Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

DEEP HYBRID MODELING OF NEURONAL DYNAMICS USING GENERATIVE ADVERSARIAL NETWORKS

by
Soheil Saghafi

Mechanistic modeling and machine learning methods are powerful techniques for approximating biological systems and making accurate predictions from data. However, when used in isolation these approaches suffer from distinct shortcomings: model and parameter uncertainty limit mechanistic modeling, whereas machine learning methods disregard the underlying biophysical mechanisms. This dissertation constructs Deep Hybrid Models that address these shortcomings by combining deep learning with mechanistic modeling. In particular, this dissertation uses Generative Adversarial Networks (GANs) to provide an inverse mapping of data to mechanistic models and identifies the distributions of mechanistic model parameters coherent to the data.

Chapter 1 provides background information on the major ideas that are important for this dissertation. It provides an introduction to parameter inference techniques and highlights some of the methodologies available for solving stochastic inverse problems. Chapter 2 starts with a brief overview of the Hodgkin-Huxley model, and then introduces other conductance-based models that are used in the dissertation. The first part of Chapter 3 focuses on methodologies for global sensitivity analysis and global optimization, in particular Sobol sensitivity analysis and Differential Evolution. The second part of this chapter explains how the Markov chain Monte Carlo (MCMC) algorithm can be used for parameter inference and then introduces a novel parameter inference tool based on conditional Generative Adversarial Networks (cGANs). In Chapter 4, the performance of cGAN and MCMC are compared on synthetic targets. Chapter 5 then uses cGAN to infer biophysical

parameters from experimental data recorded at the single-cell and network levels from neurons involved in the regulation of circadian (∼24-hour) rhythms and from brain regions associated with neurodegenerative diseases. Finally, conclusions and suggestions for further research are presented in Chapter 6.

DEEP HYBRID MODELING OF NEURONAL DYNAMICS USING GENERATIVE ADVERSARIAL NETWORKS

by
Soheil Saghafi

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology and
Rutgers, The State University of New Jersey – Newark
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Mathematical Sciences

Department of Mathematical Sciences
Department of Mathematics and Computer Science, Rutgers-Newark

May 2023

APPROVAL PAGE

DEEP HYBRID MODELING OF NEURONAL DYNAMICS USING GENERATIVE ADVERSARIAL NETWORKS

Soheil Saghafi

Dr. Casey O. Diekman, Dissertation Advisor — Date
Associate Professor, Department of Mathematical Sciences, NJIT

Dr. Victor Matveev, Committee Member — Date
Professor, Department of Mathematical Sciences, NJIT

Dr. David Shirokoff, Committee Member — Date
Associate Professor, Department of Mathematical Sciences, NJIT

Dr. Usman Roshan, Committee Member — Date
Associate Professor, Department of Computer Science, NJIT

Dr. James Kozloski, Committee Member — Date
Research Staff, Center for Multiscale Systems Biology and Modeling,
IBM Thomas J. Watson Research Center, Yorktown Heights, NY

BIOGRAPHICAL SKETCH

Author: Soheil Saghafi

Degree: Doctor of Philosophy

Date: May 2023

Undergraduate and Graduate Education:

- Doctor of Philosophy in Mathematical Sciences,
  New Jersey Institute of Technology, Newark, New Jersey, 2023
- Master of Science in Applied Mathematics,
  Khajeh Nasir Toosi University of Technology, Tehran, Iran, 2015
- Bachelor of Arts in Mathematics
  Damghan University, Semnan, Iran, 2013

Major: Mathematical Sciences

Publications:

E. Khan, S. Saghafi, C. O. Diekman and H. G. Rotstein, "The emergence of polyglot entrainment responses to periodic inputs in vicinities of Hopf bifurcations in slow-fast systems," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 2022.

S. Saghafi, P. Sanaei, "Dynamic Entrainment: A deep learning and data-driven process approach for synchronization in the Hodgkin-Huxley model," *Bulletin of Mathematical Biology [57]. (Submitted)*

S. Saghafi, T. Rumbell, V. Gurev, J. Kozloski, F. Tamagnini, K. Wedgwood and C. O. Diekman, "Inferring parameters of pyramidal neuron excitability in mouse models of Alzheimer's disease using biophysical modeling and deep learning," *Bulletin of Mathematical Biology [56]. (Submitted)*

S. Saghafi, E. Khan, N. Wei and C. O. Diekman, "Circadian rhythms and COVID-19: modeling circadian clock regulation of immune system response to SARS-CoV-2 infection and treatment with Remdesivir," *(In Preparation).*

H. Anwar, S. Saghafi, L. Deng, J. Denham, N. Cohen, C. O. Diekman, G. Haspel, "Synaptic connections form bidirectional undulatory motor pattern from distributed coupled-oscillators in a model of C. elegans locomotion network," *(In Preparation).*

E. Khan, S. Saghafi, C. O. Diekman, H. G. Rotstein, "Mechanisms of oscillations in post-translational circadian clock models," *(In Preparation)*.

