obtained with Patch-seq by Scala et al. [11] that span different layers and cover various cell types in the mouse motor cortex (Figure 3.2 c). The fourth and last data set covers a subset of the data published by Gouwens et al. [35]: it covers many interneuronal cell types in the mouse visual cortex (Figure 3.2 d). For preprocessing details regarding all data sets, we refer again to Kobak et al. [21].

On all data sets, we apply sRRR with $r = 2$ to obtain two-dimensional visualizations and set λ to yield only 20 biologically prominent genes in each data set (α is variable with values: 0.5, 0.5, 1.0, and 1.0 respectively).

Two-dimensional visualizations are shown and discussed in Section 3.2, some R^2_{val} values in Chapter 4.

Bibiplots

To construct bibiplots for all data sets in the transcriptomic space, we first use the bottleneck representation $\mathbf{XW}^*$ for a first scatter plot (Figure 3.2 a-d, left) and analogously $\mathbf{YV}^*$ to create the electrophysiological latent space in a second one (Figure 3.2 a-d, right). We can overlay both scatterplots with lines showing correlations of selected individual genes or electrophysiological features with the first and second latent component of the trained sRRR model (first and second column in $\mathbf{XW}^*$ for scatterplots on the left and first and second column in $\mathbf{YV}^*$ for scatterplots on the right, Figure 3.2 grey lines). More specifically, if we take predictor or gene $\mathbf{g}$, the line extends from the origin $(0, 0)$ to the point with coordinates $(\text{corr}\,(\mathbf{g}, (\mathbf{XW}^*)_{1\cdot})\,, \text{corr}\,(\mathbf{g}, (\mathbf{XW}^*)_{\cdot 2}))$, where corr refers to the Pearson correlation between two variables. The circle (*correlation circle*) shows maximal possible correlations. The relative scaling of the scatter plot and the lines/circle is arbitrary.

The bibiplots therefore condense information intuitively in two two-dimensional latent representations side by side. We can directly visualize and compare two latent embeddings, each from a different modality, where one predicts the other. If the regression score defined by R^2 of the sRRR model is high, then the two scatter plots will be similar to each other. Moreover, we can compare the directions of variables between the two plots visualized by the overlayed lines, and hypothesize which electrophysiological features are associated with which genes.

Analysis

The first data set encompasses two types of interneurons from layer 1 of mouse cortex: neurogliaform cells and single bouqet cells (shown in orange and green in Figure 3.2 a respectively). We can see that the first latent dimension or first sRRR component captured the difference between the two cell types (Figure 3.2a). The second sRRR component has only one gene strongly associated with it (Figure 3.2a) and contributes only a very small increase in R^2_{val}, as one can see after comparing the cross-validated values for $r = 1$ and $r = 2$ at 20 selected genes (see [21]).

[21]: Kobak et al. (2021), 'Sparse reduced-rank regression for exploratory visualization of paired multivariate data'

Similarly, the first latent dimension captures the difference between cells located in the visual and somatosensory cortex for the second data set (orange and red colors in Figure 3.2 b). The selected genes here are pointing in all directions, and indeed the second component contributes a substantial increase in R^2 [21]. This suggests that both components are biologically meaningful. See the work of Scala et al. [22] for a more in-depth analysis.

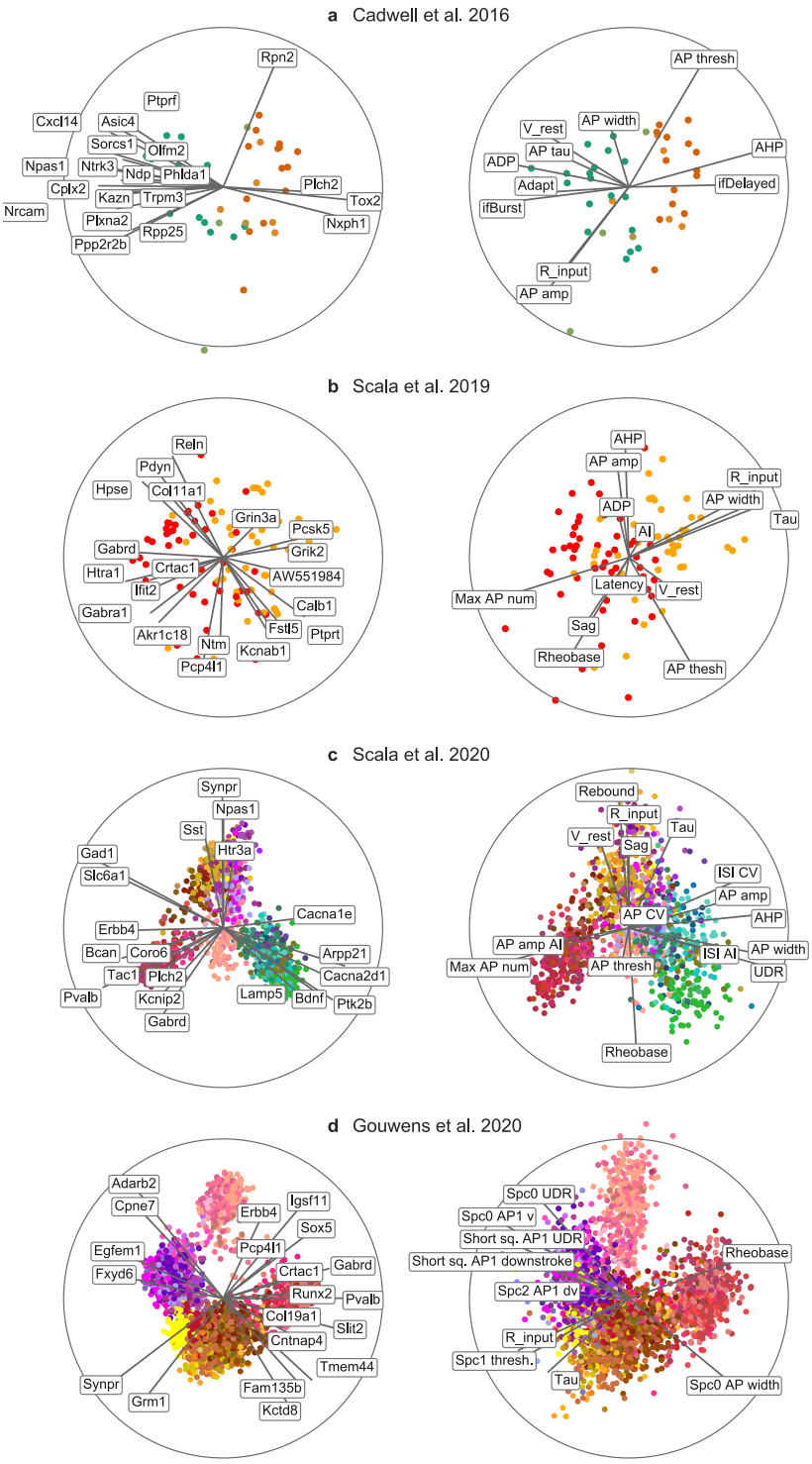

Figure 3.2: Sparse reduced-rank regression applied to Patch-seq. (a) Transcriptomic two-dimensional space (left) and electrophysiological two-dimensional space (right) — also termed 'bibiplots' — for the first data set from Fuzik et al. [25]. Color codes for cell type (orange: neurogliaform cells; green: single bouqet cells). Only 20 genes selected by the model are shown on the left. **(b)** Analogous to a, but applied to the second data set from Scala et al. [22]. Color denotes cortical area (orange: visual cortex; red: somatosensory cortex). **(c)** Third data set from Scala et al. [11]. Color denotes for transcriptomic type. Red: *Pvalb*; yellow: *Sst*; purple: *Vip*; salmon: *Lamp5* interneurons; green/blue: excitatory *Pyramidal* neurons. **(d)** Fourth data set from Gouwens et al. [35]. Color denotes same transcriptomic type as in c. For this data set, only a subset of electrophysiological features are shown on the right to reduce the clutter.

Our third data set is much larger than the previous two and includes a much more diverse selection of neuron types. As a result, the R^2 values

are substantially higher and the model needs to use rank $r \geq 5$ to reach its optimal performance. Here we nevertheless use $r = 2$ because it allows the same kind of visualization as for the other data sets (Figure 3.2 c). See the work of Scala et al. [11] for a more in-depth analysis using sRRR with rank $r = 5$. Two-dimensional bibiplots separate major classes of neurons, such as *Pvalb*, *Sst*, *Vip*, and *Lamp5* expressing interneurons (red/orange/purple/salmon), and excitatory cells (green). Moreover, some selected genes are directly related to ion channel dynamics, such as the calcium channel subunit genes *Cacna1e* and *Cacna2d1* or the potassium channel-interacting protein gene *Kcnip2*. The same is true in the V1 data set (Figure 3.2d), where e.g. a potassium channel gene *Kctd8* is among those selected by the model.

It is worth noting that the $\mathbf{YV}^*$ representation in the electrophysiological space for our third data set (Figure 3.2 c, right) is very similar to the PCA lower-dimensional representation in Kobak et al. [21] (Figure 2b). This indicates that our sRRR model explains the dominant modes of variation among the dependent variables. We observe the same in other datasets analyzed here, even though in principle it is possible that latent components derived with PCA of the $\mathbf{Y}$ matrix are different from the ones derived with our sRRR model.

The reader is referred to GitHub for a `Python` implementation of sRRR models applied to Patch-seq, and to GitHub for its application in the study of Scala et al. [11].

3.3 Conclusion

Sparse reduced-rank regression, a linear statistical framework, selects the most meaningful predictors to predict variability in a response variable and has the additional bonus of producing lower-dimensional embeddings that are intuitive and directly interpretable, especially in biology. For paired Patch-seq data sets, it can select genes as part of the minimization procedure in Equation 3.5, that contribute most to predicting electrophysiological variability. So-called bibiplots can help hypothesize which genes are expressed to explain the variability in individual electrophysiological features.

In Chapter 5, we will be introduced to the Hodgkin-Huxley model, a biophysical model that includes deterministic mechanisms. Whereas in statistical tools such as sRRR or sparse CCA we optimize (linear) parameters $\mathbf{W}$ and $\mathbf{V}$ to relate one space to another directly, deterministic mechanisms explain carefully, at every time step in the *model*, *how* we move from one state to the next.

In Chapter 8 we will move even one step further and combine both sRRR and biophysical models as a first attempt to make the partly *causal* bridge from an experimental viewpoint observed in the genetic makeup of neurons to a simulation viewpoint explicit in the model parameters of a biophysical model, that in itself models the electrophysiological makeup of the same neurons.

Sparse Bottleneck Neural Networks

4

In Chapter 2 we introduced Patch-seq and its importance for the multimodal study of neurons in the nervous system. We also showed how a pipeline implemented in `Python` can be utilized to extract expert-defined electrophysiological features automatically and its that their computed values can be verified with 'sanity check plots'.

Chapter 3 then introduced sparse reduced-rank regression, a biologically intuitive and linear statistical tool that embraces the multimodal nature observed in neuroscience. It selects most predictive genes, linearly combines them into latent components that subsequently predict electrophysiological response variables. We can furthermore conveniently use the latent components to produce low-dimensional joint-view embeddings, and, much like PCA, we can verify which genes and electrophysiological features the sRRR components correlate highly with, and overlay that information on the same embedding.

The next natural question to ask is whether we can do better, by going beyond a linear framework. It is not unreasonable to assume that nonlinear computations used in the regression task could be helpful to uncover the highly nonlinear and complex steps the biological cell undertakes to eventually produce its firing patterns. The cell transcribes certain genes, translates them into functional proteins, specifically ion channels in the cell membrane. Those channels open under certain conditions, for instance when the membrane voltage reaches a certain threshold, or when a neurotransmitter (a chemical) finds access to its binding site. In Chapter 5, we will see that, when certain ion channels are in the open state, the membrane voltage undergoes rapid and highly nonlinear fluctuations, leading to so-called action potentials. So especially for this stage in the cell, we expect nonlinear computational units to be useful in relating gene expression levels to firing patterns.

In Chapter 4, we therefore introduce sparse bottleneck neural networks (sBNNs), a machine learning framework for the nonlinear prediction of electrophysiological features based on gene expression levels. We will see that it outperforms sRRR on the prediction task, and that two-dimensional bottlenecks can provide insightful visualizations not immediately captured by sRRR unless one introduces more than two dimensions in the latent space.

The reader is invited to read **Bernaerts**, Berens, and Kobak [19] for an in-depth exploration, of which the main points will be outlined next.

4.1 Sparse Bottleneck Neural Networks

As discussed in Chapter 3, sRRR uses a linear combination of few predictors or genes, selected by introducing a group lasso penalty in the cost function (see Section 3.1.2 and Equation 3.5) to predict electrophysiological variability. Moreover, the linear latent embedding proves very useful

to visualize and interpret this two-view data in biology and neuroscience. We would like to therefore build a deep neural network that shares many of the intuitive features of sRRR. It should be able to be pruned at its input layer to select few genes and a bottleneck should allow for direct joint-view nonlinear embeddings.

4.1.1 Architecture

In our architecture (Figure 4.1c), the sBNN *encoder* part compresses the $p = 1000$ high-dimensional transcriptomic data into a two-dimensional bottleneck representation via several fully connected layers, from which one head of the network reconstructs the transcriptomic data (termed *autoencoder*), whereas an additional head predicts the electrophysiological properties of the neurons (termed *alloencoder*). The encoder and decoder have two hidden nonlinear layers each (we use 512 and 128 units for the encoder, and 128 and 512 units for the decoder). We use exponential linear units for the hidden layers, but linear units for the bottleneck layer and the output layer because the response variables can take values in $\mathbb{R}$ and are not necessarily constrained to be positive or to lie between 0 and 1.

To enforce sparsity, we can impose the same group lasso ℓ_1 penalty as introduced in Section 3.1.2 and Equation 3.5 to the first encoding layer weight. We moreover use a constant ℓ_2 penalty of 10^{-10}.

Importantly, linear models with lasso penalty allow for solutions that converge to having exactly zero entries [30]. This is not the case for deep learning models, where gradient descent will generally fluctuate around solutions with many small but non-zero elements. Our strategy to achieve a genuinely sparse model here is to (i) impose a strong lasso penalty and train the network until convergence; then (ii) prune the input layer by only selecting a pre-specified number of input units with the highest ℓ_2 row norms in the first layer, and deleting all the other input units; and finally, (iii) fine-tune the resulting model with lasso penalty set to zero. This procedure mimics the 'relaxed' elastic net procedure described in Section 3.1.3: the lasso-regularized model is pruned and then fine-tuned without a lasso penalty.

[30]: Friedman et al. (2010), 'Regularization paths for generalized linear models via coordinate descent'

To produce outputs approaching electrophysiological measurements we can use the same MSE loss as introduced for the first time in Equation 3.1.

For the entire implementation in `Keras` [36] (version 2.7.0) and TensorFlow [37] (version 2.7.0) libraries in `Python`, the reader is referred to GitHub.

[36]: Chollet et al. (2015), *Keras*

[37]: Martín Abadi et al. (2015), *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*

4.1.2 Training

Pre-training

Transferring the weights from an initial task A to a target task B with subsequent fine-tuning can improve the performance on task B compared to random weight initialization [38]. As such it can be demonstrated that for Patch-seq, directly training the network on the regression task, i.e. predicting electrophysiological measurements from gene expression levels,

[38]: Yosinski et al. (2014), 'How transferable are features in deep neural networks?'

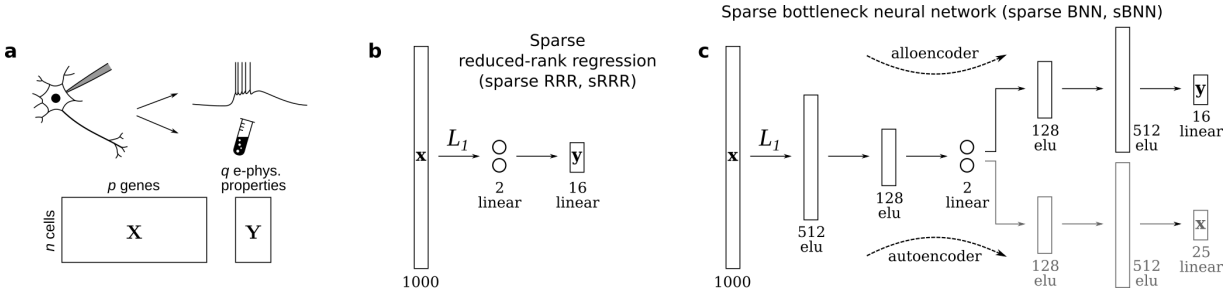

Figure 4.1: Patch-seq, sRRR and sBNN schemas. (a) Patch-seq allows for electrophysiological recordings, collected here in matrix **Y** and extraction of RNA content of the same cells, collected in matrix **X**. **(b)** SRRR schema, here again for reference. Also see Figure 3.1. **(c)** SBNN schema. Encoder takes $p = 1000$ genes in input layer and produces two-dimensional latent space in its (linear) output. From latent components, the *alloencoder* predicts 16 electrophysiological features, and the *autoencoder* predicts 25 genes selected after pruning with a group lasso ℓ_1 penalty in the first layer weight.

results in suboptimal performance (Figure 4.2), both on the validation and on the training sets. Better results can be achieved after pre-training the network on a classification task. This approach is inspired by the usual practice for image-based *regression* tasks to start with a convolutional neural network with weights trained for *classification* on `ImageNet` and to fine-tune them on the task at hand [39].

[39]: Lathuilière et al. (2019), 'A comprehensive analysis of deep regression'

To perform classification, one can perform k-means clustering to partition the whole data set of n points into K[1] clusters based on the values of the response variables. Thereafter, the output layer of the network can be replaced with a K-element *softmax*[2] so that the network can be used to predict cluster identity with the cross-entropy loss:

$$\mathscr{L}_{CE} = -\sum_{k=1}^{K} p(s_k) \log\left(q(s_k)\right). \tag{4.1}$$

Here, the cross-entropy loss is stated for one training data example s. 'True' class probabilities $p(s_k)$ are derived with k-means. The network, however, also outputs *softmax* probabilities $q(s_k)$, which we want to get as close as possible to $p(s_k)$. The final loss that the network minimizes sums Equation 4.1 up for all training data samples s.

1: For Patch-seq experiments here, $K = 20$.

2: When an output layer of a deep neural network outputs values in $\mathbb{R}$, the layer can be transformed to a density, that is, all output nodes summing up to 1, by applying the *softmax* operation. Given the value at output layer node o_k, applying the *softmax* means transforming o_k with the operation $\frac{\exp(o_k)}{\sum_j \exp\left(o_j\right)}$, where the sum is over all output layer nodes indexed by j.

We hold out 40% of the training set as the validation set (Section 3.1.3) in order to perform early stopping and choose the training epoch that has the lowest unpenalized cross-entropy validation loss (during 50 epochs). We can then use the weights obtained at that epoch as a starting point for subsequent training with the original MSE loss.

Freezing

[40]: Howard et al. (2018), 'Universal Language Model Fine-tuning for Text Classification'

A common advice in transfer learning is to hold the bottom layers frozen after the transfer (as those are the most generalizable [38]), train the upper layers first and then unfreeze all layers for fine-tuning [40]. Here, we hold the bottom two layers frozen for the first 50 fine-tuning epochs, and then train the network with all layers unfrozen for another 50 epochs (Figure 4.2, vertical dashed lines).

Pruning

[41]: Han et al. (2015), 'Learning both Weights and Connections for Efficient Neural Networks'
[42]: Blalock et al. (2020), 'What is the State of Neural Network Pruning?'

This can then be followed by pruning the input layer [41, 42] at epoch 100 (Figure 4.2). When we prune, we keep 25 input layer nodes i with highest first layer weight row norms or high $\|\mathbf{W}^{(1)}_{\mathbf{i},\cdot}\|^2$. Remember that we are introducing a *group* lasso penalty, enforcing some rows in the first layer matrix $\mathbf{W}^{(1)}$ to have such a reduced *weight* in the feed-forward direction of the neural network, that corresponding input nodes or genes attribute close to nothing in the prediction task. Pruning in this sense, leads to gene selection.

Optimization

[43]: Kingma et al. (2015), 'Adam: A Method for Stochastic Optimization'

We can use the standard *Adam* optimizer [43] but with learning rate set to 0.0001. The default value 0.001 sometimes can lead to very noisy convergence characteristics, especially for small data set sizes like ours. The learning rate is then further reduced to 0.00005 after *unfreezing*, encouraging an optimal local solution not too far from the current state. It takes ~1 minute to train any of these models for the entire 250 epochs on NVIDIA Titan Xp.

4.2 Sparse Bottleneck Neural Networks for Patch-seq

3: For Patch-seq standards.

Equipped with our architecture design and training schedule, we are ready to deploy our deep neural network on two large[3] Patch-seq data sets, which we will refer to as M1 (mouse *m*otor cortex) and V1 (mouse *v*isual cortex). They have already been introduced here in Section 3.2, but we briefly reiterate them here for convenience.

4.2.1 Patch-seq Data Sets

The M1 data set refers to $n = 1213$ excitatory and inhibitory neurons obtained with Patch-seq by Scala et al. [11] that span different layers and cover various cell types in the mouse motor cortex (M1, see Figure 3.2c). We focus on $q = 16$ expert-defined electrophysiological features that are 'well-behaved', i.e. did not correlate too highly with one another and were well represented by a Gaussian distribution, and $p = 1000$ most variable genes, leading to $\mathbf{X}$ and $\mathbf{Y}$ matrices of size 1213×1000 and 1213×16 respectively. It should be assumed for the following discussion that all columns in $\mathbf{X}$ and $\mathbf{Y}$ have zero mean and unit variance, i.e. were standardized before training the network. The M1 data set can be found on Github.

The V1 Patch-seq data set contains $n = 3395$ inhibitory neurons from primary visual cortex (V1) of adult mice Gouwens et al. [44], with $p = 1252$ and $q = 55$. It covers many interneuronal cell types in the mouse visual cortex derived with Patch-seq (Figure 3.2 d). Analogous to the M1 data set, columns in both $\mathbf{X}$ and $\mathbf{Y}$ are standardized. The V1 data set can also be found on Github.

It is good to note that we are equipped with only low $n \sim 1000s$ data set sizes according to machine learning standards, yet have a lot of predictors (genes) $p = 1000$, some layers with $n \sim 100s$ of nodes, and eventually few response variables $q = 16$ very informative of electrophysiology in neurons. That is a lot of variables and little data to tune them. Nevertheless, we will see in Section 4.2.2 that with sufficient regularization, deep neural networks can be utilized to perform better on the prediction task on held-out data during cross-validation than linear frameworks such as sparse reduced-rank regression, even though they do not come with as much parameters.

4.2.2 Predictive Performance

The cross-validated R^2_{val} of the $r = 2$ sRRR model selecting 25 genes is 0.35 ± 0.02 and 0.19 ± 0.01 respectively for the M1 and V1 data set (mean±SD across cross-validation folds, see Table 4.1 and Section 3.1.3). Without a two-dimensional linear bottleneck, the full-rank sRRR model R^2_{val} increases to 0.40. Our nonlinear sBNN model with *two-dimensional* bottleneck reaches 0.40 for M1 (Figure 4.2, full green line) and 0.25 for the V1 data set. SBNN therefore outperforms $r = 2$ sRRR models and performs as well as a full-rank sRRR linear models. Note that the full rank here is 16.

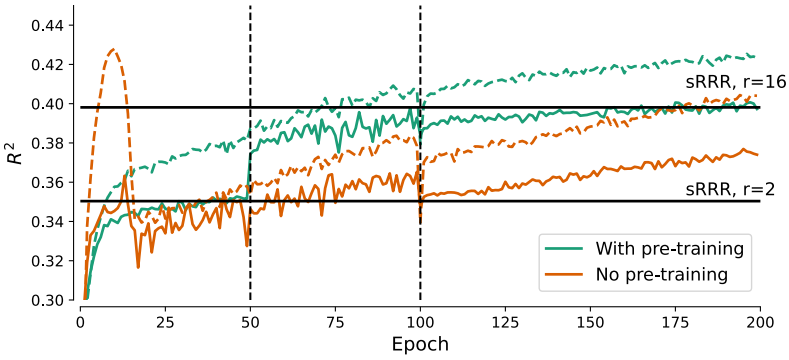

Figure 4.2: SBNN training and validation set performance during training of M1 Patch-seq data set. Training set (dashed lines) and validation set (full lines) R^2 scores of the sBNN with and without pre-training. The bottom two layers are frozen for 50 epochs after pre-training. All models were pruned to 25 input units at epoch 100 and trained for further 100 epochs. Horizontal lines show performances of $r = 2$ and $r = 16$ sRRR models with 25 genes.

Model	M1	V1	CITE-Seq
SRRR, rank-2	.35 ± .02	.19 ± .01	.23 ± .04
SRRR, full-rank	.40 ± .02	.25 ± .01	.35 ± .06
SBNN	.40 ± .02	.26 ± .01	.38 ± .07

Table 4.1: Cross-validated Multivariate R^2 Performance Scores for the SRRR and SBNN Frameworks Regarding the M1, V1, and CITE-Seq Data Sets. Mean±SD across 10 cross-validation folds.

R^2_{val} can vary across individual electrophysiological features (Figure 4.3). For the M1 data set, the highest cross-validated R^2_{val} values are obtained for the upstroke-downstroke ratio of the action potential (R^2_{val} = 0.77 ± 0.03), maximum action potential count (R^2_{val} = 0.73 ± 0.04), and action potential width (R^2_{val} = 0.72 ± 0.05), whereas some other features have R^2_{val} below 0.1.

4.2.3 Gene Selection

The group lasso penalty on the first layer weight selects genes relevant for electrophysiology and predictive of different transcriptomic types.

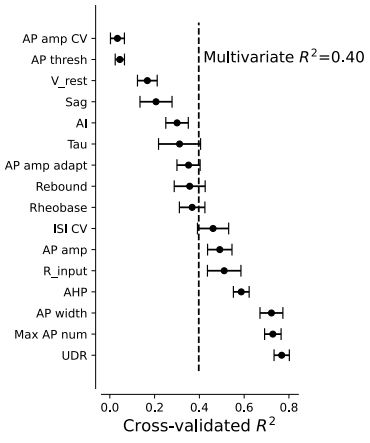

Figure 4.3: Sparse bottleneck neural network performance on individual electrophysiological features. Cross-validated R^2_{val} scores calculated for each individual electrophysiological feature (mean ± s.d. across 10 cross-validation folds).

In the sBNN model, *Sst*, *Vip*, *Pvalb*, *Gad1*, *Coro6*, *Ndst3*, etc. are selected (see Table 4.2 for the M1 data set, where we sort genes in descending order by the ℓ_2 norms of the first layer weights). *Sst*, *Vip*, *Pvalb* and *Lamp5* are well-known marker genes of specific neuronal populations [5, 8], and *Gad1* encodes an enzyme responsible for catalyzing the production of gamma-aminobutyric acid (GABA), a crucial molecule in neuronal signaling. For the V1 data set, the list starts with *Vip, Synpr, Pdyn, Fxyd6, Trhde, Cacna2d3*, etc. Here, *Fxyd6* and *Cacna2d3* encode important ion transport transmembrane proteins, directly affecting the electrophysiology of cells.

In both Patch-seq data sets, selected genes beyond the known cell class markers often have direct and interpretable relations to ion channel dynamics and the electrophysiological properties. For example, *Kcnc2*, selected by the sBNN for the V1 data set, encodes for the Kv3.1 potassium voltage-gated channel directly affecting the firing rate most notably in fast-spiking *Pvalb* interneurons [45–48]. Similarly *Kcnab1*, selected in the M1 data set, also encodes for a potassium voltage-gated channel, which is expressed higher in fast-spiking *Pvalb* interneurons (Figure 4.5 a).

The strength of the lasso penalty seems to play an important role (Figure 4.4). For low penalties such as 0.0001, the ℓ_2 weight norms for different input units before pruning are all of similar size (Figure 4.4 b), leading to arbitrary genes being selected after pruning and bad performance afterwards (Figure 4.4 a). Strong penalties (0.01–1.0) lead to rapidly decaying weight norms (Figure 4.4 b), similar sets of genes being selected, and similar final performance (Figure 4.4 a), with 0.1 penalty achieving the highest performance.

A curious effect arises when we impose the group lasso penalty: while the training and test performance are initially both improving, after several epochs they both deteriorate rapidly when we have not already pre-trained the network (Figure 4.2, orange curves). It can also be observed, during pre-training (training curves not shown here), as we would apply the group lasso penalty already then. This is not due to overfitting because the training performance becomes worse as well. Instead, the shape of the performance curves suggests that the lasso penalty 'begins kicking in', bringing the penalized MSE loss down while the unpenalized MSE loss goes up. After this initial rapid decrease, the validation test set performance slowly improves again with further training. It is speculated that this behavior is reminiscent of a phase transition in the gradient descent dynamics: in the earlier ~10 epochs mainly the MSE (or cross-entropy) term is being optimized, while in the next ~10 epochs it is mainly the lasso penalty[19].

[5]: Tremblay et al. (2016), 'GABAergic interneurons in the neocortex: from cellular properties to circuits'
[8]: Tasic et al. (2018), 'Shared and distinct transcriptomic cell types across neocortical areas'
[45]: Muona et al. (2015), 'A recurrent de novo mutation in KCNC1 causes progressive myoclonus epilepsy'
[46]: Wymore et al. (1994), 'Genomic Organization, Nucleotide Sequence, Biophysical Properties, and Localization of the Voltage-Gated K+ Channel Gene KCNA4/Kv1.4 to Mouse Chromosome 2/Human 11p14 and Mapping of KCNC1/Kv3.1 to Mouse 7/Human 11p14.3-p15.2 and KCNA1/Kv1.1 to Human 12p13'
[47]: Jiang et al. (2016), 'Chapter 4 - Involvement of cortical fast-spiking parvalbumin-positive basket cells in epilepsy'
[48]: Erisir et al. (1999), 'Function of Specific K+ Channels in Sustained High-Frequency Firing of Fast-Spiking Neocortical Interneurons'

Table 4.2: 25 selected genes for the n = 1213 Patch-seq data set, ranked in descending ℓ_2 norm order of first layer weights.

Data	Genes
M1	*Sst, Vip, Pvalb, Gad1, Coro6, Ndst3, Synpr, Tac2, Ndn, Htr3a, Lamp5, Unc13c, Cplx1, Enpp2, Sparcl1, Atp1a3, Dlx6os1, Col24a1, Plch2, Tiam2, Kcnab1, Tafa1, Ptk2b, Igf1, Magel2*
V1	*Vip, Synpr, Pdyn, Fxyd6, Trhde, Cacna2d3, Ptpru, Kcnc2, Pvalb, Hpse, Pcp4l1, Egfem1, Lamp5, Penk, Pde11a, Necab1, Chodl, Cpne7, Fstl4, Crtac1, Zfp536, Grm1, Shisa6, Kit, Grin3a*

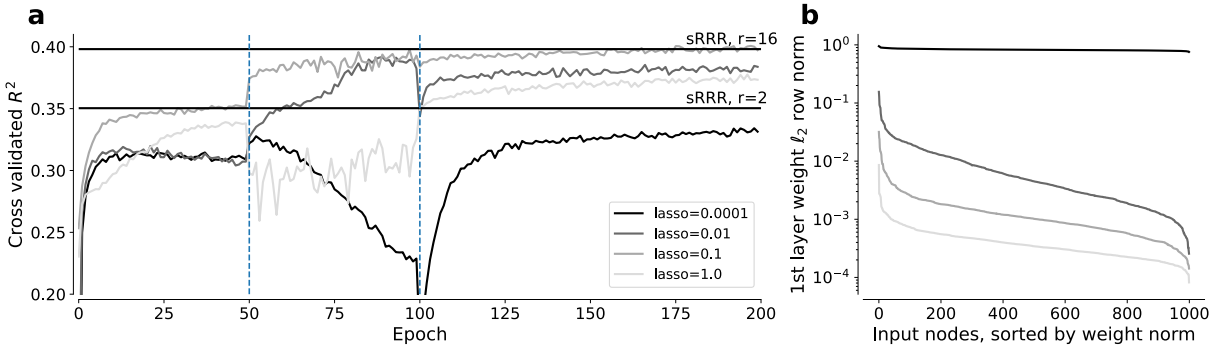

Figure 4.4: Effect of the group lasso penalty strength, M1 data set. **(a)** Validation (solid lines) R^2_{val} of the sBNN model, depending on the value of the lasso penalty (color-coded, see legend). Analogous to Figure 4.2. **(b)** The ℓ_2 norms of the first layer weight for each input unit just before pruning, depending on the value of the lasso penalty. Vertical axis is on the log scale.

4.2.4 Visualization

As promised, we can construct a two-dimensional nonlinear embedding after training the network and passing through the whole data set of M1 and V1[4] (see Figure 4.5). Two-dimensional bottleneck linear outputs for the M1 and V1 data set are shown in Figure 4.5 a,b (middle), respectively. We can observe that major cell families are well separated in the embeddings, as per case for the sRRR model (Figure 3.2). For the $r = 2$ sRRR model, however, we cannot observe a clear separation between *Sst* and *Vip* families in Figure 3.2 c for the M1 data set, even though they are known to be different in both transcriptome and electrophysiology. Indeed, higher ranks are necessary to show their differences [11]. Our sBNN model, however, separates them with only two latent dimensions, demonstrating the use of nonlinear computational units when the bottleneck's dimension is constrained.

4: Note that we train 2 different networks, i.e. two sBNN models, one with M1 data and one with V1 data, from scratch, before passing through the whole data set through respective networks.

Autoencoder gene predictions and alloencoder electrophysiological predictions can be intuitively overlayed on the same bottleneck representation in Figure 4.5 shown on the left and right, respectively. We pick nine exemplary genes that were selected by the model and can verify that for instance *Sst* and *Pvalb* marker genes are predicted to be highly expressed in *Sst* and *Pvalb* interneurons respectively (Figure 4.5 a, M1 data set). Likewise, predicted high expression of *Vip* is correctly assigned to *Vip* interneurons in the V1 data set (Figure 4.5 b). Similarly, nine randomly picked electrophysiological features allows us to reason which genes could be responsible for their high or low prediction. As such, the sBNN model also allows for meaningful biological interpretations *beyond* the level of major cell families: for instance for *Vip* neurons (Figure 4.5a, purple colors) we can see a separation between different transcriptomic types in the latent space, with differences in several predicted electrophysiological properties such as the membrane time constant and rebound, and correlated differences in gene expression e.g. of *Ptk2b* (Figure 4.6).

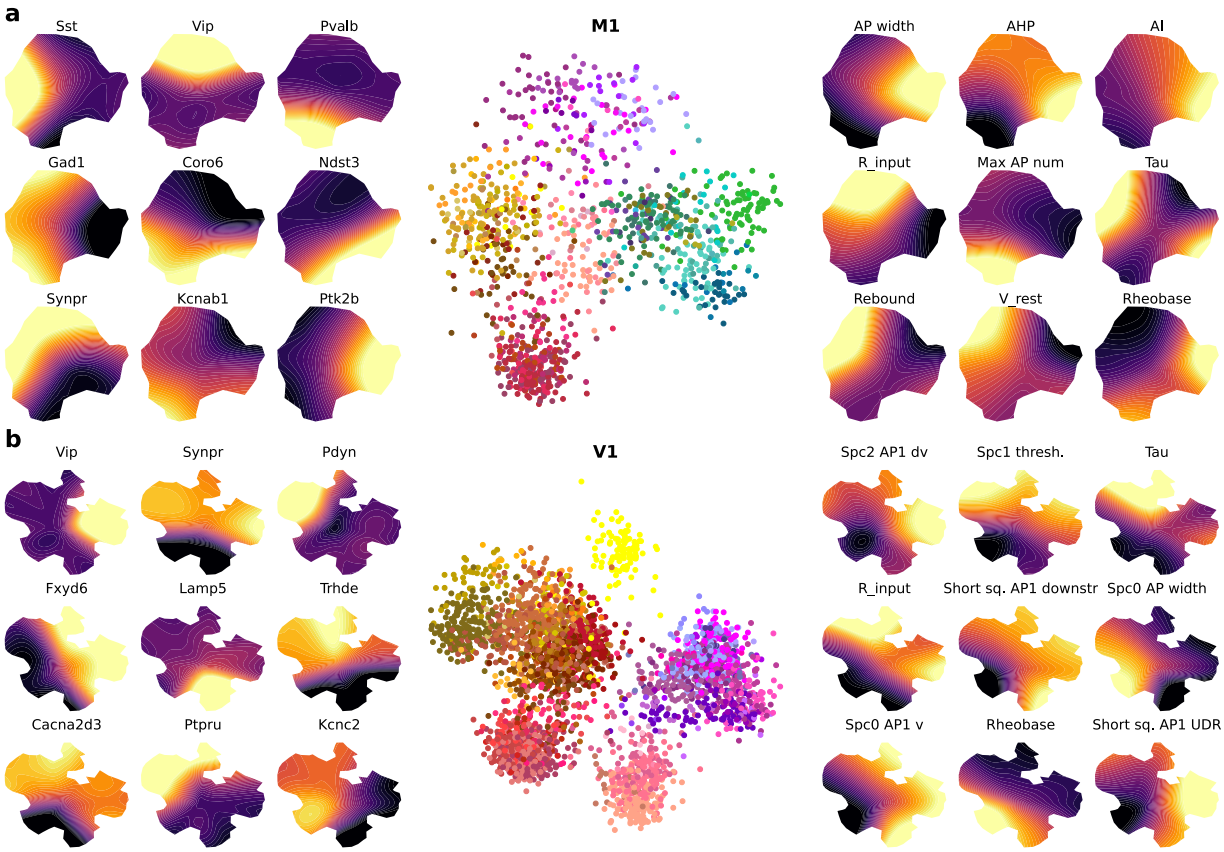

Figure 4.5: Latent visualization with gene and electrophysiological feature overlays. (a) M1 data set. Middle: two-dimensional bottleneck activations after passing the whole data set of $n = 1213$ neurons through. Left: autoencoder model predictions for nine exemplary genes out of 25 selected genes. Right: alloencoder model predictions for nine exemplary electrophysiological features. Colors correspond to transcriptomic types. Red: *Pvalb*; yellow: *Sst*; purple: *Vip*; salmon: *Lamp5* interneurons; green/blue: excitatory *Pyramidal* neurons. **(b)** V1 data set. Analogous to a.

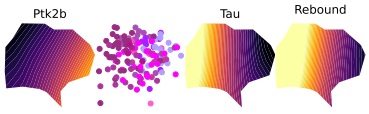

Figure 4.6: SBNN latent visualization, zoomed-in. Zoomed-in version of Figure 4.5 a for *Vip* neurons. The *Ptk2b* ion channel gene, rebound and membrane time constant overlays are shown.

4.3 Sparse Bottleneck Neural Networks beyond Patch-seq

Sparse bottleneck neural networks need not be limited in their application to Patch-seq data sets only. Indeed, sBNN's design (Section 4.1.1) suggests its use is most relevant for paired data sets where it is assumed that a lower-dimensional representation from high-dimensional input space can predict most of the variability in the low-dimensional output space, and where in addition it is believed that many predictors can be 'thrown away'.

Here, we demonstrate sBNN's competing performance beyond Patch-seq, and apply it to a CITE-seq data set [49] of $n = 7652$ cord blood mononuclear cells [5].

4.3.1 CITE-seq Data Set

Whereas with Patch-seq experiments we obtain transcriptomic read counts and electrophysiological measurements from *Patch*-clamp experi-

[49]: Stoeckius et al. (2017), 'Simultaneous epitope and transcriptome measurement in single cells'

5: Data set is publicly available at NCBI.

ments, with CITE-seq[6] experiments we obtain read counts and *epitope* measurements such as *CD3, CD4* and *CD8* (antibody-derived tag levels) which are relevant for immunological protein marker detection. The feature dimensionality for CITE-seq is $p = 1000$ and $q = 13$.

6: CITE for for "Cellular Indexing of Transcriptomes and Epitopes".

4.3.2 Predictive Performance

Applied to the CITE-seq data set, cross-validated performance R^2_{val} of the $r = 2$ sRRR model selecting again 25 genes is 0.23 ± 0.04 and reaches $0.35 \pm .06$ for the $r = 13$ or full-rank sRRR model (Table 4.1). Interestingly, the sBNN model with two-dimensional bottleneck reaches $R^2_{val} = 0.38$ even outperforming the full-rank linear counterpart (Table 4.1).

4.3.3 Gene Selection

The sBNN model selects *Ms4a1, Cd8b, Gnly, Ighm, Nng7, Tcl1a*, etc. (Figure 4.7 a) Within this list, *Cd7, Cd3d, Cd8a, Cdab, Ighm* and *Ighd* are genes responsible for *CD3, CD7, CD8* and *IgM* transmembrane protein (sub)units. Out of $p = 1000$ genes, therefore, sBNN selects highly relevant genes for CITE-seq: it selects those genes coding for protein subunits that either directly constitute or are highly relevant for the *epitope*, that is, the part of the protein to which the antibody attaches in the experiment.

[50]: Luecken et al. (2019), 'Current best practices in single-cell RNA-seq analysis: a tutorial'

[51]: Maaten et al. (2008), 'Visualizing data using t-SNE'

[52]: McInnes et al. (2018), 'UMAP: Uniform Manifold Approximation and Projection'

4.3.4 Visualization

Figure 4.7: SBNN latent visualization with gene and electrophysiological feature overlays for CITE-seq. Analogous to Figure 4.5 a,b but for CITE-seq. $n = 7652$. Colors correspond to immune populations [49]. Red: CD8 T cells; green: CD16 monocytes; yellow: CD14 Monocytes; blue: natural killer cells; aquamarine: CD4 T cells; magenta: B cells; brown: precursors.

4.4 Related Work

A multitude of linear and nonlinear methods for low-dimensional visualization of non-paired 'one-view' data are used in single-cell literature [50]. These range from principal component analysis (PCA) and kNN-based methods such as t-SNE [51], UMAP [52], or PHATE [53], to autoencoder and variational inference frameworks such as DCA [54], SAUCIE [55], or scVI [56]. Even though the latter show great potential of nonlinear para-

[53]: Moon et al. (2019), 'Visualizing structure and transitions in high-dimensional biological data'

[54]: Eraslan et al. (2019), 'Single-cell RNA-seq denoising using a deep count autoencoder'

[55]: Amodio et al. (2019), 'Exploring single-cell data with deep multitasking neural networks'

[56]: Lopez et al. (2018), 'Deep generative modeling for single-cell transcriptomics'

metric models for single-cell data analysis, they focus on one modality, namely transcriptomics.

Analyzing paired joint-view data, on the other hand, remains a challenge [26]. Most methods that are being used, focus on the symmetric integration of the two modalities into one joint representation. This has been approached using adaptive combination of kNN graphs [57]; multi-view extension of factor analysis [58]; coupled autoencoders [59, 60] constrained to have similar latent spaces; or joint variational autoencoders [61–64]. In fact, a `NeurIPS` competition focused on multimodal data integration of single-cell data [65] and offered three different challenges, including prediction of one modality from another and joint embedding. Most of the winners were based on deep-learning models and autoencoders [66].

These algorithms are very promising but most of them do not conform to all of our design principles, that is, to treat both modalities asymmetrically — directly predicting one modality from another —, to select predictors, and to produce meaningful (two-dimensional) embeddings. Instead they either treat the two modalities symmetrically, or formulate a prediction task that does not aim at data exploration and does not generate low-dimensional embeddings. Second, most method do not aim at feature selection which we believe is important for biological interpretability. Third, the methods that generate low-dimensional latent spaces typically use latent dimensionality of ~128 which is not amenable for direct visualization.

[26]: Lipovsek et al. (2021), 'Patch-seq: Past, present, and future'

[57]: Hao et al. (2021), 'Integrated analysis of multimodal single-cell data'

[58]: Argelaguet et al. (2020), 'MOFA+: a statistical framework for comprehensive integration of multi-modal single-cell data'

[59]: Gala et al. (2019), 'A coupled autoencoder approach for multi-modal analysis of cell types'

[60]: Gala et al. (2021), 'Consistent cross-modal identification of cortical neurons with coupled autoencoders'

[61]: Ashuach et al. (2021), 'MultiVI: deep generative model for the integration of multi-modal data'

[62]: Gong et al. (2021), 'Cobolt: integrative analysis of multimodal single-cell sequencing data'

[63]: Gayoso et al. (2021), 'Joint probabilistic modeling of single-cell multiomic data with totalVI'

[64]: Lotfollahi et al. (2022), 'Multigrate: single-cell multi-omic data integration'

[65]: Luecken et al. (2021), 'A sandbox for prediction and integration of DNA, RNA, and proteins in single cells'

[66]: Lance et al. (2022), 'Multimodal single cell data integration challenge: Results and lessons learned'

4.5 Discussion

A sparse bottleneck neural network is a deep neural network framework that leverages its inherent nonlinear computational units to outperform its linear counterpart: sparse reduced-rank regression. Inspired by the latter's intuitive design, a group lasso penalty is imposed on the first layer weight to select predictors, and a bottleneck is introduced to combine them in a nonlinear fashion into a low-dimensional latent representations used for prediction and visualization.

We have seen that R^2_{val} can improve significantly when we introduce nonlinear computations in the prediction task, and that the efficacy of sBNNs need not to be limited to Patch-seq data sets, but works very well for CITE-seq too. It would be interesting to see the application of sBNNs on many more kinds of paired data sets as well, for instance not limited to biology.

We speculated at the beginning of Chapter 4 that introducing nonlinear computational units could benefit the task of predicting electrophysiology, because decades-old neuroscientific research in living biological systems dictates that observed neuronal firing patterns are highly nonlinear, with membrane voltage fluctuations varying rapidly on millisecond timescales. In Chapter 5, we will move beyond just statistical approaches to describe the relationship between transcriptome and electrophysiology and make a step towards deterministic models, that is the Hodgkin-Huxley model.

Biophysical Modeling 5

Chapter 3 (Sparse Reduced-Rank Regression) on page 15 and Chapter 4 (Sparse Bottleneck Neural Networks) on page 25 introduced statistical tools that can be employed to learn a relationship between variables, namely a response variable such as the observed electrophysiology of a neuron and a predictor variable such as the observed gene expression levels in a neuron. These tools can be powerful: if the parameters of such a relationship can be learned effectively, for instance by minimizing a certain cost or error, then predictions can be made: given *new* gene expression levels, the learned mapping gives you predicted electrophysiology. Consequently, if the experimenter has access to the nucleus of a cell, and can give you this transcriptomic information, you, as a data analyst, can predict how the cell would respond to current injection (in the form of summarizing statistics), prior to having performed that experiment.

As touched upon in Section 3.3, these statistical tools do not, however, tell you which causal mechanisms lead to observed firing patterns, given knowledge of which genes are expressed and which are not. Indeed, when certain genes are expressed, certain proteins will be prevalent in the cell. In the case of neurons, specific genes give rise to typical proteins called ion channels that are located in the outer layer or membrane of the cell. Ion channels, as their name adeptly implies and as we will introduce in Section 5.2, allow for the free movement of specific ions through the cell's membrane. Those small currents lead to rapid fluctuations in the cell's membrane voltage and are called action potentials, and their typical shape and rapid succession give rise to the observed electrophysiology of the cell.

The biological design of a neuron is therefore incredibly intricate, specific and directed. If we want to get a deeper understanding of the exact biophysical mechanisms that lie at the core of typical firing patterns, we would need to go beyond the learned parameters in statistical frameworks. We would need access to a full description of the biophysics in a neuron, or a model that explains the experimental observation. This is the field of biophysical modeling.

5.1 Motivation

In biophysical modeling, *modelers* attempt to mathematically formulate the underlying *physics* in a *biological* system[1]. With a set of (differential) equations that explains the physics we think responsible for an observation in biology, the model output is matched to the experimental outcome.

In the pursuit of such a great and complex endeavor, the modeler has to make decisions on different scales. First, as is the case in many (biological) systems, not every part of the whole has a physical counterpart or mathematical description to it (yet). As we will see in Section 5.2, the Hodgkin-Huxley model describes how ions that flow through specific

1: Similarly for instance, in bio*chemical* modeling, the *modeler* attempts to mathematically describe the underlying *chemistry* in a *biological* system.

channels in the outer layer of the cell can be interpreted as currents and the outer layer itself as a capacitance or battery that holds a potential difference between the inner and outer environment of the cell. Potential differences located in parallel to currents and a capacitance describe a physical circuit that is well understood in physics and well described in mathematics. In contrast, the path from genes being transcribed to messenger RNA, which in turn translate to functional proteins in the cell, including those ion channels in the cell's membrane, is incredibly complex. This would either involve fitting a huge amount of parameters in a very big and complex dynamical system, which could take years even on the best computing infrastructure available, or interpreting this part in the system as a *black box*. In black-box modeling, a system is interpreted in terms of its inputs and outputs. The system itself or the logic therein remains obscure (*black*) to the modeler: when the output/input relationship is learned, a black-box modeler cares less about the internal mechanisms that explain the flow from input to output. Sparse reduced-rank regression and sparse bottleneck neural networks can be interpreted as black-box models: a linear or nonlinear relationship is learned from genes to electrophysiology, but this regression does not explain *how* we get to the observed firing patterns of a neuron or *which* biophysical mechanisms underlie them.

When the modeler has decided for which parts of the system a mathematical framework can be built, then still does the modeler need to choose *which* set of equations describe the observation best. One would generally want to keep the dynamical system as simple as possibly but not simpler.[2] Simpler models allow for better understandings, but when the apparent need for adding complexity in the model arises, it should not be ignored. In fact, it is the added complexity deemed necessary by the modeler that can guide experimental design and new discovery in biology. This is one of the greatest strengths in biophysical modeling for which we will see an example in Section 5.2.

2: Often attributed to Einstein referring to him saying during a lecture:

> ··· the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience.

Finally, the parameters of such a model need to be *fit* to the observation in biology: we want the model output to match the experimental outcome as best as we can. When we have variables that summarize the observation, we can attempt to match those, for instance through minimizing an error: the difference between the feature values derived from the model with those derived from experiment. It could be that many parameter combinations do sufficiently well of a job, however, which again, can be of great biological significance. Depending on energy considerations, for instance, a neuron might use a different set of ion channels to its disposal in the membrane [67], which is reflected in a different set of parameter values in the model. In Chapter 6, we will see how we can fit models with some of the latest techniques in machine learning and moreover obtain probabilities for every possible parameter combination.