Presentations:

S. Saghafi, T. Rumbell, V. Gurev, J. Kozloski, F. Tamagnini, K. Wedgwood and C. O. Diekman, "Deep hybrid modeling of neuronal dynamics using generative adversarial networks," *American Mathematical Society Special Session on Mathematical Modeling*, Georgia Institute of Technology, Atlanta, GA, 2023.

S. Saghafi, "A brief introduction of generative adversarial networks," *Invited Talk*, Machine Learning and Optimization Seminar, NJIT, Newark, NJ, February, 2023.

S. Saghafi, "A brief introduction of Convolutional Neural Network (CNN) plus a hands-on project in digit recognition," *Invited Talk*, Machine Learning and Optimization Seminar, NJIT, Newark, NJ, October, 2022.

S. Saghafi, "A Markov chain Monte Carlo version of the genetic algorithm differential evolution: easy Bayesian computing for real parameter spaces," *Invited Talk*, Machine Learning and Optimization Seminar, NJIT, Newark, NJ, April, 2022.

S. Saghafi and E. Khan "Circadian rhythms and COVID-19: modeling circadian clock regulation of immune system response to SARS-CoV-2 infection and treatment with Remdesivir," *NSF Student Conference on COVID-19 Modeling*, University of Texas at Dallas (Virtual), January 29, 2021.

S. Saghafi, "Mathematical modeling of interaction between innate and adaptive immune responses in COVID-19 and implications for viral pathogenesis," *Invited Talk*, IBM Thomas J. Watson Research Center, NY, July, 2021.

Posters:

S. Saghafi, T. Rumbell, K. Wedgwood, F. Tamagnini, V. Gurev, J. Kozloski and C. O. Diekman, "Deep hybrid modeling of neuronal dynamics using generative adversarial networks," Society for Neuroscience (SfN), San Diego, CA, November, 2022.

S. Saghafi, T. Rumbell, K. Wedgwood and C. O. Diekman, "Deep hybrid modeling of neuronal dynamics using generative adversarial networks," NAI-NJIT Workshop on Sustainable Societies: Data Revolution, New Jersey Institute of Technology, Newark, NJ, October, 2022.

S. Saghafi, T. Rumbell, K. Wedgwood and C. O. Diekman, "Deep hybrid modeling of neuronal dynamics using generative adversarial networks," Frontiers in Applied & Computational Mathematics (FACM), New Jersey Institute of Technology, Newark, NJ, May, 2022.

T. Rumbell, S. Saghafi, C. O. Diekman, J. Kozloski and V. Gurev, "Generative adversarial networks for parameter inference in populations of neuroscience models," Society for Neuroscience (SfN), San Diego, CA, November, 2021.

S. Saghafi, C. O. Diekman, "Deep hybrid modeling of neuronal dynamics using generative adversarial networks," Dana Knox Student Research Showcase, New Jersey Institute of Technology, Newark, NJ, April, 2019.

Life is a miracle,

To the creator of this miracle who gave me the chance of being a part of it.

ACKNOWLEDGMENT

First and foremost, I would like to thank Dr. Casey Diekman who was the best mentor I've ever had and I was privileged to complete my dissertation under his supervision. In these past several years, I have had his support in different ways, not only in my research but also in my daily life. If I went back in time, I would always choose him again as my dissertation advisor.

I would also like to thank my committee members professors: Victor Matveev and David Shirokoff from the department of mathematical sciences, professor Usman Roshan from the department of computer science and last but not least Dr. James Kozloski from the multiscale systems biology and modeling center at IBM, for their time and assistance with this dissertation.

Next, I would like to thank Dr. Timothy Rumbell for mentoring me during my internship at IBM T. J. Watson Research Center and inspiring me in computational neuroscience. Thanks also to Dr. Kyle Wedgwood for more than two years of collaboration and providing helpful discussions regarding this project. I want to express my gratitude to Dr. Horacio Rotstein for his insightful guidance in both the introductory and advanced computational neuroscience courses, as well as for his collaboration on a couple of projects throughout my PhD program. Last but not least, I want to thank Dr. Victoria Livingstone for reviewing my dissertation and adding some linguistic suggestions to make it easier to understand for others outside of my field.

I am grateful for the funding and assistance offered by the Department of Mathematical Sciences, which enabled me to finish this dissertation. Additionally, I would like to thank the National Science Foundation for their financial support of this project through grants 1555237 and 2152115.

To all my friends: Hamed Azimi, Christeen Bisnath, Mahdi Bandegi, Shima Ghavimi, Reza Hashemipour, Milad Shojaee and Pejman Sanaei for their countless supports throughout the Ph.D. program and Lauren Barnes, Subhrasish Chakraborty, Emel Khan, Matthew Moye, Tadanaga Takahashi, Connor Robertson and Kosuke Sugita from the math department for their insightful discussions on math and machine learning problems and for helping to maintain a wonderful atmosphere in the office.