[67]: Deistler et al. (2022), 'Energy-efficient network activity from disparate circuit parameters'

5.2 The Hodgkin-Huxley Model

In the work of Hodkgin and Huxley [68], in squids, for the first time it was demonstrated how ionic mechanisms underlie the initiation and propagation of action potentials: rapid membrane voltage fluctuations in

time. In 1963, they received the Nobel Prize in Physiology or Medicine for their revolutional work.

5.2.1 Background

In the Hodgkin-Huxley model, depicted by a physical circuit in Figure 5.1, the outer lipid bilayer of the cell functions as a capacitance C_m, as it keeps the gradient of charges between the inner and outer environment of the cell *intact*. This lipid bilayer is indeed largely impermeable to ions and hydrophobic but does contain so-called leaky channels denoted by conductances g_L through which random ions can flow down their electrochemical gradient E_L. The cell's membrane hosts, besides leaky channels, also a spectrum of proteins called ion channels (denoted by conductances g_n) that allow the free flow of specific ions down their gradient when specific conditions are met. This condition is usually the current membrane voltage V, that is the voltage difference between the inner and outer environment of the cell, that reaches a certain threshold V_T. This threshold can be reached for instance because sufficient other neurons have excited this one, electrically or chemically (through neurotransmitters), or because an experimenter injected current, which is what we will mean with I_p throughout this thesis[3]. When that threshold is reached, the cell displays an *all or none* behavior: rapid fluctuations of the cell's membrane voltage V will inevitably follow: so-called action potentials are generated.

3: Moreover, besides leaky protrusions and specific ion channels, the membrane also hosts ion pumps, which help keep the natural gradient intact and ion exchangers, that exchange ions between the inner and outer environment of the cell. Both ion exchangers and ion pumps are often represented by I_p too in the physical circuit depicted by Figure 5.1.

[69]: Oldham (2008), 'The doctrine of description : Gustav Kirchhoff, classical physics, and the "purpose of all science" in 19th-century Germany'

We apply Kirchhoff's law [69] to a physical circuit as demonstrated in Figure 5.1. A law that states that the sum of all currents meeting in a point amounts to zero. We can take the convergence point of currents in the node depicted right below E_L in Figure 5.1. When an experimenter injects current I_p flowing from the right into the node, it is divided into linear currents through leaky channels $g_L\,(V - E_L)$, nonlinear currents through ion channels $g_n\,(V - E_n)$, and a capacitance current $C\frac{dV}{dt}$, which we can state as:

$$I_p = g_L\,(V - E_L) + \sum_n g_n\,(V - E_n) + C_m\frac{dV}{dt}, \tag{5.1}$$

and where the $\sum_n$ is introduced to state that there can be multiple and specific ion channels. In the case of the squid axon that was studied by Hodkgin and Huxley [68], two ion channels allowing the flux of only sodium (Na^+) with current $g_{Na^+}(V - E_{Na^+})$ and potassium (K^+) with current $g_{K^+}(V - E_{K^+})$ respectively, were introduced to explain the action potential generation observed. When the membrane threshold is reached, Na^+-channels open and create a positively charged rapid influx of Na^+. This creates a fast depolarization of the membrane voltage. Due to this rapid increase in the membrane voltage, however, K^+-channels respond, and in turn allow the rapid efflux of K^+ ions, creating a rapid *repolarization* of the membrane voltage. During the same time, Na^+-channels inactivate due to a closing of one of the subunits or gates within the channel.

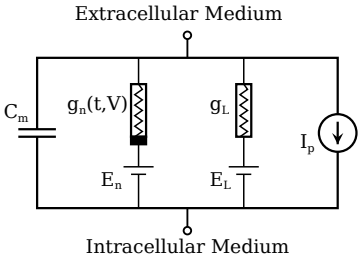

Figure 5.1: Physical circuit describing the Hodgkin-Huxley model. The outer lipid bilayer of the cell is depicted by a capacitance C_m, the voltage difference between the inner and outer environment by V, specific ion channels by conductances g_n, leak currents by conductance g_L, possible current sources or so-called ion pumps and ion exchangers by I_p and the electrochemical gradient driving the flow of ions by E_L. From Wikipedia.

As the gates open and close in response to the membrane voltage value and in function of time, Equation 5.1 can be rewritten as

$$\begin{aligned} I_p &= g_L\,(V - E_L) + g_{Na^+}\,(V - E_{Na^+}) + g_{K^+}\,(V - E_{K^+}) + C_m \frac{dV}{dt} \\ &= g_L\,(V - E_L) + \overline{g}_{Na^+} m^3 h\,(V - E_{Na^+}) + \overline{g}_{K^+} n^4\,(V - E_{K^+}) + C_m \frac{dV}{dt}, \end{aligned} \tag{5.2}$$

where m is the probability with which one *activation* gate opens and h the probability with which the *inactivation* gate opens in the $Na+$-channel. Indeed, Na^+-channels have 4 subunits, 3 activation gates and one inactivation gate, all needing to be in the open state for Na^+ ions to flow into the cell. Analogously, K^+-channels contain 4 subunits, all activation gates, that are in the open state with probability n. When all gates are in the open state, Na^+ influx and K^+-efflux happens with maximal conductances $\overline{g}_{Na^+}$ and $\overline{g}_{K^+}$ respectively.

The probability with which (in)activation gates are in the open state are modeled as

$$\frac{dn}{dt} = \alpha_n\,(V(t))\,(1 - n) - \beta_n\,(V(t))\,n \tag{5.3}$$

, where α_n denotes the membrane voltage dependent rate of opening and β_n the membrane voltage dependent rate of closing ($n \in (m, h, n)$)[4].

4: Exact dependencies of α and β on the membrane voltage V for both Na^+-influx and K^+-efflux in Hodkgin and Huxley [68].

During the refinement of the Hodgkin-Huxley model, Hodgkin and Huxley inevitably introduced powers in Equation 5.2, as they noticed it better explained the shape of an action potential in the observed firing pattern of a neuron in a squid. They did not know yet, however, that these correspond to actual gates or subunits within the channels themselves. Interestingly, these powers were later confirmed by experiments in biology: it turns out that multiple *gates* (protein subunits) need to be in the open state within the same ion channel in order for ions to flow through the channel down their gradient. A great example of biophysical modeling creating testable hypotheses end guiding experimental design.

5.2.2 The Hodgkin-Huxley-based Model for Patch-seq

The Hodgkin-Huxley model introduced in Section 5.2 is of great importance till this day in neuroscience to accurately describe and fit models to observed electrophysiology in a wide range of neurons. In fact, for the purpose of this thesis, we use a single-compartment Hodgkin-Huxley-based model (henceforth sometimes abbreviated as the HH model) described by Pospischil et al. [70] that was designed to reproduce electrophysiological behavior of a wide variety of neurons across species with a minimal set of ion channels. To account for the variability across excitatory and inhibitory cortical neurons, we add additional ion channels explained in Hay et al. [71] and introduce r_{SS}, a parameter influencing how rapid gates reach open and closed steady states in some sodium and and potassium currents. Without these modifications we could not fit wider *AP widths* for instance observed in firing patterns of Pyramidal cells in our data set (see later).

As we have seen in Section 5.2, the Hodgkin-Huxley model solves the following ODE $V_m(t) = f\,(V_m(t), \boldsymbol{\theta})$ for $V_m(t)$, the membrane voltage as

a function of time, which in our case amounts to:

$$\frac{dV_m(t)}{dt} = \frac{1}{C}\left(I_{Na} + I_{Nat} + I_{Kd} + I_M + I_{Kv3.1} + I_L + I_{leak} - I_{inj} - I_{noise}\right) \tag{5.4}$$

$$I_{Na} = \overline{g}_{Na}\, m^3\, h\, (E_{Na^+} - V_m(t)) \tag{5.5}$$

$$I_{Nat} = \overline{g}_{Nat}\, \hat{m}^3\, \hat{h}\, (E_{Na^+} - V_m(t)) \tag{5.6}$$

$$I_{Kd} = \overline{g}_{Kd}\, n^4\, (E_{K^+} - V_m(t)) \tag{5.7}$$

$$I_M = \overline{g}_M\, p\, (E_{K^+} - V_m(t)) \tag{5.8}$$

$$I_{Kv3.1} = \overline{g}_{Kv3.1}\, v\, (E_{K^+} - V_m(t)) \tag{5.9}$$

$$I_L = \overline{g}_L\, q^2\, r\, (E_{Ca^{2+}} - V_m(t)) \tag{5.10}$$

$$I_{leak} = \overline{g}_{leak}\, (E_{leak} - V_m(t))\,. \tag{5.11}$$

Here, $\overline{g}_x$ and E_x denote the maximum channel conductance and reversal potential of membrane ion channel x respectively. C is the membrane capacitance and $I_{inj} = 300pA$ denotes the magnitude of experimental current injected between current stimulation onset at 100 ms and stimulation offset 700 ms. In order to model small membrane voltage fluctuations observed experimentally, we further introduce Gaussian current noise $I_{noise} \sim \mathcal{N}(10, 1)$ at every time point.

Analogously to Section 5.2, ion channel activation and inactivation gates follow dynamics $\frac{dx}{dt} = \alpha_x\,(V_m(t))\,(1-x) + \beta_x\,(V_m(t))\,x$, where $x \in \left\{m, h, \hat{m}, \hat{h}, n, p, v, q, r\right\}$. Opening α_x and closing β_x rate constants depend on the membrane voltage $V_m(t)$ as previously described by [70, 71]. In order to account for the 25°C temperature at which Patch-seq experiments were performed, we use a temperature coefficient $Q_{10} = 2.3$ to scale the kinetics with which gates in ion channels open and close. Parameter r_{SS} further scales the rates with which sodium and potassium currents with maximal conductances $\overline{g}_{Na}$ and $\overline{g}_{Kd}$ reach steady states.

We implement the model with the Brian2 toolbox developed by Stimberg, Brette, and Goodman [72] in `Python`, which can efficiently transpile and simulate models in C++.

[70]: Pospischil et al. (2008), 'Minimal Hodgkin-Huxley type models for different classes of cortical and thalamic neurons.'
[71]: Hay et al. (2011), 'Models of Neocortical Layer 5b Pyramidal Cells Capturing a Wide Range of Dendritic and Perisomatic Active Properties.'

Model Parameters

In Table 5.1, the 13 free parameters used in this HH model are described.

Summary Statistics

We automatically extract 23 electrophysiological features (Table 5.2) from the measured or simulated voltage traces $V(t)$. In Chapter 2, however, the pipeline focused on the extraction of electrophysiological features summarizing both depolarization and hyperpolarization characteristics of the cell. Here, we focus on traces with action potentials and keep only the membrane voltage response to current injections of 300 pA magnitude. Code can be found on GitHub.

Table 5.1: Description of parameters of the Hodgkin-Huxley-based model.

Model parameter	Prior range	Description
C	$[0.1, 15]\ \frac{\mu F}{cm^2}$	The membrane capacitance C measures how much charge can be stored per voltage difference V_m across the membrane.
R_{input}	$[20, 1000]\ M\Omega$	The input resistance R_{input}, equals the membrane voltage V_m deflection from resting state divided by injected current. The inverse is called the leak conductance g_{leak}.
τ	$[0.1, 70]\ ms$	Here, τ describes the time for the membrane potential to *increase* by a fraction of $(1 - 1/e)$, or 63 %, from its resting membrane state during the application of the positive 300 pA current pulse.
$\overline{g}_{Nat}$	$[0, 250]\ \frac{mS}{cm^2}$	Maximal conductance of the fast inactivating Na^+ current described by Hay et al. [71] and Colbert and Pan [73].
$\overline{g}_{Na}$	$[0, 100]\ \frac{mS}{cm^2}$	Maximal conductance of the Na^+ current described by Pospischil et al. [70] and Traub and Miles [74].
$\overline{g}_{Kd}$	$[0, 30]\ \frac{mS}{cm^2}$	Maximal conductance of the delayed rectifier K^+ current described by Pospischil et al. [70] and Traub and Miles [74].
$\overline{g}_{M}$	$[0, 3]\ \frac{mS}{cm^2}$	Maximal conductance of the slow non-inactivating *muscarinic* K^+ current, also found in Pospischil et al. [70] and Yamada, Koch, and Adams [75].
$\overline{g}_{Kv3.1}$	$[0, 250]\ \frac{mS}{cm^2}$	Maximal conductance of the fast non-inactivating K^+ current described by Hay et al. [71] and Rettig et al. [76].
$\overline{g}_{L}$	$[0, 3]\ \frac{mS}{cm^2}$	Maximal conductance of the high-threshold Ca^{2+} current, found in Pospischil et al. [70] and Reuveni et al. [77].
E_{leak}	$[-130, -50]\ mV$	Reversal potential of the leak current.
τ_{max}	$[50, 4000]\ ms$	Time constant describing how rapid the *muscarinic* current channel opens (see g_M).
V_T	$[-90, -35]\ mV$	Parameter that can adjust the AP threshold.
r_{SS}	$[0.1, 3]$	Rate to steady state (SS). Parameter introduced to change how rapid gates reach open and closed steady states in Na^+ current with maximal conductance $\overline{g}Na$ and K^+ current with maximal conductance $\overline{g}_{Kd}$.

Table 5.2: Description of electrophysiological features summarizing Hodgkin-Huxley-based model simulations. *To make their distribution more Gaussian, these features are additionally log-transformed, except for the *AP average amp adapt* for which we used the *sigmoid* transformation.

Electrophysiological feature	Description
AP threshold	Membrane voltage at the time where the first derivative of the voltage w.r.t. time reaches a threshold, which elicits the first AP.
AP amplitude	Height of the 1st AP, measured from threshold to maximum voltage.
AP width	Width at half height of the 1st AP
AHP	Afterhyperpolarization. Depth of the membrane voltage drop after the 1st AP, measured from AP threshold.
3rd AP threshold	Analogous to *AP threshold* but for the 3rd AP.
3rd AP amplitude	Analogous to *AP amplitude* but for the 3rd AP.
3rd AP width	Analogous to *3rd AP width* but for the 3rd AP.
3rd AHP	Analogous to *AHP* but for the 3rd elicited AP.
*AP count**	Number of elicited APs in the current injection window $100-700\,ms$.
*AP counts 1st 8th**	Number of elicited APs in $100-175\,ms$.
*AP count 1st quarter**	Number of APs in $100-250\,ms$.
*AP count 1st half**	Number of APs in $100-400\,ms$.
*AP count 2nd half**	Number of APs in $400-700\,ms$.
*AP amp adapt**	AP amplitude adaptation. 1st elicited *AP amplitude* divided by the amplitude of the 2nd elicited AP.
*AP average amp adapt**	AP average amplitude adaptation. Average ratio of all two consecutive AP heights as calculated by *AP amp adapt* during current injection window.
*AP CV**	Standard deviation divided by the mean of all AP amplitudes of APs elicited during the current injection window.
*ISI adapt**	Interspike interval (ISI) adaptation. ISI: time elapsed between two APs. ISI adapt: ratio of the 2nd ISI (between 2nd and 3rd elicited AP) to the 1st ISI (between 1st and 2nd elicited AP).
*ISI CV**	Standard deviation divided by the mean of all ISIs.
*Latency**	Time it takes to elicit the 1st AP, measured from current stimulation onset to *AP threshold*.
Rest V_m mean	Mean of the membrane voltage V_m before current stimulation onset $0-100\,ms$. Also called resting membrane potential.
V_m mean	Mean of the membrane voltage V_m during current stimulation window $100-700\,ms$.
V_m std	Standard deviation of the membrane voltage V_m during current stimulation window $100-700\,ms$.
V_m skewness	Skewness of the membrane voltage V_m during current stimulation window $100-700\,ms$.

5.2.3 Hodgkin-Huxley-based Models for Fitting Experimental Electrophysiology.

The adapted Hodgkin-Huxley-based model fits firing patterns observed with Patch-seq experiments across a wide variety of neurons. In Figure 5.2, Figure 5.3 and Figure 5.4, we showcase the model's capability of fitting firing patterns obtained from a fast-spiking *Pvalb Kank4* interneuron, irregular firing *Sst Hpse* neuron and *L4/5 IT_1* Pyramidal cell, respectively. True experimental observations are shown on top in black. These are the cell's membrane voltage responses to 300 pA current stimulation. Three HH model simulations with smallest Euclidean distance to the experimental observation in 23-dimensional electrophysiological feature space, are shown in gray.

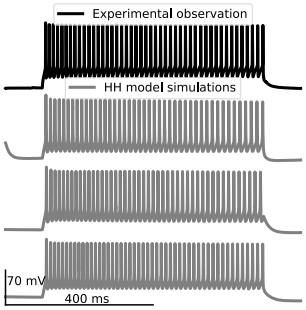

Figure 5.2: HH Model Fits Experimental Observations from Patch-seq Firing pattern of a fast-spiking *Pvalb Kank4* interneuron is shown on top in black. Three HH model simulations with smallest Euclidean distance in electrophysiological feature space to the observation are shown in gray.

Each model simulation is generated with one particular 13-dimensional HH model parameter combination (Table 5.1), also called a *point estimate*. In Chapter 6, we will come across a machine-learning based framework that derives probabilities over the full space of possible model parameter combinations. As such, we do not only know single best solutions, we know of all HH model parameter combinations how likely they fit the experimental observation.

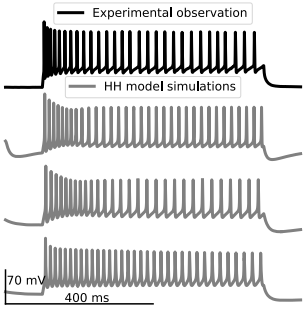

Figure 5.3: Analogous to Figure 5.2. *Sst Hpse* interneuron.

5.3 Conclusion

The work of Hodkgin and Huxley [68] demonstratively revolutionized the field of neuroscience. The Hodgkin-Huxley model was a first of its kind, describing mathematically the mechanistic steps from ionic flows through channels in the cell membrane to observed firing patterns in neurons. In contrast to statistical tools introduced in Chapter 3 and Chapter 4, the Hodgkin-Huxley model is not a black box. We can peer into the mechanics of the system and understand *how* it goes from model parameters including the membrane capacitance and ion channels to action potential generation and firing patterns.

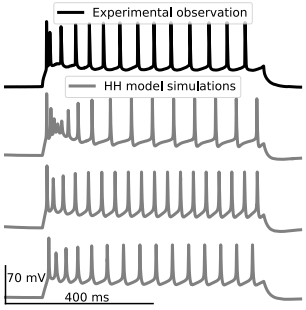

Figure 5.4: Analogous to Figure 5.2. Pyramidal *L4/5 IT_1* neuron.

Still to this day, many adaptations to the Hodgkin-Huxley model allow for the accurate fitting of observed electrophysiology in a variety of neurons, such as Patch-seq experimental observations in mouse motor cortex, as we qualitatively deduced in Section 5.2.3.

As touched upon, however, we do not know how *probable* certain parameter combinations are that lead to similar model outputs fitting the same experimentally observed firing pattern. How do we know which parameter combinations are likely and which less likely? Can very different parameter combinations produce similar firing patterns? What could that mean for the neuron in living systems? Advanced statistical tools in machine learning can infer probabilities over the entire space of biologically reasonable model parameters, and that will be the topic of Chapter 5.

In Chapter 8, we will attempt to bridge the genetic identity of neurons in the form of gene expression levels to their phenotypic identity in the form of observed electrophysiology with the combination of a statistical model introduced in Chapter 3 and a biophysical model introduced here.

Simulation-based Inference 6

In the subfield of machine learning called simulation-based inference (SBI), scientists attempt to *infer* the parameters of a model that explain observed data, *based* on *simulations* generated from the model itself. Beyond providing single numerical values for all parameters, which produce a single simulation sufficiently close to the observation, they attempt to provide uncertainties over all possible parameter values. Uncertainty distributions provide the modeler with probabilities for each combination of parameter values. Consequently the modeler knows which combinations of parameter values are likely to generate simulations sufficiently close to the observed data and which parameters are not likely to do so.

SBI has recently seen fruitful and real physical applications ranging from identifying mechanistic models of neural dynamics [78] and the energy management thereof [67] to gravitational wave parameter estimation [79] and particle physics [80]. Indeed, recent work in SBI on enhancing network architectures and training frameworks [81, 82] has shown its superiority over traditional Approximate Bayesian Frameworks (ABC) [82].

There are five ingredients necessary for SBI applied to Patch-seq:

- i. a prior distribution over model parameters, that is, initially assigned probability values to each and every combination of parameter values,
- ii. a model capable of generating simulations that can explain the data,
- iii. features summarizing the simulated data,[1],
- iv. a network architecture, or differently put, a neural network that can learn the mapping from features to probability distributions over model parameters.
- v. features summarizing *experimental* data, serving as input to the simulation-based trained network.

In Chapter 5 (Biophysical Modeling) on page 35, we introduced the HH model, a sophisticated highly nonlinear model flexible enough to generate voltage traces explaining raw electrophysiological recordings obtained with Patch-seq. We also have a `Python`-based pipeline, much like the one introduced in Chapter 2, for the automated extraction of electrophysiological features from simulations (see Section 5.2.2). Indeed, it are our expert-defined features including the *latency*, *AP amplitude* and *firing rate* that summarize how the cell responds to current injection.

Plausible parameter ranges, defined by the uniform prior distribution, were also introduced in Chapter 5 (Biophysical Modeling) on page 35, in Table 5.1, motivated by biology. Indeed, it is up to the *Bayesian* statistician to introduce as much knowledge about the system a priori available, that is, to capitalize on biologically available prior knowledge and introduce it into the Bayesian inference procedure.

[78]: Gonçalves et al. (2020), 'Training deep neural density estimators to identify mechanistic models of neural dynamics'

[67]: Deistler et al. (2022), 'Energy-efficient network activity from disparate circuit parameters'

[79]: Dax et al. (2021), 'Real-Time Gravitational Wave Science with Neural Posterior Estimation'

[80]: Brehmer et al. (2022), 'Chapter 16: Simulation-Based Inference Methods for Particle Physics'

[81]: Durkan et al. (2020), 'On Contrastive Learning for Likelihood-free Inference'

[82]: Lueckmann et al. (2021), 'Benchmarking Simulation-Based Inference'

1: This step is not *strictly* necessary, but can reduce the dimensionality to make inference easier. Additionally, in neural science one is often more interested in summarizing features like the firing rate, rather than the exact precise timings of each of the action potentials, or membrane voltage values.

Equipped with a prior distribution over parameters, the model to generate simulations with parameters and the pipeline to derive features from the simulations, we can move on to the 4th ingredient: training a neural network that learns a mapping from the electrophysiological features to HH model parameters, which is what we describe next.

6.1 Neural Posterior Estimation

Within the field of SBI, scientists can train a *neural* network to directly *estimate* a *posterior* distribution, hence termed neural posterior estimation or NPE. Training data is usually made up of data observed in the real world such as images in computer vision, text in large language models or genetic and electrophysiological data to train bottleneck neural networks as shown in Chapter 4. The neural network might learn to say whether there is a dog or cat in the image for which we have the true label, or learn how to predict the next word when it receives text, or predict the firing rate based on the expression of a few genes. The points is that, in all previous cases, we have verified or true examples representing both the input and output space (image of a dog, and the label 'dog', a piece of text and the next word, expressions of genes *Cacna2d1* and *Canca2d3*, and the *AP width* of the neuron). In NPE, however, where we try to learn a mapping from summarizing statistics to model parameters, we do not have access to 'true' model parameters connected to experimentally observed data. Indeed, the experimenter used the Patch-seq protocol to provide us with raw electrophysiological recordings but cannot provide us with a 'true' model parameter combination connected to it one-to-one. So how do we train the neural network? We do so by using simulations. After all, that is why it is called simulation-based inference.

6.1.1 Simulated Training Data

In order to produce enough simulations that cover experimental observations [2], we need many parameter combinations from our biologically motivated uniform prior distribution p. We can therefore sample

$$\boldsymbol{\theta}^{(n)} \sim p(\boldsymbol{\theta}),$$

where $n \sim 15$ million. As described in Chapter 5, they set the values for parameters including the *membrane capacitance*, *sodium conductance* and so forth. As we have 15 million different parameter combinations, they each produce a different simulation: each a different membrane voltage response to 300 pA injection. From each and every one of those, electrophysiological features including the *latency* and *AP amplitude* can automatically be extracted as described in Chapter 2. To conclude, we end up with tuples $(\boldsymbol{\theta}, \mathbf{y})^{(n)}$ that serve as our (simulated) training data.

2: That is, we need enough simulated electrophysiology vectors that cover the space of 'true' experimentally observed ones.

6.1.2 Training

Equipped with simulated data $(\boldsymbol{\theta}, \mathbf{y})^{(n)}$, we are now in a position to learn a mapping from electrophysiological feature space $\mathbf{Y}$ to model parameter

space $\boldsymbol{\Theta}$. We do so by optimizing the parameters $\boldsymbol{\phi}$ of a neural network (our mapping) which we denote by $q_{\boldsymbol{\phi}}(\boldsymbol{\theta} \mid \mathbf{y})$. More precisely, however, we train what is called a neural density estimator. Instead of optimizing parameters $\boldsymbol{\phi}$ to predict the model parameter combination $\boldsymbol{\theta}^{(n)}$ directly based on the electrophysiology vector $\mathbf{y}^{(n)}$ [3], we want the neural network to output a full distribution over model parameters. A distribution that gives high probability to model parameter combinations that lead to simulations with very similar summary statistics to the ones we fed into the neural network.

3: This would resemble for instance sparse bottleneck neural networks where we optimize network parameters to directly predict electrophysiology vector $\mathbf{y}^{(n)}$ based on gene expression levels $\mathbf{x}^{(n)}$

How do we train such a neural network? We cannot minimize a mean-squared error as we introduced in Chapter 3, but we can minimize

$$\mathscr{L} = -\sum_{n} \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}^{(n)} \mid \mathbf{y}^{(n)})), \tag{6.1}$$

with respect to $\boldsymbol{\phi}$. That is, at each training step, with a current estimate of $\boldsymbol{\phi}$, each $\mathbf{y}^{(n)}$, when fed into the network, leads to parameters[4] describing a different probability distribution $q_{\boldsymbol{\phi}}(\boldsymbol{\theta} \mid \mathbf{y}^{(n)})$, which we can evaluate at $\boldsymbol{\theta}^{(n)}$, that is, we calculate $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}^{(n)} \mid \mathbf{y}^{(n)})$. We can do this for all n simulated training data samples, take the natural logarithm and sum them all up. This is a quantity we would want to maximize, or the minus of that a quantity that we would want to minimize.

4: Not to be confused with our HH model parameters. Rather, they could for instance be the mean and covariance parameterizing a Gaussian distribution. Specific examples will be introduced in Section 6.2.

We can interpret the neural density estimator $q_{\boldsymbol{\phi}(\boldsymbol{\theta}|\mathbf{y})}$ as a *surrogate* posterior, because when we feedforward $\mathbf{y}$, the neural network spits out parameters of a full distribution over model parameters, given known $\mathbf{y}$. But does it in any way come close to the *true* posterior $p(\boldsymbol{\theta} \mid \mathbf{y})$? As it turns out, provided the neural density estimator is flexible enough and we have an infinite amount of data to our disposal, a theoretical guarantee follows that minimizing $\mathscr{L}$ corresponds to minimizing the *forward* $\mathscr{KL}$-divergence between true and *surrogate* posterior: $\mathscr{KL}(p \mid\mid q) = \mathscr{KL}\left(p\left(\boldsymbol{\theta} \mid \mathbf{y}\right) \mid\mid q_{\boldsymbol{\phi}}\left(\boldsymbol{\theta} \mid \mathbf{y}\right)\right)$. Indeed, we can easily show that

$$\begin{aligned}
\mathscr{KL}(p \mid\mid q) &= \mathbb{E}_{p(\boldsymbol{\theta}|\mathbf{y})} \log\left(\frac{p\left(\boldsymbol{\theta} \mid \mathbf{y}\right)}{q_{\boldsymbol{\phi}}\left(\boldsymbol{\theta} \mid \mathbf{y}\right)}\right) && (6.2)\\
&= -\mathbb{E}_{p(\boldsymbol{\theta}|\mathbf{y})} \log\left(q_{\boldsymbol{\phi}}\left(\boldsymbol{\theta} \mid \mathbf{y}\right)\right) + \mathbb{E}_{p(\boldsymbol{\theta}|\mathbf{y})} \log\left(p\left(\boldsymbol{\theta} \mid \mathbf{y}\right)\right) && (6.3)\\
&\approx -\sum_{n} \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}^{(n)} \mid \mathbf{y}^{(n)})) + cte, && (6.4)
\end{aligned}$$

where we have used that $\mathbb{E}_{p(\boldsymbol{\theta}|\mathbf{y})} \log\left(p\left(\boldsymbol{\theta} \mid \mathbf{y}\right)\right)$ does not depend on $\boldsymbol{\phi}$ and that we can approximate the expectation w.r.t. $p\left(\boldsymbol{\theta} \mid \mathbf{y}\right)$ with the sum, also called a Monte Carlo estimate.

As the $\mathscr{KL}$-divergence is always positive, this quantity is minimal where $p(\boldsymbol{\theta} \mid \mathbf{y}) = q_{\boldsymbol{\phi}}(\boldsymbol{\theta} \mid \mathbf{y})$ or where the $\mathscr{KL}$-divergence is zero.

One can use stochastic gradient descent algorithms (SGD) or standard algorithms such as Adam [43] to optimize the objective in Equation 6.1.

6.1.3 Amortized Neural Density Estimator

Consider now a real-word experimental *o*bservation $\mathbf{y}^{(o)}$: real *latency*, *AP amplitude*, *AP width*, ... values derived from a raw experimental recording obtained with Patch-seq. We can use our simulation-based trained neural density estimator network $q_{\boldsymbol{\phi}^*}(\boldsymbol{\theta} \mid \mathbf{y})$ where $\boldsymbol{\phi}^*$ denotes

the *optimal* parameters $\boldsymbol{\phi}$ of the neural network, that is parameters for which the loss $\mathcal{L}(\boldsymbol{\phi}^*)$ is at a local minimum. We feedforward $\mathbf{y}^{(o)}$ leading up to inferred posterior $q_{\boldsymbol{\phi}^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$. Now we have full knowledge of the probability mass in the model parameter space defined by the prior. Given observed electrophysiology vectors, we know which model parameter combinations are likely (and which less likely) for the HH model to reproduce the electrophysiological recording. Indeed, the neural density estimator is *amortized*; we need only train the neural network once, yet we can sequentially feedforward electrophysiology vectors of all the cells in our data set and obtain $n = 955$ *surrogate* posterior distributions[5] $q_{\boldsymbol{\phi}^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$, another clear advantage of NPE[6].

5: This number is less than 1213 as previously suggested as the size of this data set. Not all cells have defined (i.e. not NaN and not infinite) values for the 23 electrophysiological features introduced in Section 5.2.2.

6: In Sequential NPE, one samples from obtained posterior $q_{\boldsymbol{\phi}^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$ to generate more simulations closer to the experimental recording than randomly drawn samples from the prior could. Using these simulations to train a second neural density estimator arguably draws the *surrogate* even closer to the *true* posterior. Yet, as the new samples were drawn based on only one cell's electrophysiology $\mathbf{y}^{(o)}$, the second trained network is finetuned to that cell only, and loses its amortization.

6.1.4 Conclusion

Except the kind of neural density estimators that are commonly used, we discussed all ingredients, to cook up a recipe that allows us to do inference, neural posterior estimation style (Figure 6.1). We sample from the prior distribution to obtain many HH model parameter combinations, that produce simulated firing patterns, from which we derive our favorite summarizing statistics. Combinations of HH model parameters with summarizing statistics can be used to train the neural density estimator, that, when optimized, can subsequently derive the posterior over HH model parameters given summarizing statistics of an experimentally observed firing pattern. HH model parameters obtained by sampling from high probability posterior regions are expected to produce simulations consistent with the experimental observation.

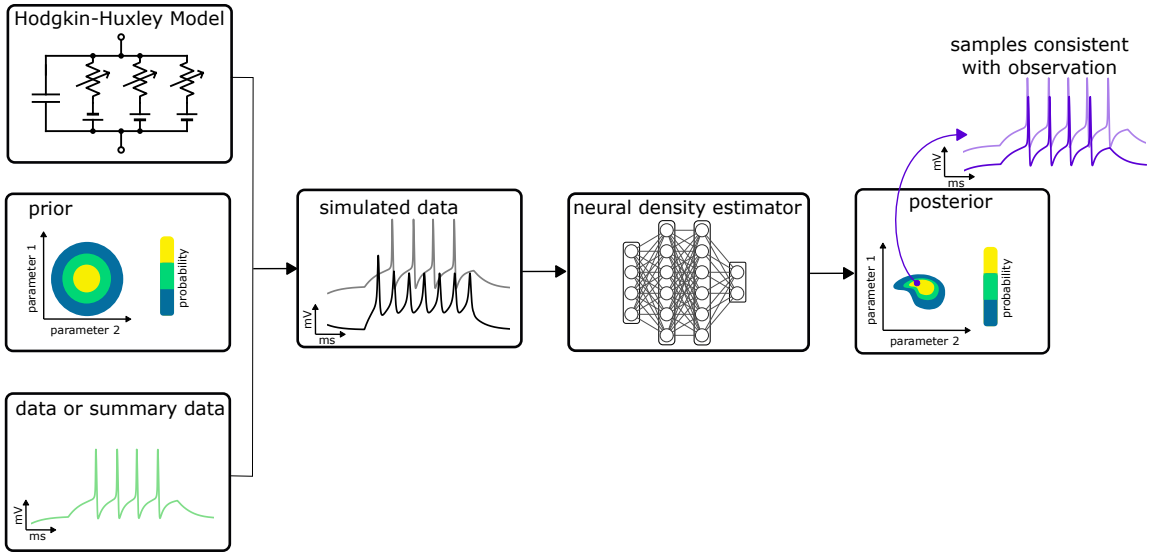

Figure 6.1: Neural posterior estimation, schematic. HH model parameters are sampled from the prior distribution and produce firing patterns by simulating the HH model. Summarizing statistics of simulations, together with corresponding HH model parameters are used to train the neural density estimator, whereas the summarizing statistics of the experimentally observed firing pattern, are fed through the trained network to obtain the posterior. HH model parameters sampled from high posterior regions give rise to HH model simulations consistent with experimental data. Adapted from Gonçalves et al. [78].

6.2 Architecture

6.2.1 Mixture Density Networks

For now we have remained pretty vague on the architecture of our neural density estimator or neural network. A common choice, proven useful on not just synthetic but real-world data as well, is to use *mixture density networks* where the output layer of $q_{\phi(\theta|\mathbf{y})}$ is constituted by the means $\boldsymbol{\mu}_k$, covariances $\boldsymbol{\Sigma}_k$ and weights $\boldsymbol{\pi}_k$ of a *mixture of Gaussians* with k components [7]. During each training step, updated parameters $\boldsymbol{\phi}$ lead then to updated $\boldsymbol{\mu}_k$'s, $\boldsymbol{\Sigma}_k$'s and $\boldsymbol{\pi}_k$. In Gonçalves et al. [78], mixture density networks are used on real-world electrophysiological data to infer HH model parameters, very similarly as discussed here.

7: Oftentimes $k = 10$ or $k = 5$ is used, but this is essentially up to the user.

6.2.2 Normalizing Flows

Recently, however, the use of *normalizing flows* has become increasingly popular. Following notation and train of thought introduced in Papamakarios et al. [83], we consider first a base distribution $p_u(\mathbf{u})$, for instance a multivariate normal or uniform distribution. We can draw samples $\mathbf{u} \sim p_u(\mathbf{u})$, pass them through a transformation T, to obtain transformed random samples $\mathbf{x} = T(\mathbf{u})$. We would hope to construct T in such a way, that producing a histogram of samples $\mathbf{x}$ matches the true target density $p_x(\mathbf{x})$.

T can be flexible, for instance a neural network with parameters $\boldsymbol{\phi}$. Yet, the defining property of flow-based models is that the transformation T must be invertible (i.e. T^{-1} exists and is properly defined), and that both T and T^{-1} are differentiable. In that case, $\mathbf{u}$ and $\mathbf{x}$ have the same dimensionality and the density of p_x can be found under a change of variables:

$$p_x(\mathbf{x}) = \mid \det \mathbf{J}_T(\mathbf{u}) \mid^{-1} p_u(\mathbf{u}), \tag{6.5}$$

where $\mathbf{u} = T^{-1}(\mathbf{x})$, and $\mathbf{J}$ is the Jacobian or matrix collecting first order partial derivatives, that is, matrix element $\mathbf{J}_T(\mathbf{u})_{ij} = \frac{\partial T_i}{\partial u_i}$. Alternatively we could write:

$$p_x(\mathbf{x}) = \mid \det \mathbf{J}_{T^{-1}}(\mathbf{x}) \mid p_u(T^{-1}(\mathbf{x})). \tag{6.6}$$

Essentially, we tune parameters $\boldsymbol{\phi}$ to transform density $p_u(\mathbf{u})$ with T into $p_x(\mathbf{x})$, the distribution of interest. The Jacobian can be thought of measuring the relative change in volume from $\mathbf{du}$ to $\mathbf{dx}$ due to T.

After sampling $\mathbf{u}_0$ from a first base distribution $p_0(\mathbf{u})$, we can in fact, repeatedly transform $\mathbf{u}_1 = T_1(\mathbf{u}_0)$, $\mathbf{u}_2 = T_2(\mathbf{u}_1)$, ..., $\mathbf{u}_K = T_K(\mathbf{u}_{K-1})$ so that $\mathbf{u}_K = \mathbf{x}$ and $T = T_K \circ T_{K-1} \cdots T_2 \circ T_1$. Put differently, we can apply multiple neural networks or transformations, each taking the output of the previous network as input to transform random numbers in sequence and eventually produce random vectors $\mathbf{u}_K$ covering the target distribution. This is where the term *flow* originates form. The fact that they are thought of as *normalizing* is because the inverse $T^{-1} = T_1^{-1} \circ T_2^{-1} \cdots T_{K-1}^{-1} \circ T_K^{-1}$ can be thought of transforming data $\mathbf{x}$ back to $\mathbf{u}$ covering a *base* distribution[8]. Moreover $\det \mathbf{J}_{T_j \circ T_i}(\mathbf{u}_i) = \det \mathbf{J}_{T_j}(T_j(\mathbf{u}_i)) \cdot \det \mathbf{J}_{T_i}(\mathbf{u}_i)$. Normalizing flows

8: In fact, to check whether a normalizing flow is calibrated well one can pass many target samples $\mathbf{x}$ through T^{-1} and verify their histogram is of base distribution form.

are therefore said to *composable*. The resulting flow is still a normalizing flow and one can thus imagine increased flexibility of the transformation if thought to be required for increasingly complex target distributions.

So how does this tie into our original objective Equation 6.1? When we try to obtain the posterior, i.e. distribution over $\boldsymbol{\theta}$, we essentially want our random vectors $\mathbf{u}$ to be transformed with T_ϕ to $\boldsymbol{\theta}$ (previously $\mathbf{x}$). Equation 6.1 then becomes:

$$\mathcal{L}(\boldsymbol{\phi}) = -\sum_n \left(\log(p_u(T^{-1}(\boldsymbol{\theta}^{(n)}), \boldsymbol{\phi})) + \log \mid \det \mathbf{J}_{T^{-1}}(\boldsymbol{\theta}, \boldsymbol{\phi}) \mid_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(n)}} \mid \right). \tag{6.7}$$

It can be shown that Equation 6.7 can be reframed as minimizing a forward $\mathcal{KL}$-divergence, as already discussed in Section 6.1.2.

How can we make this conditioned on electrophysiological feature vectors $\mathbf{y}$? One approach is to concatenate $\mathbf{y}$ to each $\mathbf{u}_i$, at the input of each neural network or transformation T_i. When we evaluate Equation 6.7, then, $T_i = T_i(\mathbf{u}_{i-1}, \mathbf{y})$.

Lastly, the second term in Equation 6.7 can still be computationally expensive to compute, because of the determinant of a potentially high-dimensional matrix. Autoregressive flows, however, can make that computation much faster. *Autoregressive* stems from the fact that these flows are implemented so that $\boldsymbol{\theta}_i = \boldsymbol{\theta}_i(\mathbf{u}_{<i})$, so that for instance the 4-th element in vector $\boldsymbol{\theta}$ will only depend on the 1st, 2nd and 3rd element in vector $\mathbf{u}$. To achieve this, one needs to *mask* certain weights in the neural network or flow T_ϕ. That brings us to the class of normalizing flows used throughout this thesis when we say we use neural posterior estimation, namely *masked autoregressive flows* [84].

[84]: Papamakarios et al. (2017), 'Masked autoregressive flow for density estimation'

6.3 Related Work

As one might guess, the type of flow or neural density estimator can be chosen out of potentially an infinite set of possibilities. Many other interesting flexible flows are discussed in Papamakarios et al. [83]. In practice, it is a priori often not obvious what flow is best or sufficiently flexible for one's experimental data set. Note that with the simulation-based inference package https://www.mackelab.org/sbi/, one can flexibly choose the architecture including mixture density networks and a variety of flow architectures to do inference. Inference results in this thesis are obtained with this package.

One might opt to train a *neural* network to *estimate* the *likelihood* rather than the posterior directly, aptly termed Neural Likelihood Estimation (NLE) [85, 86]. This could be interesting if it is assumed for instance that simulations are considered highly informative in constraining θ, so that one would not need bias in the form of prior $p(\boldsymbol{\theta})$. Depending on the dimensionality of, in this case, $\boldsymbol{\theta}$ or $\mathbf{y}$ one or the other setting might be more computationally expensive to train the flow. Note, however, that if one wishes to obtain posterior samples $\boldsymbol{\theta}_i$ with NLE, that one needs to resort to sampling methods including for instance Markov Chain Monte Carlo (MCMC) approaches, which can be computationally more exhaustive.

[85]: Lueckmann et al. (2018), 'Likelihood-free inference with emulator networks'
[86]: Papamakarios et al. (2019), 'Sequential Neural Likelihood: Fast Likelihood-free Inference with Autoregressive Flows'

When we use mixture density networks (Section 6.2.1) to estimate the likelihood $q_\phi(\mathbf{y} \mid \boldsymbol{\theta})$ based on simulated training data $(\boldsymbol{\theta}, \mathbf{y})^{(n)}$, we can analytically compute marginal likelihoods as we are working with Gaussians. We can thus easily 'average away' certain electrophysiological features and calculate how much the posterior uncertainty changes. The distribution might for instance become much 'broader' (uncertain) in the neighborhood of the maximum-a-posterior estimate. Consequently, we can quickly figure out which electrophysiological feature are important in constraining our estimates for the HH model parameters, without having to train a new neural density network from scratch. This has been the focus of Beck et al. [20], of which we will discuss its application to this data set in Chapter 7.

As we are equipped with full distributions over HH model parameters, it is possible to construct posterior *paths*. These paths serve as high probability subspaces in high-dimensional HH model parameter space, revealing multiple parameter combinations that can produce simulations recovering the experimental observation. In contrast, moving orthogonally to said paths, quickly reveals model parameter combinations for which the produced simulation fails to recover the observation. Details on how to construct such paths can found in Gonçalves et al. [78]. The degeneracy of biophysical models has been extensively studied in the pyloric rhythm of the crustacean stomatogastric ganglion for instance [67, 87, 88], but so-called compensation mechanisms are well supported in biological systems too [89–91]. Altering the expression of one channel can for instance induce changes in the expression of others to keep certain function and cell viability [92]. Compensation mechanisms can also be relevant in the presence of energy constraints [67]. Full distributions obtained with NPE provide for a natural way to study these mechanisms: one needs simply to follow the *path* of highly likely parameter combinations. Implied by high relevance in biology, further applications of this methodology on experimental data is highly warranted.

[67]: Deistler et al. (2022), 'Energy-efficient network activity from disparate circuit parameters'

[87]: Marder et al. (2007), 'Understanding Circuit Dynamics Using the Stomatogastric Nervous System of Lobsters and Crabs'

[88]: Prinz et al. (2004), 'Similar network activity from disparate circuit parameters'

[89]: Turrigiano (1999), 'Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same'

[90]: Spitzer (1999), 'New dimensions of neuronal plasticity'

[91]: Galante et al. (2001), 'Homeostatic plasticity induced by chronic block of AMPA/kainate receptors modulates the generation of rhythmic bursting in rat spinal cord organotypic cultures'

[92]: MacLean et al. (2003), 'Activity-Independent Homeostasis in Rhythmically Active Neurons'

[67]: Deistler et al. (2022), 'Energy-efficient network activity from disparate circuit parameters'

6.4 Discussion

To conclude, when we infer uncertainties about HH model parameters conditioned on electrophysiological measurements, we

- sample $\boldsymbol{\theta}^{(n)} \sim p(\boldsymbol{\theta})$, where n ≈ 15 million,
- simulate HH models to obtain corresponding $\mathbf{y}^{(n)}$,
- keep well-defined simulations, deleting ones with undefined summarizing statistics, reducing n to about half or ≈ 7 million,
- train a masked autoregressive flow T_ϕ by minimizing objective Equation 6.7.

Our simulation-based trained flow T_{ϕ^*} can then be utilized to estimate uncertainties about HH model parameters, given a true observation $\mathbf{y}^{(o)}$, as discussed in Section 6.1.3. The only thing we need to do is sample enough[9] random vectors from our base distribution $\mathbf{u}^{(i)} \sim p_u$ and feedforward through the trained flow, conditioned on the observation to obtain $\boldsymbol{\theta}_i = T_{\phi^*}(\mathbf{u}^{(n)}; \mathbf{y}^{(o)})$. Finally, we can evaluate the probability of the outputted HH model parameter combinations $\boldsymbol{\theta}^{(n)}$ by plugging it into Equation 6.5.

9: For practical purposes, between 10 000 and 100 000 samples is usually considered sufficient, but this may depend on the dimensionality of either $\mathbf{y}$ or $\boldsymbol{\theta}$.

Unfortunately, applying simulation-based inference or neural posterior estimation on real-world data is not as straightforward as it seems. Whereas flexible neural density estimators such as masked autoregressive flows can be applied successfully on synthetic data considered as 'true' observations [78, 84], they might fail considerably on real-word observations. Despite their flexibility, the simulation-based trained flow easily fails to generalize to experimental data, rendering unreliable posterior estimates. This will be the topic of Chapter 7.

Bridging Simulation and Experimental Viewpoints in Neuroscience

Neural Posterior Estimation with Noise

7

The Hodgkin-Huxley-based model discussed in Chapter 5 is capable of accurately capturing firing patterns observed from experiments with Patch-seq, both qualitatively and quantitatively. In fact, multiple parameter combinations can oftentimes produce model outputs that are satisfyingly close to experimental recordings according to the electrophysiologist's judgment.

In Chapter 5, we discussed neural posterior estimation, a machine-learning based framework, to obtain full probability measures over the entire biological plausible model parameter space. With it, we understand which model parameter combinations are likely, and which less likely, to fit observed electrophysiology.

As we will see next in Section 7.1, however, obtaining reliable probability distributions for experimental data including Patch-seq can be cumbersome, which we attribute to a small but consistent mismatch between the model and the data.

Section 7.2 discusses a working strategy to overcome problems with neural posterior estimation due to mismatches between model and data. We introduce noise to the summarizing statistics of simulations used to train the deep neural density estimator and show how the obtained posteriors, the full probability distributions of model parameters given observed experimental features, give much more reliable estimates.

This chapter discusses the work of **Bernaerts** et al. [18], one of the main papers included in this thesis, and the reader is referred to this work for a more extensive discussion.

7.1 Neural Posterior Estimation

7.1.1 Setting

In Section 5.2.3, we have seen that the HH model is capable of producing firing rates that capture qualitatively what is experimentally observed, as well as matching quantitatively derived electrophysiological features including the *firing rate*, *AP width*, and so on. We would like to move a step further. We would want to know how probable each possible parameter combination[1] is in producing a Hodgkin-Huxley-based model simulation sufficiently close the experimental observation of interest, qualitatively and quantitatively.

1: Defined in a biologically plausible prior space.

In Chapter 5 we therefore introduced Neural Posterior Estimation (NPE), a framework that naturally lends itself to this setting. To recapitulate, given experimentally observed electrophysiology vector $y^{(o)}$ we are interested in obtaining a reliable posterior distribution $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$. That is, we have trained a normalizing flow q with parameters $\boldsymbol{\phi}$ based on a batch of HH model simulations, and hope the output to be a reliable

distribution when the input is, in this case, an experimental observation (see Section 6.1.3).

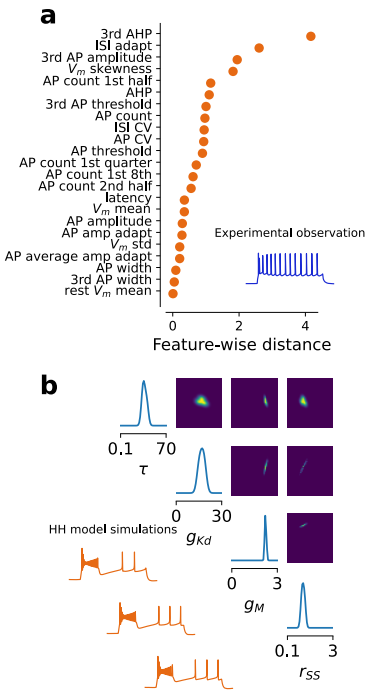

Figure 7.1: Neural posterior estimation without noise applied to an experimental observation leads to unreliable posterior estimates. (a) We calculate the difference of every electrophysiological feature value derived from the experimental observation shown in the inset, with the ones derived from the HH model simulation produced with the maximum-a-posteriori estimate. Differences are shown after Z-scoring. 0 would denote a perfect fit for that electrophysiological feature. **(b)** One-dimensional (diagonal) and two-dimensional marginals (upper diagonal) of $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$ trained with NPE (4 out of 13 model parameters are shown). Three HH model simulations are shown corresponding to three model parameter combinations with highest probability out of 10 000 samples.

7.1.2 Problem

Posterior estimates derived with neural posterior estimation often lead to unreliable distributions. Unfortunately, when we feed an experimental observation shown here in Figure 7.1 a, our trained neural density estimator — our trained masked autoregressive flow — can produce unreliable model parameter distributions. Here, we select three model parameter combinations with highest probability out of 10 000 randomly sampled model parameter combinations $\boldsymbol{\theta}^{(n)} \sim q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$, and plot their HH model simulations in the lower left corner of Figure 7.1 b. Clearly, simulations do not qualitatively match the experimental observation, and as we can deduce from Figure 7.1 a, especially features including the *ISI adapt* and *3rd AHP* are not fitted well.

7.1.3 Diagnostics

When we feedforward a HH model simulation's electrophysiological feature vector $\mathbf{y}^{(i)}$, the trained neural density estimator tends to produce much more reliable posterior estimates $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(i)})$. In Figure 7.3 a we show a HH model simulation, considered here as the 'experimental observation'. In this setting, the three model parameter combinations with highest probability out of 10 000 random samples from the posterior estimate $\boldsymbol{\theta}^{(n)} \sim q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(i)})$ produce HH model simulations that are qualitatively very similar to the 'observation' (Figure 7.3 b, orange). Moreover, the maximum-a-posteriori (MAP) estimate generates a HH model simulation that fits all electrophysiological features very well (Figure 7.3) a), except for the *ISI CV* (Section 5.2.2 a). The difficulty in fitting *ISI CV* could be due to a lack of simulations that approach the exact value for the *ISI CV* of this randomly picked simulation considered here as the 'observation'. Alternatively, the neural network might appreciate some electrophysiological features more than others to predict HH model parameters and consequently 'weighs' them differently.

The fact that NPE seems to work on simulated true data but not experimental data begs the question: how different are experimental observations generally from HH model simulations? Do they occupy a very different electrophysiological feature subspace than simulations do? Put differently, is there a systematic mismatch between the data and the model (Figure 7.2 a)?

To show a potential data-model mismatch, we select a simulation from the synthetic data set of 15 million simulations and rank ordered all other simulations by their distance in the feature space to this reference (Figure 7.2, top). While few simulations are very close to the reference, most lie somewhat farther away, but still produce qualitatively very similar outcomes (Figure 7.2 b, orange). In contrast, if we choose an experimental firing pattern as reference, the distance to the closest simulation in the electrophysiological feature space is larger than between any simulation from the synthetic data set (Figure 7.2b, blue). Consequently, simulated

traces are much more dissimilar to the experimental observations than to the simulated references.

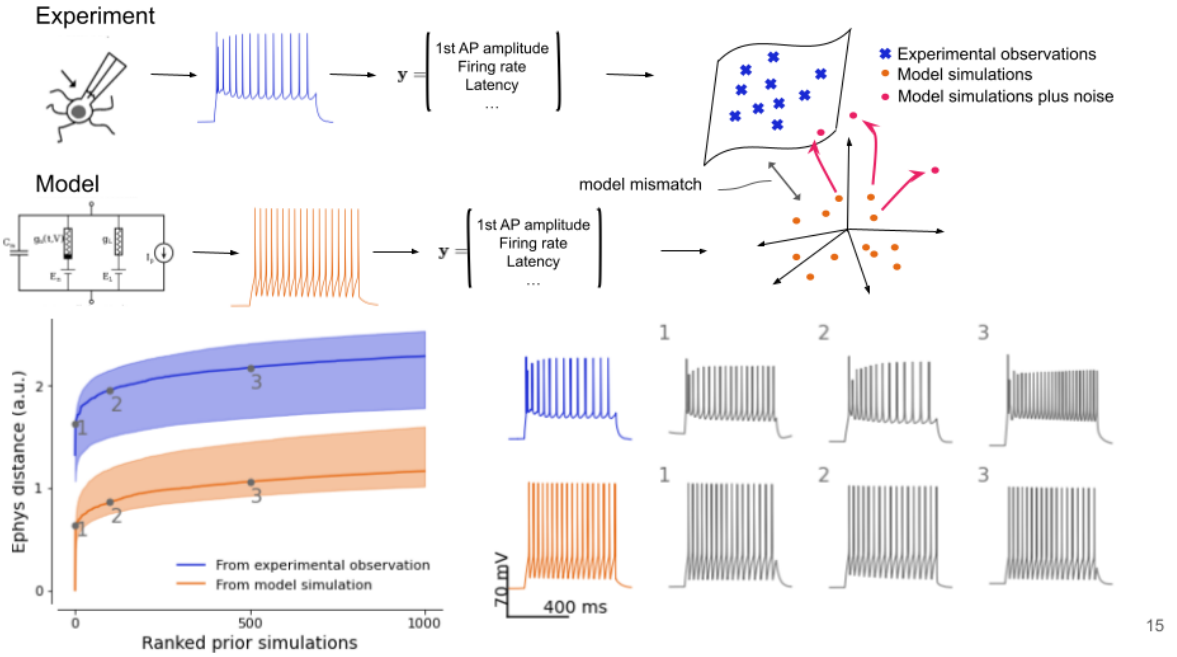

Figure 7.2: Neural posterior estimation, diagnostics. (Top) Sketch illustrating a data-model mismatch: in electrophysiological feature space, simulations do not cover the space of experimental observations. **(Bottom)** Simulations are further away from experimental observations (blue) than from other simulations (orange). Qualitatively, simulations increasingly further away from an experimental observation look more dissimilar than from another simulation. Numbers 1,2 and 3 refer to the 2nd, 101st and 501st closest simulations respectively.

7.2 Neural Posterior Estimation with Noise

7.2.1 Out-of-distribution Test Data.

The problem we are faced with is a long-standing and common issue in *machine learning*. Our trained neural network — our trained masked autoregressive flow q_{ϕ^*} — fails to generalize to *test* data, in this case experimentally observed neuronal firing patterns, as it has only seen *training* data made up of HH model simulations. Our test data is out of distribution, that is, it is not adequately captured by the distribution covering training data with which the neural density estimator was trained. Remember that we can only train the neural density estimator with simulations for which there exist one-to-one (up to some current noise, see Section 5.2.2 and Equation 5.4) tuples $(\boldsymbol{\theta}, \mathbf{y})^{(n)}$ and that we are not equipped with model parameters for experimental observations $\mathbf{y}^{(o)}$.

7.2.2 Adding Noise.

Introducing noise to the summary statistics of training data benefices the generalizing capabilities of neural density estimators including normalizing flows. As a (training) data manipulation strategy, one could shift the

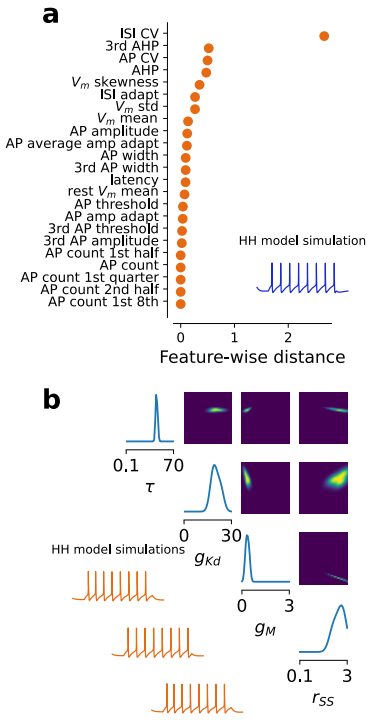

Figure 7.3: NPE without noise applied to HH model simulations leads to reliable posterior estimates. (a,b) Analogous to Figure 7.1 but when a HH model simulation is feedforwarded through the trained neural density estimator $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(i)})$, where $\mathbf{y}^{(i)}$ is derived from a HH model simulation.

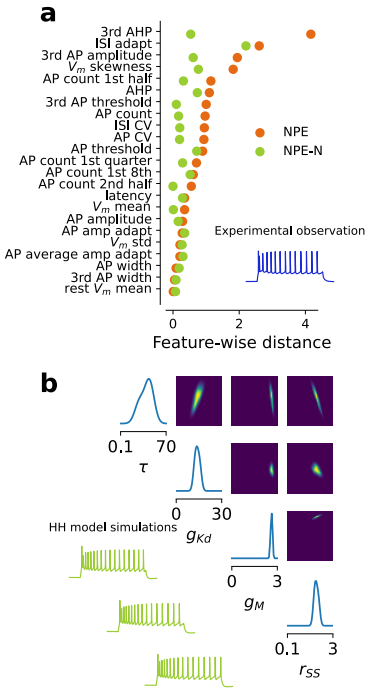

Figure 7.4: Neural posterior estimation with noise, applied to an experimental observation, leads to reliable posterior estimates. (a) Analogous to Figure 7.1 a, but with additional differences to electrophysiological features values derived with the HH model simulation set up with maximum-a-posteriori estimate from using NPE *with* noise (NPE-N). **(b)** One-dimensional (diagonal) and two-dimensional marginals (upper diagonal) of $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$ trained with NPE-N. Three HH model simulations are shown corresponding to three model parameter combinations with highest probability out of 10 000 samples.

subspace occupied by simulations by adding small amounts of isotropic Gaussian noise (Figure 7.2, top, red arrows).

Intuitively, we can see how this can be helpful: even though the coordinates of our training data samples (our HH model simulations) will be shifted randomly in electrophysiological feature space, many simulations will get closer to the experimental observation manifold, by chance. Mathematically, instead of training our neural density network q_ϕ with tuples $(\boldsymbol{\theta}, \mathbf{y})^{(n)}$ (Section 6.1.2), we train it with $(\boldsymbol{\theta}, \mathbf{y} + \boldsymbol{\varepsilon})^{(n)}$, where $\boldsymbol{\varepsilon}^{(n)} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and Σ is a diagonal matrix occupied with standard deviations of the training data for each electrophysiological feature (Table 5.2). Our neural density network is therefore expected to *see* more data close to the experimental manifold, closing the model-data gap and resolving to some extent the distributional shift, and is therefore expected to generalize better. We term this strategy NPE-N for neural posterior estimation with noise.

The skeptical reader might note that this changes the model itself significantly. It might seem that by post-hoc introducing noise to the summary statistics of HH model simulations, we change the deterministic (up to some current noise described in Equation 5.4) mapping from model parameters to electrophysiological features, making it much more stochastic. To put this in perspective, we need to remember our original goal, which is for the trained neural density estimator to give posterior estimates $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$ that are reliable. More specifically, we want samples from that distribution $\boldsymbol{\theta}^{(n)} \sim q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$ to produce HH model simulations for which the summarizing statistics $\mathbf{y}^{(n)}$ fit $\mathbf{y}^{(o)}$ (*without* the post-hoc introduction of noise), and for which the firing pattern qualitatively matches the observed electrophysiology in the cell with Patch-seq. We thus only manipulate *simulated* training data, as an attempt to obtain more reliable posterior estimates when we feedforward an *experimental* observation. We then wish to get posterior samples that produce satisfying firing patterns, those observed with Patch-seq, and using the model explicitly described in Equation 5.4.

So let us return to our experimental observation shown in Figure 7.1. When we feedforward $\mathbf{y}^{(o)}$ through the neural density estimator that was trained with the manipulated training data set, we obtain a different posterior estimate $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$. We observe that 1-dimensional marginal distributions for τ and r_{SS} are broader and shifted (in the mode), respectively (Figure 7.4 b). This posterior estimate is also much more reliable, as HH model simulations produced by samples $\boldsymbol{\theta} \sim q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$ with high posterior weight qualitatively match the firing pattern and summarizing statistics $\mathbf{y}^{(o)}$ of the neuron much better (Figure 7.4 b and a, respectively).

7.2.3 Examples

To further showcase the strength of our adapted training strategy, we select cells representing *Pvalb, Sst, Vip* and *Lamp5* interneuron families besides one belonging to the excitatory *Pyramidal* class in Figure 7.5, Figure 7.6, Figure 7.7, Figure 7.9 and Figure 7.8 respectively.

In some cases, both NPE and NPE-N produce reliable model parameter distributions as samples produce firing patterns (Figure 7.5 b,c and Figure 7.6 b,c, orange for NPE and yellowgreen for NPE-N) qualitatively similar to the experimental observation (Figure 7.5 a and Figure 7.6 a, blue). Yet, the maximum-a-posteriori estimate of posterior distribution generated with NPE-N produces a HH model simulation with summarizing statistics much closer to the experimental observation than with NPE (Figure 7.5 a and Figure 7.6 a). Especially features including the *AHP*, *AP amplitude* and *AP threshold* (see Table 5.2) are fitted much better with NPE-N.

For a more complex firing pattern observed in the *Vip Serpinf1_1* interneuronal family and showcased in Figure 7.7 a (blue), however, the difference between NPE and NPE-N is more obvious: HH model simulations generated with samples from NPE-N posterior estimates (Figure 7.7 c, yellowgreen) capture much more electrophysiological complexity such as changes in AP frequency during the current injection time window as well as action potential shapes of the 1st and 3rd generated AP (Figure 7.7 a), in comparison to HH model simulations generated with samples from NPE posterior estimates (Figure 7.7 b, orange). NPE-N seemingly produces more conservative posterior estimates, as both 1-dimensional and 2-dimensional marginals are generally broader in comparison to NPE (Figure 7.7 c,b respectively). A very similar story can be written for the *Lamp5 Egln3_1* interneuron showcased in Figure 7.9. In this case, samples from the NPE posterior estimate fail to produce simulations with APs after a certain point in time (Figure 7.9 b), even though they are experimentally observed.

For the Pyramidal cell showcased in Figure 7.9, it is less clear whether NPE-N outperforms NPE. Most electrophysiological features derived from the HH model simulation generated with the MAP estimate are fitted as well as or better with NPE-N, except the *AP amplitude* and *ISI adapt* (Figure 7.8 a). Arguably, we would want the posterior estimate to generate samples that produce simulations with an initial small burst of APs, followed by steady firing of APs as is experimentally observed. Samples from the NPE-N posterior estimate capture that transition qualitatively better, but produce overall too many APs, whereas samples from the NPE posterior estimate produce simulations with no clear transition and too few APs (Figure 7.8 b,c).

Finally, we show an *Sst Th_1* interneuron in Figure 7.10 as both NPE and NPE-N here fail to provide for posterior distributions from which samples can generate HH model simulations that are satisfyingly similar to the experimental observation. HH model simulations from both NPE and NPE-N posterior estimates fail to stop firing APs at the appropriate point in time, and fail to decrease the amplitude of every subsequently generated AP appropriately. We therefore stumble upon a possible limitation of the model itself described by Equation 5.4 in Equation 5.4. It could be that we need a more flexible model, either in the form of more compartments or more model parameters (ion channels) to explain the observed electrophysiology in this neuron.

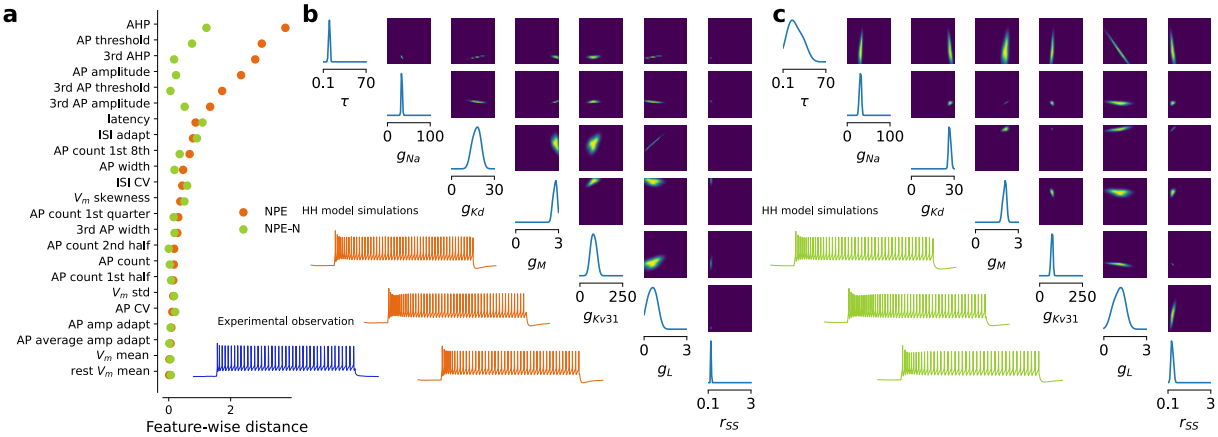

Figure 7.5: NPE vs NPE-N, illustration 1: fast-spiking *Pvalb Calb1_1* interneuron. (a) Analogous to Figure 7.4 a. **(b)** Analogous to Figure 7.1 b, but 7 model parameters are shown. **(c)** Analogous to Figure 7.4 b, but 7 model parameters are shown.

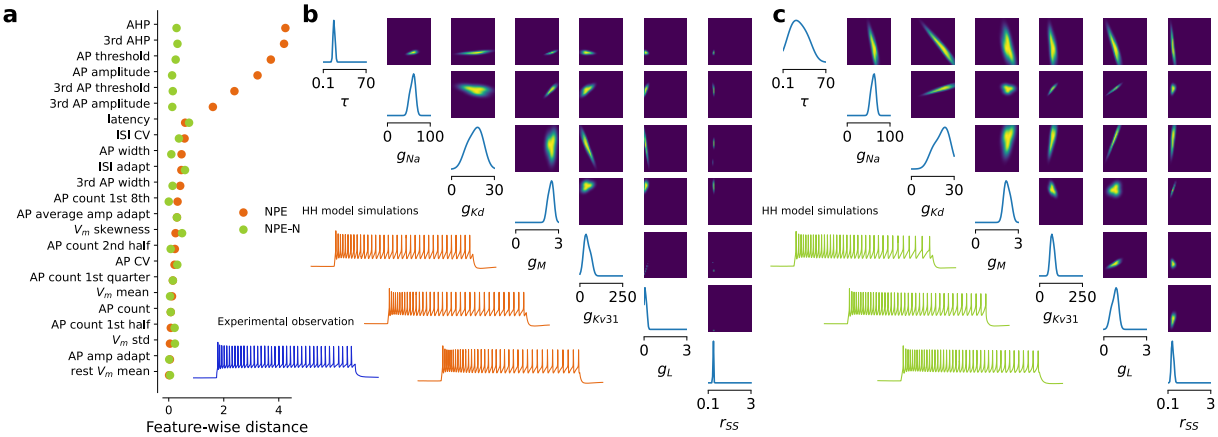

Figure 7.6: NPE vs NPE-N, illustration 2: *Sst Crhr2_1* interneuron. (a,b,c) Analogous to Figure 7.5.

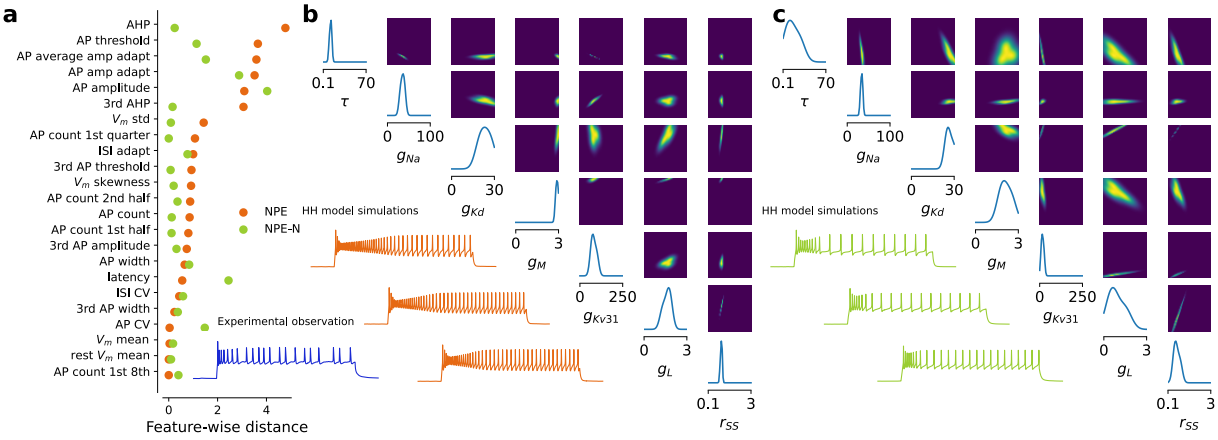

Figure 7.7: NPE vs NPE-N, illustration 3: *Vip Serpinf1_1* interneuron. (a,b,c) Analogous to Figure 7.5.

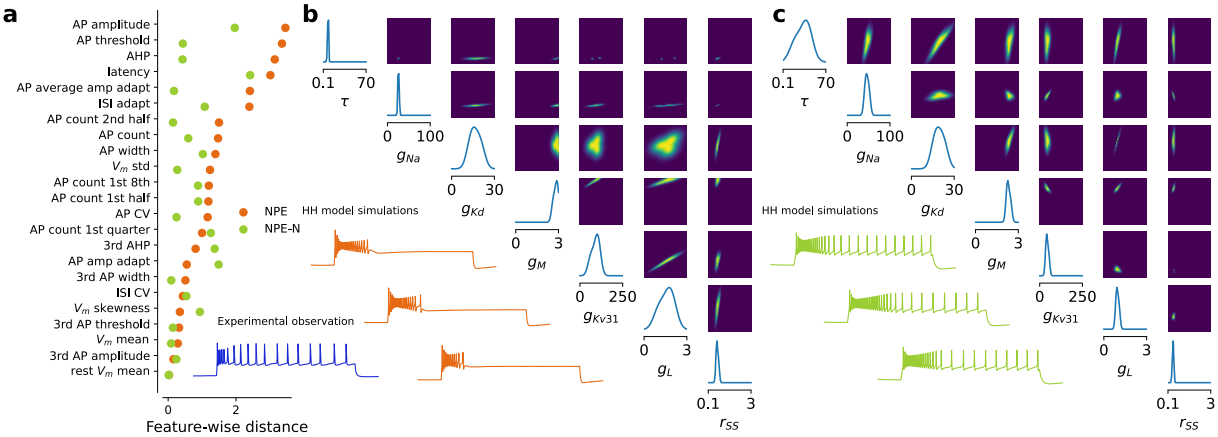

Figure 7.8: NPE vs NPE-N, illustration 4: *Lamp5 Egln3_1* interneuron. (a,b,c) Analogous to Figure 7.5.

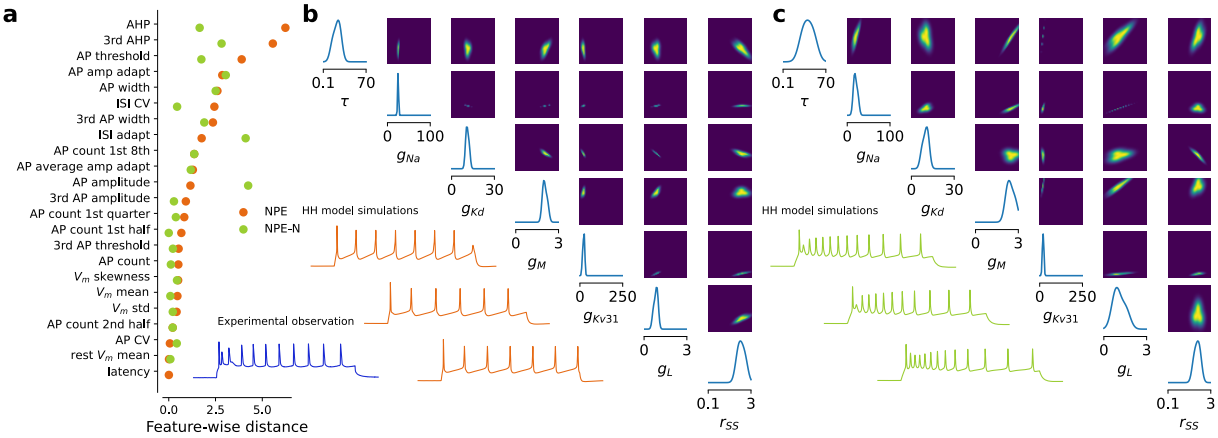

Figure 7.9: NPE vs NPE-N, illustration 5: *L6 CT Cpa6* Pyramidal cell. (a,b,c) Analogous to Figure 7.5.

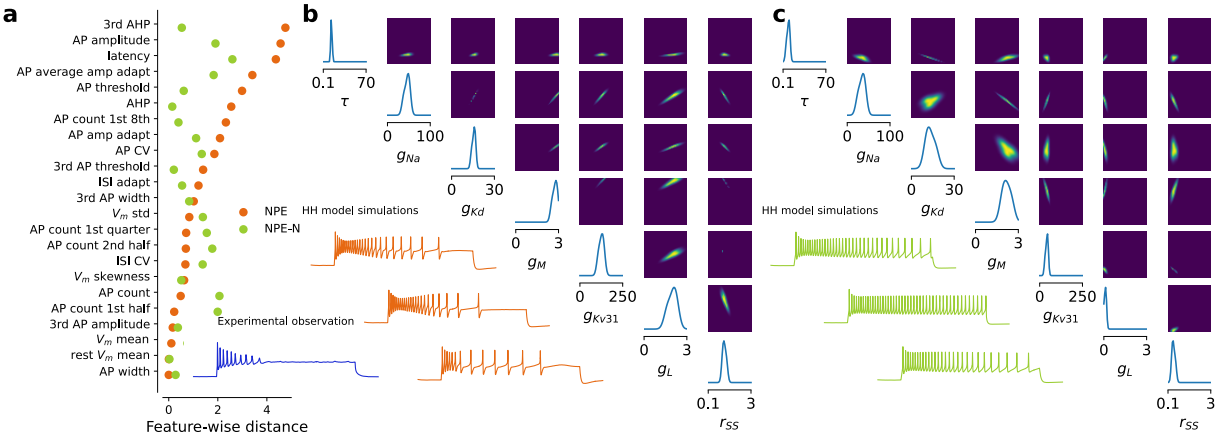

Figure 7.10: NPE vs NPE-N, illustration 6: *Sst Th_1* interneuron. (a,b,c) Analogous to Figure 7.5.

7.2.4 NPE-N applied to Patch-seq Data Set.

In Section 7.2.3 we discussed six specific cells representing different subclasses in the whole data set of $n = 955$ MoP neurons. In this section, we seek to obtain a picture that describes the general performance of NPE-N on this data set.

In Figure 7.11 a, we show a t-SNE embedding of $n = 955$ MoP neurons. Interneuronal *Pvalb* (red), *Sst* (yellowish), *Vip* (purple) and *Lamp5* (pink) cells as well as excitatory Pyramidal *Pyr* cells are well separated. We can confirm the cell's identity by overlaying marker gene expressions (Figure 7.11 b).

As we are equipped with density estimator $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y})$ trained with adapted training schedule discussed with NPE-N in Section 7.2.2, we can feed every experimental observation $\mathbf{y}^{(o)}$ and obtain $n = 955$ posterior estimates $q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$. We can obtain the MAP parameter estimate for each posterior estimate and overlay the same embedding with its parameter values for each cell (Figure 7.11 d. We can for instance immediately deduce that high r_{SS} parameter values are needed to fit the firing patterns of *Pyr* cells (Figure 7.11 d). Indeed, as discussed previously, r_{SS} is in fact introduced to fit the larger *AP widths* observed in *Pyr* cells (Figure 7.11 e,f). Potassium conductance g_{Kd} on the other hand proves important to fit firing patterns of fast-spiking *Pvalb* cells, which need rapid repolarization of the membrane potential after action potential generation in order to sustain high firing rates (Figure 7.11 e,f). Embeddings overlayed with summarizing statistics derived from HH model simulations generated with MAP estimates, shown in Figure 7.11 e, are very similar to embeddings overlayed with summarizing statistics derived from experimental observations, shown in Figure 7.11 f. This further demonstrates the applicability of using NPE-N to real-world observations such as obtained with Patch-seq.

NPE-N can be used to gain insight about uncertainties about MAP estimates. If we are only interested in unique parameter estimates that can generate HH model simulations each satisfyingly matching their respective $\mathbf{y}^{(o)}$, we can equally as well minimize a mean-squared-error in electrophysiological feature space, as conducted in Section 5.2.3. This point estimate is obtained much faster because it does not involve training a deep neural network. We are, however, interested in probability distributions over the full parameter space, revealing not only the most likely model parameter combinations but the very unlikely as well. Additionally, if we want to understand how certain NPE-N is about its MAP estimate for each observation, we can calculate the *posterior entropy* which is a measure for the broadness of the posterior. Broader posteriors would indicate higher uncertainty about the MAP estimate in order to reproduce the firing rate experimentally observed. Vice versa, narrower posteriors denote lower uncertainty about the MAP estimate. Posterior entropy can be calculated as $\sum_{k=1}^{K} -\log q_{\phi^*}(\boldsymbol{\theta}^{(k)} \mid \mathbf{y}^{(o)})$, where we sample $\boldsymbol{\theta}^{(k)} \sim q_{\phi^*}(\boldsymbol{\theta} \mid \mathbf{y}^{(o)})$ and choose $K = 1000$. We thus obtain $n = 955$ uncertainty estimates and overlay those on the same embedding, as shown in Figure 7.11 c. *Vip* neurons, which are relatively sparsely sampled in the data set, show higher posterior uncertainty. In contrast, *Pvalb* neurons show lowest uncertainty indicating that their posteriors are best constrained. One reason for this may be that *Pvalb* neurons fire

APs more stereotypically, whereas *Vip* neurons show greater variability in their firing patterns [11, 93], that may require greater flexibility in the model to reproduce all summarizing statistics.

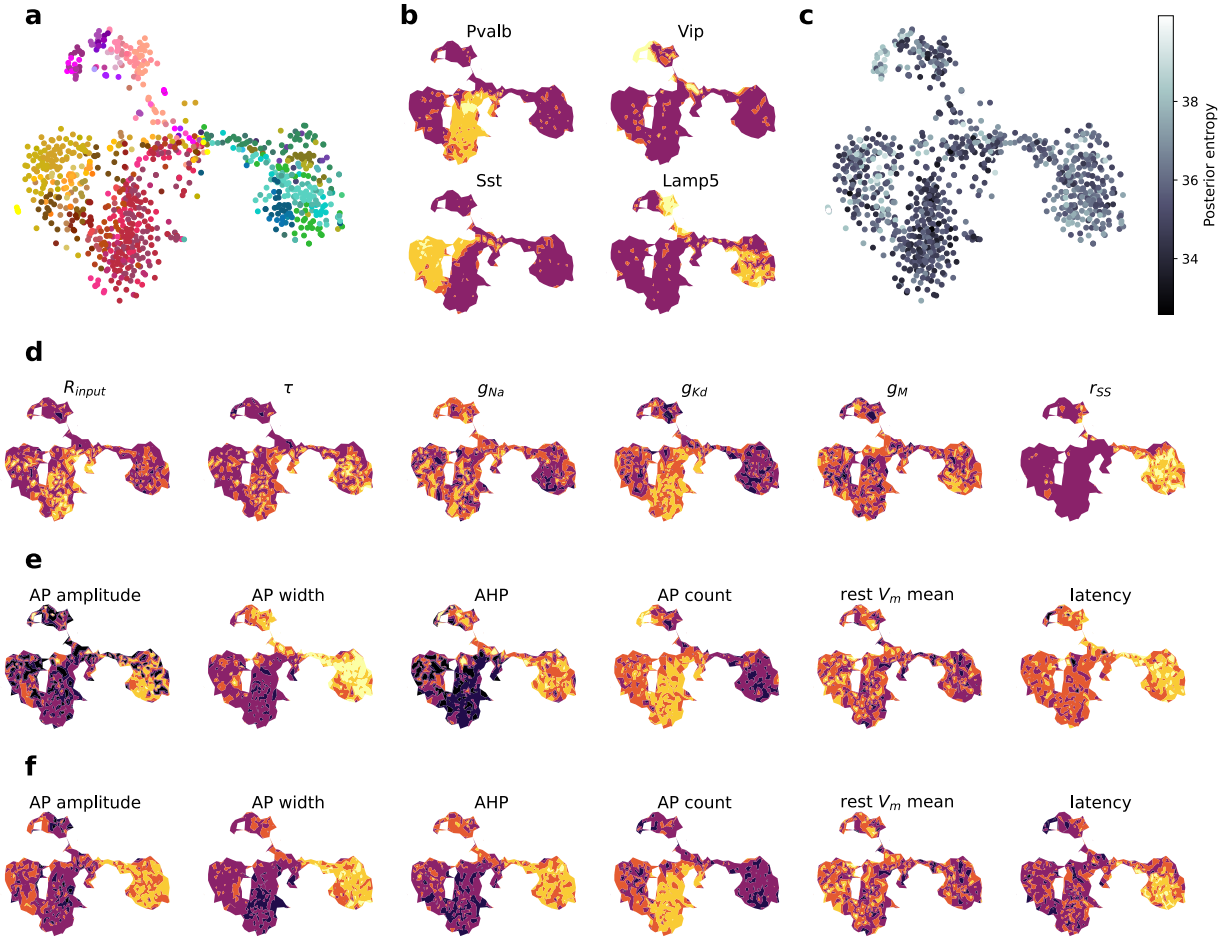

Figure 7.11: NPE-N applied to the electrophysiology of $n = 955$ neurons obtained with Patch-seq. Two-dimensional embedding reveals difference in HH model parameters between neural families. (a) T-sne embedding of $n = 955$ Mop neurons based on transcriptomic data. Colors denote transcriptomic types again (see for instance Figure 2.2). Cells in the middle of the embedding tend to show lower quality transcriptomic data and therefore group together. **(b)** Marker genes expression levels overlayed and interpolated on embedding confirm known subclasses (dark purple: low expression, yellow: high). **(c)** Uncertainty of MAP parameters for each cell overlayed on the embedding. **(d)** Selection of MAP parameters, overlayed on embedding. **(e)** Selection of summary statistics derived from simulations corresponding to MAP estimates, overlayed on embedding. **(f)** Selection of summary statistics describing observed electrophysiology, overlayed on embedding.

7.2.5 Discussion

The simulation gap manifests itself as what in the machine learning community is more well-known as the 'distribution shift'. Here, training and test data come from different distributions leading to poor generalizing capability of for instance convolutional neural networks in image classification and graph neural networks [94–96]. In this work, the shift is caused by the gap between summary statistic values derived from model simulations (training data) with respect to Patch-seq experiments (test data). As there is ongoing research to alleviate problems ascribed to this 'shift', we remain optimistic that advances in neural network architectures will speed up their generalizing capability, perhaps making training data manipulations such as the addition of noise, eventually redundant.

[11]: Scala et al. (2021), 'Phenotypic variation of transcriptomic cell types in mouse motor cortex'
[93]: Apicella et al. (2022), 'VIP-Expressing GABAergic Neurons: Disinhibitory vs. Inhibitory Motif and Its Role in Communication Across Neocortical Areas'
[94]: Taori et al. (2020), 'Measuring Robustness to Natural Distribution Shifts in Image Classification'
[95]: Wu et al. (2022), 'Handling Distribution Shifts on Graphs: An Invariance Perspective'
[96]: Quiñonero-Candela et al. (2022), *Dataset Shift in Machine Learning*

As a couple of ongoing strategies, in parallel work [97–100], robust neural posterior estimation[97] takes the opposite approach to the strategy outlined in Section 7.2.2 and denoises the measured data towards the model using Monte-Carlo sampling. On the other hand, robust synthetic likelihood approaches [101] that estimate likelihoods rather than posteriors, work similarly to our approach. Which strategy works best for which models and circumstances remains to be evaluated, but these strategies will allow to apply SBI in cases where models provide relatively coarse, but useful approximations of the true phenomena. Alternatively, one could make the model more realistic. In our case, some of the model mismatch is likely also caused by the use of single-compartment models in contrast to other studies, which use HH models with two or more compartments [71, 102, 103]. Adding more compartments and consequently more parameters to the model, however, increases the time needed to run simulations and the amount of simulated data needed to train neural density estimators, and for which especially the former can turn out to be prohibitive on most computing infrastructure available today.

Equipped, not with mere point estimates but distributions to our disposal, we know which model parameter combinations are likely (and less likely) to explain observed electrophysiology, a clear Bayesian advantage over 'evolutionary fitting algorithms' as for instance employed in related work by Gouwens et al. [102]. Indeed, as the authors point out themselves, multiple parameter combinations can explain same electrophysiology which genetic algorithms may or may not capture. Interestingly, the authors also experienced issues fitting *AP widths* of all cells in their data set. Interestingly, where we opted to introduce parameter r_{SS} directly influencing the gating mechanics in one ion channel, they opted for a different combination of ion channels tailored to the exact width of the AP fired by the cell.

In Section 6.3, we discussed that when we obtain the likelihood with mixture density networks, rather than the posterior with a normalizing flow, we can analytically marginalize out electrophysiological features, and thus obtain estimates with a reduced set of summarizing statistics, quickly. For some electrophysiological recordings, we can show that posterior estimates can be obtained with NLE, with a similar data manipulation strategy, close enough to the ones obtained with NPE-N[18]. We can thus employ the methodology introduced in Beck et al. [20] to investigate the relative importance of electrophysiological features in constraining the posterior condition on *true* electrophysiological data. On average, the *Rest* V_m *mean* turns out the single most important feature, followed by the V_m *mean*, *AP amplitude*, the *AP threshold* and V_m *std* [18] (Table 5.2).

We will expand on fitted parameter values across cell classes and cell types together with their relation to gene expression levels in Chapter 8.

[97]: Ward et al. (2022), 'Robust Neural Posterior Estimation and Statistical Model Criticism'
[98]: Cannon et al. (2022), *Investigating the Impact of Model Misspecification in Neural Simulation-based Inference*
[99]: Schmitt et al. (2021), *Detecting Model Misspecification in Amortized Bayesian Inference with Neural Networks*
[100]: Frazier et al. (2021), *Synthetic Likelihood in Misspecified Models: Consequences and Corrections*
[101]: Frazier et al. (2021), 'Robust Approximate Bayesian Inference With Synthetic Likelihood'
[71]: Hay et al. (2011), 'Models of Neocortical Layer 5b Pyramidal Cells Capturing a Wide Range of Dendritic and Perisomatic Active Properties.'
[102]: Gouwens et al. (2018), 'Systematic generation of biophysically detailed models for diverse cortical neuron types'
[103]: Nandi et al. (2022), 'Single-neuron models linking electrophysiology, morphology, and transcriptomics across cortical cell types'

Predict Fitted Model Parameters from Genes

8

The field of neuroscience is truly of interdisciplinary nature. Great care in experimental design leads to new experimental observations that can update biophysical models, both on the equation level, as well as on the parameter level. Updated biophysical models can give rise to new biological insights, new hypotheses to be tested, and thus further guide experimental design or even entire new protocols (Chapter 5).

We have seen that Patch-seq paved the way for neuroscientists to unravel the relationship between genetic and phenotypic identity in the *same* set of neurons, whereas in earlier days, most experiments done on a set of cells, led to unidimensional information, be it on a genetic or phenotypical level. Data scientists, therefore, now have the responsibility to extract as much knowledge from these low-throughput but multidimensional and highly informative techniques.

We discussed sparse reduced-rank regression in Chapter 3, a linear statistical tool that is easy to understand and that comes with great biological interpretability: it selects genes best capable of predicting electrophysiological variability, and produces interpretable two-dimensional embeddings. In Chapter 4, we discussed its nonlinear natural extension with sparse bottleneck neural networks, a framework that outperforms its linear counterpart on the prediction task, and produces clearer separation between transcriptomic types in a two-dimensional embedding. Indeed, the path from ion channels coded for by highly specific genes to complex firing patterns is of a very nonlinear nature as implied by the HH model discussed in Chapter 5, which suggests the use of nonlinear computing nodes in prediction tasks.

In Chapter 6 we introduced the field of simulation-based inference and in Chapter 7, we successfully applied NPE-N to HH models explaining electrophysiological behavior obtained with Patch-seq. Beyond simply fitting the HH model to an experimental observation with a point estimate (one combination of model parameter values), we know how likely every possible model parameter combination is in doing so.

In Chapter 8, we attempt to increase our knowledge of the relationship between genotypic and phenotypic identity in neurons. We would be keen on understanding the path from genes expressed in the neuron, to protein function, to behavioral electrophysiology in the form of firing patterns, as best as we can.

8.1 A Semi-deterministic Bridge from Genotype to Phenotype

In order to build a framework that connects transcriptome with HH model parameters and electrophysiology, we will first recollect what we have learned in previous Chapters, and outline what ingredients we can work with. We will then discuss a framework, based on statistical and

deterministic tools to do so. Important to note is that we attempt here to learn a mapping from gene expression levels in the neuron, which we obtained with Patch-seq *experiments*, to HH model parameters, which we derived with NPE-N and that exist in biophysical *simulation* space, back to electrophysiology, also obtained with Patch-seq *experiments*. Finally, we discuss a couple of insights that we can derive from applying this framework on the data set of 955 MoP neurons.

8.1.1 Setting

Transcriptomic reads or gene expression levels of all $n = 955$ MoP neurons as well as their firing patterns and derived electrophysiological features are obtained with Patch-seq, as discussed in Chapter 2 (Automated Extraction of Electrophysiological Features) on page 9. Moreover, in Chapter 7 (Neural Posterior Estimation with Noise) on page 53, we learned how to obtain reliable HH model parameter posterior estimates for all $n = 955$ cells, that is, for each cell, we learned not just MAP parameter estimates, but we learned the probability of each possible parameter combination[1] in reproducing experimentally observed electrophysiology. Let us focus on $n = 955$ MAP parameter estimates for now. In principle, then, we are equipped with:

1: Limited in the space defined by our biologically motivated prior.

- Transcriptomic reads in matrix $\mathbf{X}$ of size $n \times p$, containing expression levels of $p = 1000$ genes for each of the $n = 955$ cells,
- Electrophysiology matrix $\mathbf{Y}$ of size $n \times q$, containing $q = 23$ electrophysiological feature values for the same $n = 955$ cells, and
- HH model parameter matrix $\mathbf{\Theta}$ of size $n \times m$, containing $m = 13$ MAP parameter values for the same $n = 955$ cells.

8.1.2 Approach

One part of the mapping, that is from HH model parameters in the form of MAP parameters estimates collected in $\mathbf{\Theta}$, to electrophysiology $\mathbf{Y}$ is already in place. Indeed, the HH model formulated in Section 5.2.2 provides for a deterministic[2] mapping, a set of differential equations informing the biophysical modeler, at every time step, *how* the membrane voltage evolves $\mathbf{V}_m$ as a function of time t, and additional variables including the rates with which ion channels open α and close β, evolve with the membrane voltage $\mathbf{V}_m(t)$. This all depends highly on the $m = 13$ HH model parameter with which the HH model is set up, before we simulate it.

2: Up to limited current noise.

So what about the relationship between gene expression levels collected in $\mathbf{X}$ and HH model parameters in $\mathbf{\Theta}$? There is only very limited literature available on modeling transcription and translation and, as far as we know, none that describes the full process with dynamical systems from gene expressions to ion channel abundance as experimentally obtained here with Patch-seq. That is not to say that biophysical models of translation, that is the production of proteins from mRNA do not exist. In Adhikari et al. [104] for instance, researchers apply biophysical models composed of coupled differential equations to capture both

transcription and translation. Another good overview of both statistical and deterministic approaches can be found in the work of Zur and Tuller [105].

Intuitively, we can come up with a couple of arguments, however, as we made in Chapter 3 (Sparse Reduced-Rank Regression) on page 15 where we deployed sRRR in order to predict electrophysiology from transcriptome. Similarly, we do not expect all $p = 1000$ genes to be relevant in predicting ion channel abundance in the cell membrane, or its capacity C_m to hold charge concentration differences across the lipid bilayer. In fact, a linear combination of few selected genes might be capable to predict sufficient variability in HH model parameter estimates in our data set.

We thus deploy sRRR, a simple and intuitive statistical tool, without biophysical determinism, to establish a black-box relationship between transcriptome (experiment) and HH model parameters (simulation).

Cross-validated performance (see Section 3.1.3) of an $r = 2$ sRRR model is visualized in Figure 8.1l a: validation set prediction scores R^2_{val} vary with the amount of genes that are selected or the strength of group lasso penalty. We pick 25 genes (Figure 8.1 a, vertical dashed line) and report individual feature prediction scores in Figure 8.1 b. To ease the interpretability, we focus on ion channel and known marker genes only ($p = 427$) and train linear sRRR and nonlinear sBNN models with two-dimensional latent space. Model parameters can be predicted with reasonable accuracy (sRRR: $R^2 = .17 \pm 0.03$, mean ± SD across cross-validation folds, for a model selecting approximately 25 genes) and the nonlinear model performed just as well as the linear one (sBNN: $R^2 = 0.17 \pm 0.03$), so we analyz only the linear model further. In addition, an sRRR model with $r = 13$, built by adding 11 dimensions to the latent space did not improve R^2_{val} (Figure 8.1 a) meaning that two linear combinations contain enough information to predict HH model parameter variability in these neurons.

Our sRRR model predicts some parameters such as the conductance of the voltage dependent or delayed rectifying potassium current ($\overline{g}_{KV31}$ and $\overline{g}_{Kd}$) or the membrane capacitance C particularly well. Other model parameters are less well predicted, such as the leak current E_{leak} or the muscarinic potassium channel $\overline{g}_M$. Interestingly, the r_{SS} parameter is predicted best, which is our custom parameter again that we introduced to fit wider *AP widths* in Pyramidal cells, but also proved useful to overcome some level of data-model mismatch.

The reader is referred to **Bernaerts** et al. [18] for an in-depth description of the architecture. It can be very insightful to check this GitHub repository too for an analysis of the code. In Section 8.1.3, we will continue with embeddings and visualization of the results.

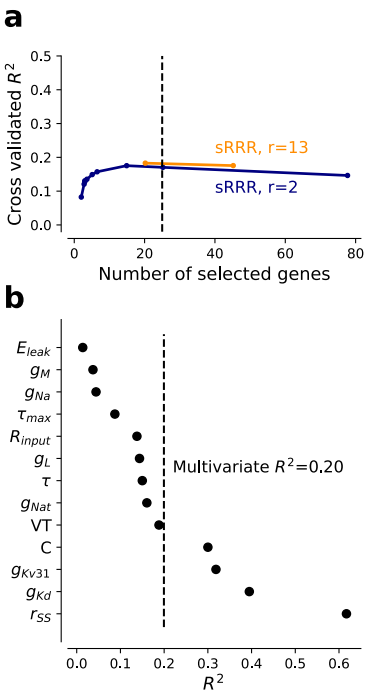

Figure 8.1: Prediction of MAP parameter estimates from gene expression with sRRR. (a) Cross-validation performance for rank-2 and full-rank sRRR models with elastic net penalty. The dashed vertical line shows the performance with 25 genes. **(b)** Rank-2 sRRR model predictive performance for each model parameter, using the entire data set.

8.1.3 Embeddings

The reduced rank of $r = 2$ makes it convenient to produce latent embeddings, that can be overlayed with both gene expression levels as well as predicted model parameter values, as discussed in Chapter 3 and visualized here in Figure 8.2. The two-dimensional latent space

of the sRRR model shows two principal directions of variation, where one separates pyramidal cells from interneurons and the other mostly different interneuron subclasses. In addition, the sRRR model identifies mechanistically plausible relationships: for example, the potassium channel conductances $\overline{g}_{Kv31}$ and $\overline{g}_{Kd}$ are both high in *Pvalb* neurons placed in the lower left corner, predicted by the expression of various potassium channel genes like *Kcnc1*, that constitutes a subunit of the Kv3.1 voltage-gated potassium channel, and *Kcnab3* respectively. Likewise, the calcium channel conductance $\overline{g}_L$ is predicted by high expression of *Cacna2d1* that directly encodes for the alpha-2 and delta subunits in the L-Type calcium channel. Our sRRR model selects *Cacna2d2* as well, which is a paralog gene with opposite expression to *Cacna2d1* (Figure 8.2, left). In addition, classical marker genes like *Vip* act as surrogate cell subclass markers and contribute to the prediction.

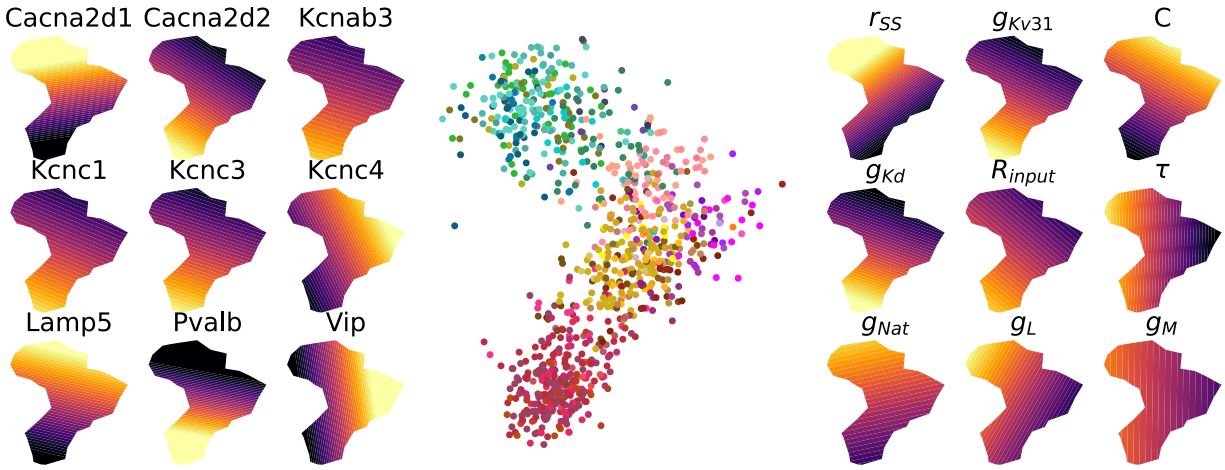

Figure 8.2: Prediction of MAP parameter estimates from gene expression with sRRR. Analogous to Figure 4.5. **(Middle)** rank-2 sRRR model latent space visualization. All 955 MoP neurons are shown. **(Left)** Selected ion channel and marker gene overlays. **(Right)** Predicted model parameter overlays.

8.1.4 Prediction

The approach outlined in Section 8.1.2 can be used to accurately predict HH models for neurons for which we only measured gene expression but not electrophysiology. One can use the trained sRRR model to tell the electrophysiologist how the neuron's membrane voltage will respond to current injection before the Patch-clamp experiment has been conducted, provided that information of gene expression levels in that neuron is available. The only thing one needs to do, is feed the vector of gene expression levels $\mathbf{x}^{(o)}$ into the sRRR model. The output will be a model parameter combination $\boldsymbol{\theta}$, which we can set up the HH model with, and thus simulate. We can now visualize the likely response of the membrane voltage to current injection. We can also derive summarizing statistics to quantify the neuron's electrophysiological characteristics.

We can compare sRRR estimates representative of neuronal families and cell types to MAP estimates derived with posterior estimates from NPE-N. We compute the average MAP estimate over cells belonging to a specific family (Figure 8.3 a, left) and then compute the average over cells belonging to a specific cell type (Figure 8.3 a, right). Analogously, we perform the same computations for sRRR estimates (Figure 8.3 b).

Especially on the family level, the sRRR model predictions capture essential variation in the model parameters (Figure 8.3 a vs b, left). For many cell types, we can also observe similar model parameter variations, but not every cell type is represented as well by the sRRR model (Figure 8.3 a vs b, right). Much of the finer structure shown by average MAP estimates in *Sst* interneuron cell types for instance seems to be lacking in the equivalent sRRR representation. This could be due, again, to a richer variety in firing patterns observed in *Sst* interneurons [11]. When we simulate the HH model with the family-average sRRR estimate, however, and compare it to the HH model simulation set up with the family-average MAP estimate, a very similar picture emerges Figure 8.4. All neuronal families are qualitatively represented by the sRRR model.

[11]: Scala et al. (2021), 'Phenotypic variation of transcriptomic cell types in mouse motor cortex'

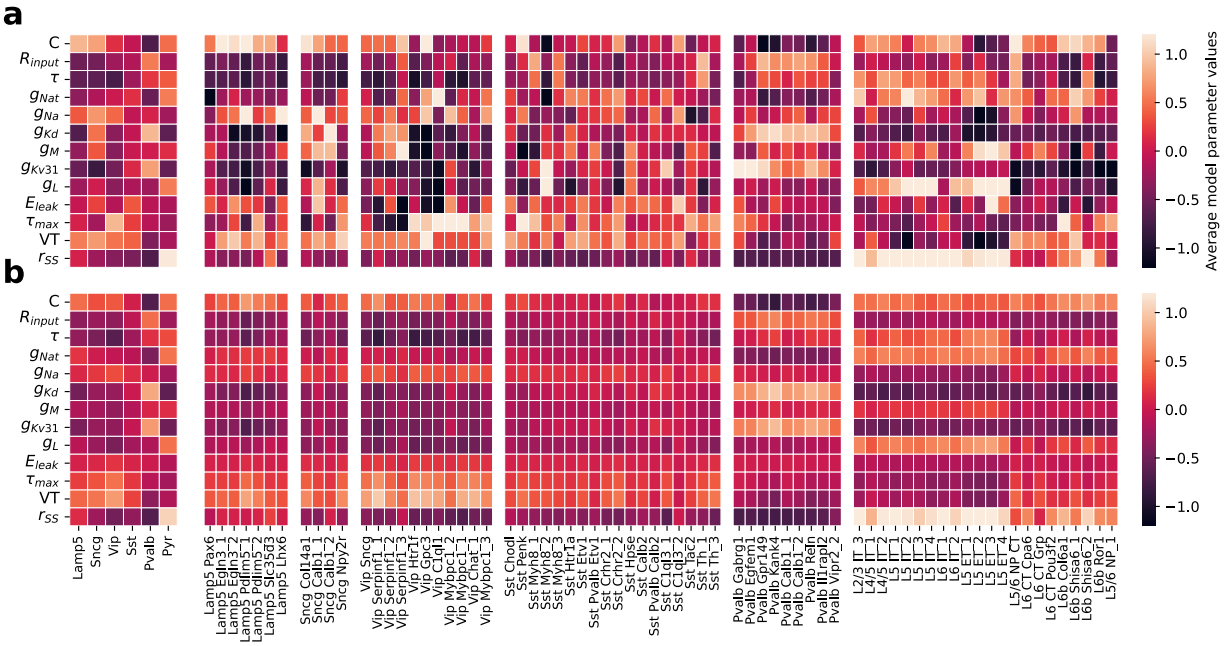

Figure 8.3: MAP parameter estimates and sRRR predictions for each subclass and cell type. (a) MAP parameter estimates averaged over cells belonging to a family and belonging to a transcriptomic cell type (left and right respectively). Estimates are Z-scored. **(b)** Analogous to (a), but with rank-2 sRRR predicted model parameter values.

8.2 Discussion

Derived conductance values in this work largely agree with those found in literature, with interesting exceptions. Pospischil et al. [70] report for instance an average maximum sodium conductance of $\overline{g}_{Na^+} = 50\frac{mS}{cm^2}$ for fast-spiking interneurons much in line with the MAP average of $\overline{g}_{Na^+} = 49\frac{mS}{cm^2}$ we found for *Pvalb* fast-spiking cells. In contrast they used on average $\overline{g}_{Kd} = 5.1$ and $\overline{g}_M = 0.07$ in comparison to our $\overline{g}_{Kd} = 20.8$ and $\overline{g}_M = 1.8$ to fit fast-spiking neurons. Differences can be attributed to a multitude of factors including the fact that we try to replicate mice instead of rat, ferret, cat or guinea-pig electrophysiological measurements, that our model introduces more ion channels and parameters in comparison to theirs providing for potential compensation mechanisms, and that we only replicate membrane voltage responses to a relatively high $300pA$ current injection. Hay et al. [71] report $\overline{g}_{s,Kv3.1} = 693\frac{mS}{cm^2}$, $\overline{g}_{s,Nat} = 2040\frac{mS}{cm^2}$ in the soma and $\overline{g}_{a,Kv3.1} = 0.13\frac{mS}{cm^2}$, $\overline{g}_{a,Nat} = 10.7\frac{mS}{cm^2}$ in the

apical dendrite to fit their neocortical layer 5b neuron whereas we needed on average $\overline{g}_{Kv3.1} = 33\frac{mS}{cm^2}$ and $\overline{g}_{Nat} = 119\frac{mS}{cm^2}$ in our 1-compartment model to fit Pyramidal cells. Indeed, whereas their multi-compartment model allows for a larger variety in conductances across compartments, it is possible that our 1-compartment model is forced to take an 'average' stand.

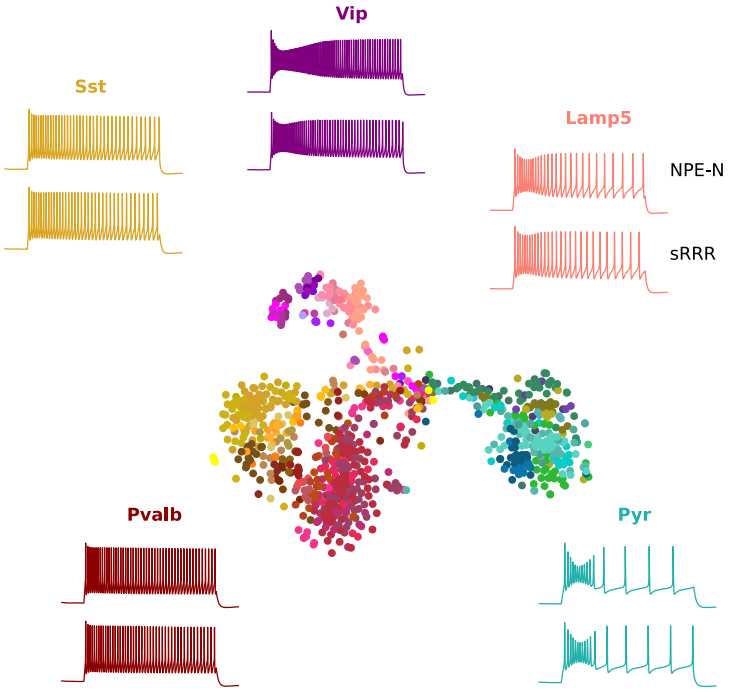

Figure 8.4: Subclass representation of MAP estimates together with sRRR predictions. T-sne embedding as in Figure 7.11 a. Simulation on top is derived from the subclass MAP estimate calculated as in Figure 8.3 a, left. Simulation on the bottom is derived from the subclass sRRR prediction calculated as in as in Figure 8.3 a, left.

Generally, care is warranted in interpreting fitted parameter values when both the model and experimental setup differ (ours vs Pospischil et al. [70] and Hay et al. [71]). Moreover, HH models show redundancy in their parametrizations and compensation mechanisms, as discussed in Section 6.3.

Albeit the one-compartment's demonstrated utility across a range of animals and neuronal types [70], especially for Pyramidal cells there is substantial benefit in using more than one compartment [71, 106, 107]. Furthermore, substantial evidence illustrates that AP generation happens, not at the soma, but at another location called the axon initial segment further suggesting the need for at least another compartment when injecting current in (and recording from) the soma [108, 109]. The more compartments and parameters, however, one includes in the model, introducing more complexity in the underlying governing equations, the longer it will usually take to simulate. Yet, we would need even more simulations to represent the many more parameter combinations that are possible in the higher-dimensional parameter setting, and can therefore take a prohibitively long time. Finally, simulation-based inference has proven its utility mostly for fairly low-dimensional parameter settings. More research is needed to understand whether NPE and NPE-N can prove useful for HH models with more compartments as well as (many) more parameters.

[70]: Pospischil et al. (2008), 'Minimal Hodgkin-Huxley type models for different classes of cortical and thalamic neurons.'

[71]: Hay et al. (2011), 'Models of Neocortical Layer 5b Pyramidal Cells Capturing a Wide Range of Dendritic and Perisomatic Active Properties.'

[106]: Herrera et al. (2020), 'A Minimal Biophysical Model of Neocortical Pyramidal Cells: Implications for Frontal Cortex Microcircuitry and Field Potential Generation'

[107]: Almog et al. (2016), 'Is realistic neuronal modeling realistic?'

[108]: Brette (2015), 'What Is the Most Realistic Single-Compartment Model of Spike Initiation?'

[109]: Leterrier (2018), 'The Axon Initial Segment: An Updated Viewpoint'

The MAP parameter estimates across and within neuronal families corroborate the idea of a banana tree of cortical cell types. In line with Scala et al. [11], we largely find stronger differences in MAP estimates between families and smoother transitions across transcriptomic types within a family.

We also find that different values are needed for the potassium conductance $\overline{g}_{Kv3.1}$ to model *Vip*, *Sst* and *Pvalb* neurons and that the expression of *Kcnc1* varies accordingly (Figure 8.2). In a similar vein, Nandi et al. [103] found different values for $\overline{g}_{Kv3.1}$ and show that *Kcnc1* is differentially expressed between these classes. Importantly, in their work, this example is hand-selected for analysis, while in our analysis the evidence emerges from the sRRR model: $\overline{g}_{Kv3.1}$ is selected due to enforcing the group lasso penalty as predictive of HH model parameter variability across the data set.

Interestingly, the sBNN performs as well as its sRRR linear counterpart on the model parameter prediction task. We speculate that the relationship between gene expression levels and electrophysiological features in cells is complex and highly nonlinear, suggested by the complex dynamics in the governing equations of the HH model (Section 5.2), and improved predictive performance of the nonlinear sBNN model over the linear sRRR model (Chapter 4). Nonlinear computing nodes in deep neural networks might mirror nonlinear dynamics in the HH model. Yet, the relationship between gene expression levels and HH model parameters could be of a more linear nature. Higher expression of genes including *Kcnc1*, *Cacna2d1* and *Kcnab3* linearly correlate with ion channel abundance in the cell membrane, mirrored in higher maximal conductance values including $\overline{g}_{Kv3.1}$ (Figure 8.2) and $\overline{g}_{L}$.

8.3 Conclusion

We established a semi-deterministic bridge from the genotype of a neuron, in the form of gene expression levels, to its phenotype, in the form of electrophysiology. The bridge is semi-deterministic as it is constituted by two parts. The first part is a non-deterministic sparse reduced-rank regression model that can predict fitted HH model parameters based on the cell's gene expression levels. It does not merely predict, but can select most meaningful genes for the task and produce intuitive two-dimensional joint-view embeddings for subsequent exploratory analysis. 'Joint-view' here refers not to the fact that we look at both transcriptome and electrophysiology in the embedding (both experimental viewpoints), but at transcriptome and HH model parameters (experimental and simulation viewpoints). The second part involves the deterministic[3] HH model. Predicted HH model parameters based on gene expression levels, can be used to generate simulations that qualitatively (and quantitatively through extracted features) match electrophysiology in Patch-seq. Note that the simulation, as final output of this bridge, can serve as a proxy for *how* the live cell will respond to current injection, purely based on genetic data.

3: Again, up to limited current noise.

Conclusion and Outlook

9 Conclusion and Outlook

Neurons form the foundational blocks of communication within our nervous system. They serve as highly specialized biological cells, yet their electrical firing patterns, interconnectivity design and intricate morphology suggest a multimodal approach towards understanding them.

As their nature can be appreciated from different angles it is perhaps not surprising that they come in different 'colors and shapes', making a multimodal conforming taxonomy of neuronal cell types challenging. Indeed, transcriptomically, a more wide-spread consensus has been established for the classification of cells into transcriptomic distinct types. Yet, neurons seem to vary more continuously across transcriptomic types (yet also discretely across families) based on their electrophysiology and morphology.

We therefore need statistical tools, that embrace the nervous system's multiple modalities, and that can help us understand the relationship between them. Ideally, we would want to build a *multimodal bridge* so that we may grasp how a neuron's genetic setup leads to phenotypic behavior.

Such a bridge would not merely help us decipher a multimodal conforming 'cortical tree of neural types', it would help us have a clearer understanding of how the expression of certain genes leads to the abundance of specific proteins including ion channels in the cell's membrane, that we know are responsible for the various electrophysiological patterns observed in neurons.

9.1 Summary

In this thesis, we developed models towards the construction of a multimodal bridge. We capitalized on recent advances in machine learning to improve both statistical and biophysical models that together can relate the transcriptome with electrophysiology observed in neurons.

In Chapter 2, we first established a pipeline that automatically extracts expert-defined features that summarize the electrophysiological firing patterns observed in a wide variety of neurons obtained with Patch-seq. Importantly, the user can create 'sanity check plots' to verify the pipeline's computation.

Equipped with computed values for a collection of electrophysiological features and the expression levels of variable genes, we dived into *sparse reduced-rank regression* introduced in Chapter 3. Sparse reduced-rank regression constitutes an intuitive framework to select specific genes, linearly combine them and predict electrophysiological features, also linearly. A *group lasso penalty* enforced on the linear 'encoder' ensures the selection of biologically relevant genes, and the linear encoder's 'low-rank' constraint ensures 'joint-view' two-dimensional embeddings.

We have seen how such visualizations are highly informative and can help guide the exploratory analysis of the multimodal nature inherent to our nervous system.

We then continued with *sparse bottleneck neural networks* introduced in Chapter 4, a natural nonlinear extension to sparse reduced-rank regression, that capitalizes on the predictive power demonstrated by deep neural networks in machine learning. Nonlinear computational units in sparse bottleneck neural networks improve the prediction of electrophysiological measurements. Additionally, two-dimensional joint-view visualizations with sparse bottleneck neural networks capture biological variability for which one needs more dimensions with sparse reduced-rank regression. We have additionally seen the applicability of this nonlinear framework beyond Patch-seq, and showed how we can predict epitopic measurements from gene expressions and produce meaningful paired-data visualizations for CITE-seq.

A case for biophysical models was made in Chapter 5, where we specifically introduced the long-standing Hodgkin-Huxley model with its wide applicability in neural science. We investigated simulations produced with a 'minimal' one-compartment Hodgkin-Huxley-based model and showed its potential to fitting variable neuronal electrophysiology observed with Patch-seq. In order for us to know which model parameters are likely and which less likely — that is, consistent with data, and our prior assumption of the allowed space for parameter values — we introduced *neural posterior estimation* in Chapter 6. With neural posterior estimation, we utilize model simulations to train a *neural* density *estimator*. Given real-world electrophysiological measurements, the trained network outputs a *posterior*, a distribution over model parameters so that for each parameter combination we know the probability with which the Hodgkin-Huxley model produces a simulation consistent with firing patterns observed in cell biology.

We introduced one of the most flexible kind of neural density estimators, *normalizing flows*, at the end of Chapter 6, but showed its limited applicability to real-world experimental data in Chapter 7, such as electrophysiology observed with Patch-seq. We saw that this is not entirely due to the model itself, as we can produce highly satisfying firing patterns, reproducing observed electrophysiology in Patch-seq (see Section 5.2.3). We rather showed how the space of simulated electrophysiology does not adequately cover the space of experimental electrophysiology, causing the neural density estimators — trained with simulations only — to generalize poorly to 'unseen' experimental (test) data. We therefore added isotropic Gaussian noise to the summarizing statistics of training data (the electrophysiological features of Hodgkin-Huxley simulations) in Chapter 7 and showed how this simple strategy can significantly help the density estimator to generalize to electrophysiological data obtained with Patch-seq.

We computed the maximum-a-posteriori estimate together with uncertainty, measured by the posterior entropy, for all Patch-seq cells and showed its variability across families and transcriptomic types. Generally, model parameter values are in agreement with modeling literature for various cell types, and the electrophysiological features derived of

corresponding simulations very similar to the experimental data (Figure 7.11).

We finally made a first attempt in Chapter 8, to our knowledge, bridging the transcriptomic view in a data set of neurons, in the form of gene expression levels, to their phenotypic description, in the form of electrophysiological measurements. We built the bridge in two steps. We first obtained model parameters in Chapter 7 for each neuron that can generate Hodgkin-Huxley-based simulations reproducing their experimentally observed electrophysiology. This constitutes a biophysical bridge from model parameters including potassium, sodium and calcium channel densities, the leak conductance and the membrane capacitance, to firing patterns described with electrophysiological features such as latency, resting membrane potential and action potential width. The advantage of a biophysical model is that we understand *how* membrane voltage responses occur under a set of model parameter values, when current is injected in the cell. We then used sparse reduced-rank regression to predict fitted model parameters from the expression levels of their genes. This two-step procedure can give the electrophysiologist an idea how the neuron will respond to current injection, without having performed the experiment, as long as there is data available regarding this neuron's expected gene expression levels (e.g. from literature or experiments). Indeed, sparse-reduced-rank regression will give you Hodgkin-Huxley-based model parameters based on given gene expression levels, with which you can generate simulations and derive electrophysiological feature values. One can thus both qualitatively and quantitatively predict a neuron's electrophysiology based on available transcriptomic information.

A friendly introduction into the research behind this thesis, moreover applicable to a wide audience can be found on YouTube.

9.2 Future Work

The use of sparse bottleneck neural networks has been demonstrated for Patch-seq data sets, but does not need to limit itself to those. Indeed, any paired data set where it is believed that a combination of specific predictors are sufficient to predict the variability in a response modality, and where direct two-dimensional visualizations can be appreciated for the joint-view exploration of them, is a candidate for sparse bottleneck neural networks. We already demonstrated its use for CITE-seq, but welcome further research into its applicability to other data sets.

Our suggested training data manipulation strategy helps the neural density estimator in neural posterior estimation to see more data close to the experimental manifold. Yet, it is important to note that when we feed experimental data to the trained flow, that we do not post-hoc change the model simulation nor its summarizing statistics, for instance generated with the maximum-a-posteriori estimate. The posterior generalizes better in the same space of model parameters and uses the same unmanipulated Hodgkin-Huxley-based model. We therefore expect neural posterior estimation with noise to be applicable to a plethora of biophysical models where their application to real-world data is crucial. Further research into the general application of neural posterior estimation with noise to

a wide variety of biophysical models and experimental data is therefore warranted.

At the same time, future research in the fields of simulation-based inference and the use of normalizing flows for density estimation could make training data manipulation strategies such as the addition of noise, eventually redundant. It is expected that the ongoing research into normalizing flows [110–113] will lead to highly flexible density estimators, eventually capable of scaling to higher-dimensional settings as well as estimating densities of increasingly sophisticated form. Additionally, it would be more convenient to have simulation-based trained density estimators that generalize better to experimental data without having to manipulate the training data set of simulations, and thus apply simulation-based inference packages such as `sbi` 'as they come' to real-world experimental data such as Patch-seq. We are confident that continued future research into the well-known phenomenon of deep neural networks that poorly generalize to out-of-distribution data, also known as the distribution shift, could eventually make deep neural networks operate better in unseen territory in general.

[110]: Lipman et al. (2023), 'Flow Matching for Generative Modeling'
[111]: Brehmer et al. (2020), 'Flows for simultaneous manifold learning and density estimation'
[112]: Rezende et al. (2020), 'Normalizing Flows on Tori and Spheres'
[113]: Köhler et al. (2020), 'Equivariant Flows: Exact Likelihood Generative Learning for Symmetric Densities'

The two-step produce outlined in this thesis to build a multimodal bridge, has plenty of room for improvement. First, it would be insightful to build a biophysical model that deterministically explains the full process from gene transcription and translation to protein abundance [104, 105, 114–116]) including ion channels in the cell membrane, eventually coupled with models for action potential generation such as the Hodgkin-Huxley model. This would make statistical tools including sparse reduced-rank regression redundant and equip us with a full understanding of *how*, in the nervous system, genotypes lead to phenotypes. Second, as discussed in Section 8.2, the use of multiple compartments would make sense to fit more complicated neuronal phenotypes of for instance excitatory Pyramidal cells, but could rapidly increase the dimensionality of our model parameter space and time needed to produce a sufficient amount of simulations for training density estimators. The latter is especially true when we introduce parameters that guide gene transcription to translation to protein abundance.

[104]: Adhikari et al. (2020), 'Effective Biophysical Modeling of Cell Free Transcription and Translation Processes'
[105]: Zur et al. (2016), 'Predictive biophysical modeling and understanding of the dynamics of mRNA translation and its evolution'
[114]: Siwiak et al. (2010), 'A Comprehensive, Quantitative, and Genome-Wide Model of Translation'
[115]: Reuveni et al. (2011), 'Genome-Scale Analysis of Translation Elongation with a Ribosome Flow Model'
[116]: Brackley et al. (2011), 'The Dynamics of Supply and Demand in mRNA Translation'

We live in exciting times for machine learning and artificial intelligence research in general. Beyond image production [15], speech recognition [16] and games [117], there currently exist machine learning frameworks that can predict protein structure from their sequence [17, 118]. Moreover, OpenAI recently launched `GPT-4`, a large-scale multimodal model that can accept both image and text and produces text [119]. It can for instance be used to help build code, debug and guide mathematical derivations. It is therefore expected that machine learning will prove revolutionizing in neural science as well, and help tackle long-standing questions of how the brain perceives the environment, computes and responds.

[15]: Karras et al. (2018), 'Progressive Growing of GANs for Improved Quality, Stability, and Variation'
[16]: Amodei et al. (2016), 'Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin'
[117]: Silver et al. (2018), 'A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play'
[17]: Jumper et al. (2021), 'Highly accurate protein structure prediction with AlphaFold'
[118]: Service (2020), ''The game has changed.' AI triumphs at protein folding'
[119]: OpenAI (2023), *GPT-4 Technical Report*

As an exciting possible avenue for the future, the use of neural posterior estimation (with noise), or simulation-based inference in general can prove revolutionizing as a machine-learning framework for the field of neural science and medicine. In this thesis, we showed how we can obtain reliable posterior distributions of Hodgkin-Huxley-based model parameters given electrophysiology. In the future, such inference procedures could indicate which parameters are for instance not necessary to

reproduce firing patterns in a neuron from a patient's tissue. In healthy tissue, the posterior might indicate that we *do* need that parameter. Consequently, as some parameters like potassium conductance abundance in the cell membrane are intimately linked with the expression of certain genes such as *Kcnc2*, these analyses can pave the way for targeted gene therapy.

9.2.1 Ethical Considerations

Machine learning experts, and data scientists in general, are not oblivious to the fact that, in order for us to understand the intricate design of our own nervous system, current research involves the killing of animals. We believe, however, that with further research into increasingly sophisticated machine learning tools, especially those that bridge multiple modalities in real-world data such as Patch-seq, we can extract more and more knowledge from data sets of possibly limited size. Combine that with an increased collaborative spirit and public availability of data sets, and the sacrifice of animals can be significantly reduced.

Bibliography

[1] Ann Gill Taylor et al. 'Top-Down and Bottom-Up Mechanisms in Mind-Body Medicine: Development of an Integrative Framework for Psychophysiological Research'. In: *EXPLORE* 6.1 (2010), pp. 29–41. DOI: https://doi.org/10.1016/j.explore.2009.10.004 (cited on page 3).

[2] Karsten Rauss and Gilles Pourtois. 'What is Bottom-Up and What is Top-Down in Predictive Coding?' In: *Frontiers in Psychology* 4 (2013). DOI: 10.3389/fpsyg.2013.00276 (cited on page 3).

[3] Otto Muzik, Kaice T. Reilly, and Vaibhav A. Diwadkar. '"Brain over body"–A study on the willful regulation of autonomic function during cold exposure'. In: *NeuroImage* 172 (2018), pp. 632–641. DOI: https://doi.org/10.1016/j.neuroimage.2018.01.067 (cited on page 3).

[4] Suzana Herculano-Houzel. 'The human brain in numbers: a linearly scaled-up primate brain'. In: *Frontiers in Human Neuroscience* 3 (2009). DOI: 10.3389/neuro.09.031.2009 (cited on page 3).

[5] Robin Tremblay, Soohyun Lee, and Bernardo Rudy. 'GABAergic interneurons in the neocortex: from cellular properties to circuits'. In: *Neuron* 91.2 (2016), pp. 260–292 (cited on pages 3, 30).

[6] Kenneth D Harris and Gordon MG Shepherd. 'The neocortical circuit: themes and variations'. In: *Nature neuroscience* 18.2 (2015), pp. 170–181 (cited on page 3).

[7] Bosiljka Tasic et al. 'Adult mouse cortical cell taxonomy revealed by single cell transcriptomics'. In: *Nature Neuroscience* 19.2 (2016), p. 335 (cited on page 3).

[8] Bosiljka Tasic et al. 'Shared and distinct transcriptomic cell types across neocortical areas'. In: *Nature* 563.7729 (2018), p. 72 (cited on pages 3, 30).

[9] Henry Markram et al. 'Reconstruction and simulation of neocortical microcircuitry'. In: *Cell* 163.2 (2015), pp. 456–492 (cited on page 3).

[10] Xiaolong Jiang et al. 'Principles of connectivity among morphologically defined cell types in adult neocortex'. In: *Science* 350.6264 (2015), aac9462. DOI: 10.1126/science.aac9462 (cited on page 3).

[11] Federico Scala et al. 'Phenotypic variation of transcriptomic cell types in mouse motor cortex'. In: *Nature* 598.7879 (2021), pp. 144–150 (cited on pages 3, 6, 9, 11–13, 20, 22, 23, 28, 31, 61, 67, 69).

[12] Giorgio A. Ascoli et al. 'Petilla terminology: nomenclature of features of GABAergic interneurons of the cerebral cortex'. In: *Nature Reviews Neuroscience* 9.7 (2008), pp. 557–568. DOI: 10.1038/nrn2402 (cited on page 3).

[13] Edward M. Callaway et al. 'A multimodal cell census and atlas of the mammalian primary motor cortex'. In: *Nature* 598.7879 (2021), pp. 86–102. DOI: 10.1038/s41586-021-03950-0 (cited on pages 3, 6).

[14] Hongkui Zeng and Joshua R Sanes. 'Neuronal cell-type classification: challenges, opportunities and the path forward'. In: *Nature Reviews Neuroscience* 18.9 (2017), pp. 530–546 (cited on page 3).

[15] Tero Karras et al. 'Progressive Growing of GANs for Improved Quality, Stability, and Variation'. In: *International Conference on Learning Representations*. 2018 (cited on pages 4, 76).

[16] Dario Amodei et al. 'Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin'. In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, 2016, pp. 173–182 (cited on pages 4, 76).

[17] John Jumper et al. 'Highly accurate protein structure prediction with AlphaFold'. In: *Nature* 596.7873 (2021), pp. 583–589. DOI: 10.1038/s41586-021-03819-2 (cited on pages 4, 76).

[18] **Yves Bernaerts** et al. 'Combined statistical-mechanistic modeling links ion channel genes to physiology of cortical neuron types'. In: *bioRxiv* (2023). DOI: 10.1101/2023.03.02.530774 (cited on pages 5, 53, 62, 65).

[19] **Yves Bernaerts**, Philipp Berens, and Dmitry Kobak. 'Sparse bottleneck neural networks for exploratory non-linear visualization of Patch-seq data'. In: *ArXiv* (2022) (cited on pages 6, 25, 30).

[20] Jonas Beck et al. 'Efficient identification of informative features in simulation-based inference'. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2022 (cited on pages 6, 49, 62).

[21] Dmitry Kobak et al. 'Sparse reduced-rank regression for exploratory visualization of paired multivariate data'. In: *Journal of the Royal Statistical Society, Series C* (2021) (cited on pages 6, 15–18, 20, 21, 23).

[22] Federico Scala et al. 'Layer 4 of mouse neocortex differs in cell types and circuit organization between sensory areas'. In: *Nature Communications* 10 (2019), p. 4174 (cited on pages 6, 12, 20–22).

[23] Cathryn R Cadwell et al. 'Electrophysiological, transcriptomic and morphologic profiling of single neurons using Patch-seq'. In: *Nature Biotechnology* 34.2 (2016), p. 199 (cited on pages 9, 14).

[24] Cathryn R Cadwell et al. 'Multimodal profiling of single-cell morphology, electrophysiology, and gene expression using Patch-seq'. In: *Nature Protocols* 12.12 (2017), p. 2531 (cited on pages 9, 14).

[25] János Fuzik et al. 'Integration of electrophysiological recordings with single-cell RNA-seq data identifies neuronal subtypes'. In: *Nature Biotechnology* 34.2 (2016), p. 175 (cited on pages 9, 20, 22).

[26] Marcela Lipovsek et al. 'Patch-seq: Past, present, and future'. In: *Journal of Neuroscience* 41.5 (2021), pp. 937–946 (cited on pages 9, 34).

[27] Alan Julian Izenman. 'Reduced-rank regression for the multivariate linear model'. In: *Journal of Multivariate Analysis* 5.2 (1975), pp. 248–264 (cited on page 17).

[28] Raja Velu and Gregory C Reinsel. *Multivariate reduced-rank regression: theory and applications*. Vol. 136. Springer Science & Business Media, 2013 (cited on page 17).

[29] Lisha Chen and Jianhua Z Huang. 'Sparse reduced-rank regression for simultaneous dimension reduction and variable selection'. In: *Journal of the American Statistical Association* 107.500 (2012), pp. 1533–1545 (cited on page 18).

[30] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. 'Regularization paths for generalized linear models via coordinate descent'. In: *Journal of Statistical Software* 33.1 (2010), p. 1 (cited on pages 18, 26).

[31] Hui Zou and Trevor Hastie. 'Regularization and variable selection via the elastic net'. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.2 (2005), pp. 301–320 (cited on page 19).

[32] Bradley Efron et al. 'Least angle regression'. In: *The Annals of Statistics* 32.2 (2004), pp. 407–499 (cited on page 19).

[33] Nicolai Meinshausen. 'Relaxed lasso'. In: *Computational Statistics & Data Analysis* 52.1 (2007), pp. 374–393 (cited on page 19).

[34] Daniela M Witten, Robert Tibshirani, and Trevor Hastie. 'A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis'. In: *Biostatistics* 10.3 (2009), pp. 515–534 (cited on page 20).

[35] Nathan W Gouwens et al. 'Integrated morphoelectric and transcriptomic classification of cortical GABAergic cells'. In: *Cell* 183.4 (2020), pp. 935–953 (cited on pages 20, 22).

[36] Francois Chollet et al. *Keras*. 2015. URL: https://github.com/fchollet/keras (cited on page 26).

[37] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/ (cited on page 26).

[38] Jason Yosinski et al. 'How transferable are features in deep neural networks?' In: *Advances in Neural Information Processing Systems*. 2014 (cited on pages 26, 27).

[39] Stéphane Lathuilière et al. 'A comprehensive analysis of deep regression'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019) (cited on page 27).

[40] Jeremy Howard and Sebastian Ruder. 'Universal Language Model Fine-tuning for Text Classification'. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2018, pp. 328–339 (cited on page 27).

[41] Song Han et al. 'Learning both Weights and Connections for Efficient Neural Networks'. In: *Advances in Neural Information Processing Systems*. 2015 (cited on page 28).

[42] Davis Blalock et al. 'What is the State of Neural Network Pruning?' In: *Proceedings of Machine Learning and Systems 2*. 2020 (cited on page 28).

[43] Diederik P. Kingma and Jimmy Ba. 'Adam: A Method for Stochastic Optimization'. In: *International Conference on Learning Representations*. 2015 (cited on pages 28, 45).

[44] Nathan W Gouwens et al. 'Integrated Morphoelectric and Transcriptomic Classification of Cortical GABAergic Cells'. In: *Cell* 183.4 (2020), pp. 935–953 (cited on page 28).

[45] Mikko Muona et al. 'A recurrent de novo mutation in KCNC1 causes progressive myoclonus epilepsy'. In: *Nature Genetics* 47.1 (2015), pp. 39–46. DOI: 10.1038/ng.3144 (cited on page 30).

[46] Randy S. Wymore et al. 'Genomic Organization, Nucleotide Sequence, Biophysical Properties, and Localization of the Voltage-Gated K+ Channel Gene KCNA4/Kv1.4 to Mouse Chromosome 2/Human 11p14 and Mapping of KCNC1/Kv3.1 to Mouse 7/Human 11p14.3-p15.2 and KCNA1/Kv1.1 to Human 12p13'. In: *Genomics* 20.2 (1994), pp. 191–202. DOI: https://doi.org/10.1006/geno.1994.1153 (cited on page 30).

[47] X. Jiang, M. Lachance, and E. Rossignol. 'Chapter 4 - Involvement of cortical fast-spiking parvalbumin-positive basket cells in epilepsy'. In: *Progress in Brain Research* 226 (2016), pp. 81–126 (cited on page 30).

[48] A. Erisir et al. 'Function of Specific K+ Channels in Sustained High-Frequency Firing of Fast-Spiking Neocortical Interneurons'. In: *Journal of Neurophysiology* 82 (1999), pp. 2476–2489 (cited on page 30).

[49] Marlon Stoeckius et al. 'Simultaneous epitope and transcriptome measurement in single cells'. In: *Nature Methods* 14 (2017), pp. 865–868 (cited on pages 32, 33).

[50] Malte D Luecken and Fabian J Theis. 'Current best practices in single-cell RNA-seq analysis: a tutorial'. In: *Molecular Systems Biology* 15.6 (2019), e8746. DOI: https://doi.org/10.15252/msb.20188746 (cited on page 33).

[51] Laurens van der Maaten and Geoffrey Hinton. 'Visualizing data using t-SNE'. In: *Journal of Machine Learning Research* 9.Nov (2008), pp. 2579–2605 (cited on page 33).

[52] Leland McInnes et al. 'UMAP: Uniform Manifold Approximation and Projection'. In: *Journal of Open Source Software* 3.29 (2018), p. 861. DOI: 10.21105/joss.00861 (cited on page 33).

[53] Kevin R. Moon et al. 'Visualizing structure and transitions in high-dimensional biological data'. In: *Nature Biotechnology* 37.12 (2019), pp. 1482–1492. DOI: 10.1038/s41587-019-0336-3 (cited on page 33).

[54] Gökcen Eraslan et al. 'Single-cell RNA-seq denoising using a deep count autoencoder'. In: *Nature Communications* 10 (2019), p. 390 (cited on page 33).

[55] Matthew Amodio et al. 'Exploring single-cell data with deep multitasking neural networks'. In: *Nature Methods* 16 (2019), pp. 1139–1145 (cited on page 33).

[56] Romain Lopez et al. 'Deep generative modeling for single-cell transcriptomics'. In: *Nature Methods* 15 (2018), pp. 1053–1058 (cited on page 33).

[57] Yuhan Hao et al. 'Integrated analysis of multimodal single-cell data'. In: *Cell* 184.13 (2021), 3573–3587.e29. DOI: https://doi.org/10.1016/j.cell.2021.04.048 (cited on page 34).

[58] Ricard Argelaguet et al. 'MOFA+: a statistical framework for comprehensive integration of multi-modal single-cell data'. In: *Genome Biology* 21.1 (2020), p. 111. DOI: 10.1186/s13059-020-02015-1 (cited on page 34).

[59] Rohan Gala et al. 'A coupled autoencoder approach for multi-modal analysis of cell types'. In: *Advances in Neural Information Processing Systems*. 2019, pp. 9263–9272 (cited on page 34).

[60] Rohan Gala et al. 'Consistent cross-modal identification of cortical neurons with coupled autoencoders'. In: *Nature Computational Science* 1 (2021), pp. 120–127 (cited on page 34).

[61] Tal Ashuach et al. 'MultiVI: deep generative model for the integration of multi-modal data'. In: *bioRxiv* (2021). DOI: 10.1101/2021.08.20.457057 (cited on page 34).

[62] Boying Gong, Yun Zhou, and Elizabeth Purdom. 'Cobolt: integrative analysis of multimodal single-cell sequencing data'. In: *Genome Biology* 22.1 (2021), p. 351. DOI: 10.1186/s13059-021-02556-z (cited on page 34).

[63] Adam Gayoso et al. 'Joint probabilistic modeling of single-cell multi-omic data with totalVI'. In: *Nature Methods* 18.3 (2021), pp. 272–282. DOI: 10.1038/s41592-020-01050-x (cited on page 34).

[64] Mohammad Lotfollahi, Anastasia Litinetskaya, and Fabian J. Theis. 'Multigrate: single-cell multi-omic data integration'. In: *bioRxiv* (2022). DOI: 10.1101/2022.03.16.484643 (cited on page 34).

[65] Malte Luecken et al. 'A sandbox for prediction and integration of DNA, RNA, and proteins in single cells'. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. Ed. by J. Vanschoren and S. Yeung. Vol. 1. 2021 (cited on page 34).

[66] Christopher Lance et al. 'Multimodal single cell data integration challenge: Results and lessons learned'. In: *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*. Ed. by Douwe Kiela, Marco Ciccone, and Barbara Caputo. Vol. 176. Proceedings of Machine Learning Research. PMLR, 2022, pp. 162–176 (cited on page 34).

[67] Michael Deistler, Jakob H Macke, and Pedro J Gonçalves. 'Energy-efficient network activity from disparate circuit parameters'. In: *Proceedings of the National Academy of Sciences* 119.44 (2022), e2207632119 (cited on pages 36, 43, 49).

[68] A. L. Hodkgin and A. F. Huxley. 'A quantitative description of membrane current and its application to conduction and excitation in nerve'. In: *The Journal of Physiology* 117 (1952), pp. 500–544 (cited on pages 36–38, 42).

[69] Kalil T. Swain Oldham. 'The doctrine of description : Gustav Kirchhoff, classical physics, and the "purpose of all science" in 19th-century Germany'. eng. PhD thesis. 2008 (cited on page 37).

[70] Martin Pospischil et al. 'Minimal Hodgkin-Huxley type models for different classes of cortical and thalamic neurons.' In: *Biological Cybernetics* 99 (2008), pp. 427–441. DOI: 10.1007/s00422-008-0263-8 (cited on pages 38–40, 67, 68).

[71] Etay Hay et al. 'Models of Neocortical Layer 5b Pyramidal Cells Capturing a Wide Range of Dendritic and Perisomatic Active Properties.' In: *PLOS Computational Biology* 7 (2011). DOI: 10.1371/journal.pcbi.1002107 (cited on pages 38–40, 62, 67, 68).

[72] Marcel Stimberg, Romain Brette, and Dan F.M. Goodman. 'Brian 2, an intuitive and efficient neural simulator'. In: *eLife* 8 (2019). DOI: 10.7554/eLife.47314 (cited on page 39).

[73] Costa M. Colbert and Enhui Pan. 'Ion channel properties underlying axonal action potential initiation in pyramidal neurons'. In: *Nature Neuroscience* 5 (2002), pp. 533–538. DOI: 10.1038/nn0602-857 (cited on page 40).

[74] Roger D. Traub and Richard Miles. *Neuronal networks of the Hippocampus*. Cambridge University Press, 1991 (cited on page 40).

[75] Walter M. Yamada, Christof Koch, and Paul R. Adams. *Methods in neuronal modeling: From synapses to networks. Multiple channels and calcium dynamics*. MIT press, 1989, pp. 97–133 (cited on page 40).

[76] J Rettig et al. 'Characterization of a Shaw-related potassium channel family in rat brain.' In: *The EMBO journal* 11 (1992), pp. 2473–2486. DOI: j.1460-2075.1992.tb05312.x (cited on page 40).

[77] I. Reuveni et al. 'Stepwise repolarization from Ca2+ plateaus in neocortical pyramidal cells: evidence for nonhomogeneous distribution of HVA Ca2+ channels in dendrites'. In: *Journal of Neuroscience* 11 (1993), pp. 4609–4621. DOI: 10.1523/JNEUROSCI (cited on page 40).

[78] Pedro J. Gonçalves et al. 'Training deep neural density estimators to identify mechanistic models of neural dynamics'. In: *eLife* 9 (2020). DOI: 10.7554/eLife.56261 (cited on pages 43, 46, 47, 49, 50).

[79] Maximilian Dax et al. 'Real-Time Gravitational Wave Science with Neural Posterior Estimation'. In: *Physical Review Letters* 127 (2021) (cited on page 43).

[80] Johann Brehmer and Kyle Cranmer. 'Chapter 16: Simulation-Based Inference Methods for Particle Physics'. In: *Artificial Intelligence for High Energy Physics* (2022), pp. 579–611. DOI: 10.1142/9789811234033_0016 (cited on page 43).

[81] Conor Durkan, Iain Murray, and George Papamakarios. 'On Contrastive Learning for Likelihood-free Inference'. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 2771–2781 (cited on page 43).

[82] Jan-Matthis Lueckmann et al. 'Benchmarking Simulation-Based Inference'. In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. Ed. by Arindam Banerjee and Kenji Fukumizu. Vol. 130. Proceedings of Machine Learning Research. PMLR, 2021, pp. 343–351 (cited on page 43).

[83] George Papamakarios et al. 'Normalizing Flows for Probabilistic Modeling and Inference'. In: *Journal of Machine Learning Research* 22 (2021), pp. 1–64 (cited on pages 47, 48).

[84] George Papamakarios, Theo Pavlakou, and Ian Murray. 'Masked autoregressive flow for density estimation'. In: *Advances in Neural Information Processing Systems*. 2017, pp. 2335–2344 (cited on pages 48, 50).

[85] Jan-Matthis Lueckmann et al. 'Likelihood-free inference with emulator networks'. In: *Symposium on Advances in Approximate Bayesian Inference, AABI 2018, Montréal, QC, Canada, December 2, 2018*. Ed. by Francisco J. R. Ruiz et al. Vol. 96. Proceedings of Machine Learning Research. PMLR, 2018, pp. 32–53 (cited on page 48).

[86] George Papamakarios, David Sterratt, and Iain Murray. 'Sequential Neural Likelihood: Fast Likelihood-free Inference with Autoregressive Flows'. In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 2019, pp. 837–848 (cited on page 48).

[87] Eve Marder and Dirk Bucher. 'Understanding Circuit Dynamics Using the Stomatogastric Nervous System of Lobsters and Crabs'. In: *Annual Review of Physiology* 69.1 (2007). PMID: 17009928, pp. 291–316. DOI: 10.1146/annurev.physiol.69.031905.161516 (cited on page 49).

[88] Astrid A. Prinz, Dirk Bucher, and Eve Marder. 'Similar network activity from disparate circuit parameters'. In: *Nature Neuroscience* 7 (2004), pp. 1345–1352. DOI: 10.1038/nn1352 (cited on page 49).

[89] Gina G. Turrigiano. 'Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same'. In: *Trends in Neuroscience* 22.5 (1999), pp. 221–227. DOI: 10.1016/S0166-2236(98)01341-1 (cited on page 49).

[90] Nicholas C. Spitzer. 'New dimensions of neuronal plasticity'. In: *Nature Neuroscience* 2.6 (1999), pp. 489–491. DOI: 10.1038/9132 (cited on page 49).

[91] Micaela Galante et al. 'Homeostatic plasticity induced by chronic block of AMPA/kainate receptors modulates the generation of rhythmic bursting in rat spinal cord organotypic cultures'. In: *European Journal of Neuroscience* 14.6 (2001), pp. 903–917. DOI: https://doi.org/10.1046/j.0953-816x.2001.01710.x (cited on page 49).

[92] Jason N. MacLean et al. 'Activity-Independent Homeostasis in Rhythmically Active Neurons'. In: *Neuron* 37.1 (2003), pp. 109–120. DOI: 10.1016/S0896-6273(02)01104-2 (cited on page 49).

[93] Alfonso junior Apicella and Ivan Marchionni. 'VIP-Expressing GABAergic Neurons: Disinhibitory vs. Inhibitory Motif and Its Role in Communication Across Neocortical Areas'. In: *Frontiers in Cellular Neuroscience* 16 (2022). DOI: 10.3389/fncel.2022.811484 (cited on page 61).

[94] Rohan Taori et al. 'Measuring Robustness to Natural Distribution Shifts in Image Classification'. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 18583–18599 (cited on page 61).

[95] Qitian Wu et al. 'Handling Distribution Shifts on Graphs: An Invariance Perspective'. In: *International Conference on Learning Representations*. 2022 (cited on page 61).

[96] Joaquin Quiñonero-Candela et al. *Dataset Shift in Machine Learning*. MIT PRESS, 2022 (cited on page 61).

[97] Daniel Ward et al. 'Robust Neural Posterior Estimation and Statistical Model Criticism'. In: *Advances in Neural Information Processing Systems*. Ed. by Alice H. Oh et al. 2022 (cited on page 62).

[98] Patrick Cannon, Daniel Ward, and Sebastian M. Schmon. *Investigating the Impact of Model Misspecification in Neural Simulation-based Inference*. 2022. DOI: 10.48550/ARXIV.2209.01845. URL: https://arxiv.org/abs/2209.01845 (cited on page 62).

[99] Marvin Schmitt et al. *Detecting Model Misspecification in Amortized Bayesian Inference with Neural Networks*. 2021. DOI: 10.48550/ARXIV.2112.08866. URL: https://arxiv.org/abs/2112.08866 (cited on page 62).

[100] David T. Frazier, Christopher Drovandi, and David J. Nott. *Synthetic Likelihood in Misspecified Models: Consequences and Corrections*. 2021. DOI: 10.48550/ARXIV.2104.03436. URL: https://arxiv.org/abs/2104.03436 (cited on page 62).

[101] David T. Frazier and Christopher Drovandi. 'Robust Approximate Bayesian Inference With Synthetic Likelihood'. In: *Journal of Computational and Graphical Statistics* 30.4 (2021), pp. 958–976. DOI: 10.1080/10618600.2021.1875839 (cited on page 62).

[102] Nathan W Gouwens et al. 'Systematic generation of biophysically detailed models for diverse cortical neuron types'. In: *Nature Communications* 9.710 (2018). DOI: 10.1038/s41467-017-02718-3 (cited on page 62).

[103] Anirban Nandi et al. 'Single-neuron models linking electrophysiology, morphology, and transcriptomics across cortical cell types'. In: *Cell Reports* 40.6 (2022), p. 111176 (cited on pages 62, 69).

[104] Abhinav Adhikari et al. 'Effective Biophysical Modeling of Cell Free Transcription and Translation Processes'. In: *Frontiers in Bioengineering and Biotechnology* 8 (2020). DOI: 10.3389/fbioe.2020.539081 (cited on pages 64, 76).

[105] Hadas Zur and Tamir Tuller. 'Predictive biophysical modeling and understanding of the dynamics of mRNA translation and its evolution'. In: *Nucleic Acids Research* 44.19 (Sept. 2016), pp. 9031–9049. DOI: 10.1093/nar/gkw764 (cited on pages 65, 76).

[106] Beatriz Herrera et al. 'A Minimal Biophysical Model of Neocortical Pyramidal Cells: Implications for Frontal Cortex Microcircuitry and Field Potential Generation'. In: *The Journal of Neuroscience* 40 (44 2020), pp. 8513–8529. DOI: 10.1523/JNEUROSCI.0221-20.2020 (cited on page 68).

[107] Mara Almog and Alon Korngreen. 'Is realistic neuronal modeling realistic?' In: *Journal of Neurophysiology* 116 (5 2016), pp. 2180–2209. DOI: 10.1152/jn.00360.2016 (cited on page 68).

[108] Romain Brette. 'What Is the Most Realistic Single-Compartment Model of Spike Initiation?' In: *PLoS Computational Biology* 11 (4 2015). DOI: 10.1371/journal.pcbi.1004114 (cited on page 68).

[109] Christophe Leterrier. 'The Axon Initial Segment: An Updated Viewpoint'. In: *The Journal of Neuroscience* 38 (9 2018), pp. 2135–2145. DOI: 10.1523/JNEUROSCI.1922-17.2018 (cited on page 68).

[110] Yaron Lipman et al. 'Flow Matching for Generative Modeling'. In: *The Eleventh International Conference on Learning Representations*. 2023 (cited on page 76).

[111] Johann Brehmer and Kyle Cranmer. 'Flows for simultaneous manifold learning and density estimation'. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 442–453 (cited on page 76).

[112] Danilo Jimenez Rezende et al. 'Normalizing Flows on Tori and Spheres'. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 8083–8092 (cited on page 76).

[113] Jonas Köhler, Leon Klein, and Frank Noe. 'Equivariant Flows: Exact Likelihood Generative Learning for Symmetric Densities'. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 5361–5370 (cited on page 76).

[114] Marlena Siwiak and Piotr Zielenkiewicz. 'A Comprehensive, Quantitative, and Genome-Wide Model of Translation'. In: *PLOS Computational Biology* 6 (2010), pp. 1–15. DOI: 10.1371/journal.pcbi.1000865 (cited on page 76).

[115] Shlomi Reuveni et al. 'Genome-Scale Analysis of Translation Elongation with a Ribosome Flow Model'. In: *PLOS Computational Biology* 7.9 (2011), pp. 1–18. DOI: 10.1371/journal.pcbi.1002127 (cited on page 76).

[116] Chris A. Brackley, M. Carmen Romano, and Marco Thiel. 'The Dynamics of Supply and Demand in mRNA Translation'. In: *PLOS Computational Biology* 7.10 (2011), pp. 1–16. DOI: 10.1371/journal.pcbi.1002203 (cited on page 76).

[117] David Silver et al. 'A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play'. In: *Science* 362.6419 (2018), pp. 1140–1144. DOI: 10.1126/science.aar6404 (cited on page 76).

[118] Robert F. Service. ''The game has changed.' AI triumphs at protein folding'. In: *Science* 370.6521 (2020), pp. 1144–1145. DOI: 10.1126/science.370.6521.1144 (cited on page 76).

[119] OpenAI. *GPT-4 Technical Report*. 2023 (cited on page 76).