Finally, I would like to thank my family: my father, Mehran Saghafi for the wonderful stories he told me when I was a child and for his invaluable advice throughout my life; my mother, Zohreh Asadi for enrolling me in primary school and accompanying me on the first day and also providing a home environment conducive for learning; my brother, Sepehr Saghafi for always being a good friend to me; and last but not least, my lovely wife, Sepideh Nikookar who came to the U.S. and left her family behind to support me as I pursue my dreams.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Overview

Mechanistic models are built on an understanding of how the components of a system behave, and are powerful tools for approximating biological systems. However, determining whether or not such a model and its outputs are coherent with a set of experimental observations is a major challenge since mechanistic models contain many unknown parameters and are not amenable to statistical inference due to their non-invertibility. The main difficulty in solving the inverse problem for mechanistic models arises from intractability of the likelihood function [13]. On the other hand, purely statistical models with tractable likelihoods do not provide much insight into the underlying biological mechanisms [20]. Improvements in methodologies for parameter identification of mechanistic models are needed to advance towards reproducibility of real-world data by models capable of both predicting and explaining physiological and pathophysiological phenomena.

1.2 Deep Hybrid Modeling

Empirical models are based on patterns observed in data rather than underlying mechanistic knowledge of a system's components. Machine learning (a.k.a. statistical learning) is a type of empirical modeling where algorithms are trained on data and then are capable of producing accurate data-driven predictions using computational techniques that do not rely on a predefined equation as a model [3, 42, 59]. However, it is often difficult to interpret how a machine learning algorithm is making its predictions in terms of the underlying biology. Hybrid modeling (HM) is a framework that integrates the machine learning approach with mechanistic models [27, 35, 39]. The primary goal of this thesis is the development and application of a flexible

hybrid modeling architecture that enables the mapping of streams of experimental observations into the space of biophysical model parameters (see Figure 1.1). We will apply this framework to tackle a set of important biological questions involving neuroscience and circadian (∼24-hour) rhythms.

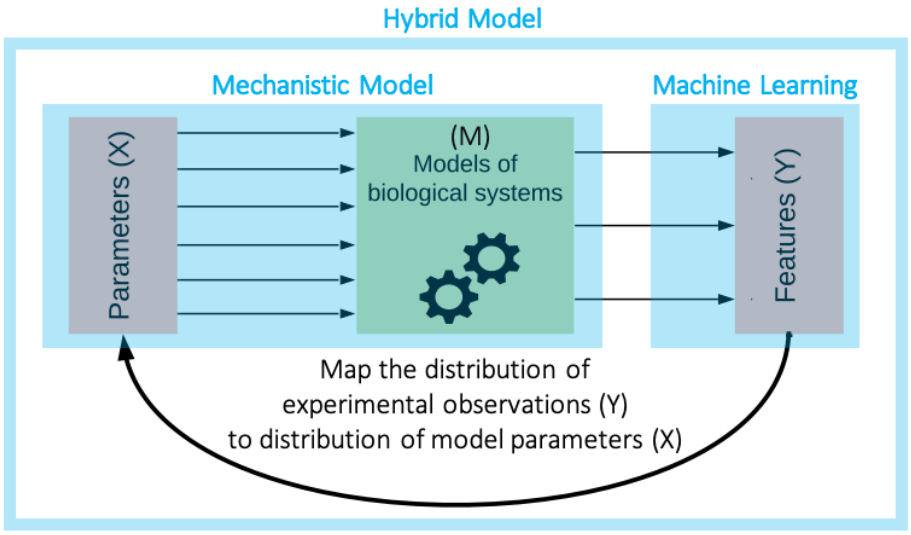

Figure 1.1 Overview of the hybrid modeling concept. HM is a hybrid machine learning/mechanistic modeling library developed at IBM Research by Viatcheslav Gurev, Jaimit Parikh, and Timothy Rumbell. The HM library enables the mapping of a stream of experimental observations (Y) into the space of biophysical parameters (X) through a mechanistic model (M). Given the joint density of features extracted from experimental observations, the goal of HM is to perform an inverse mapping to identify the distribution of mechanistic modeling parameters *coherent* to the data (see Chapter 3 for definition of *coherent*).

Machine learning (ML) algorithms can be divided into two categories: supervised learning and unsupervised learning. In supervised learning, ML models learn the boundaries between classes or labels in a dataset in the form of classification or regression problems, and then are used to make predictions given new data points. These models, which result from various ML approaches such as logistic regression, support vector machines, and classification trees, are referred to as *discriminative* models [3,42,59]. In unsupervised learning, ML algorithms such as Bayesian networks, autoregressive models, and generative adversarial networks (GANs) are used to

perform probability and likelihood estimation. These models are able to generate new data instances, thus they are referred to as *generative* models [22, 64]. In this study, we will develop GANs to solve generative modeling problems where experimental data comes from multiple individuals (or cells) across a population, and the task is to sample (or generate) mechanistic model parameter sets such that the resulting population of model outputs is consistent with the observed population data. This is referred to as the stochastic inverse problem or the "population of models" problem [10, 34]. Here, we use deep learning to perform inversion of complex biophysical models and enable the mapping of experimental data into the space of biophysical model parameters. Since this approach combines deep learning with mechanistic modeling, we refer to it as deep hybrid modeling (DeepHM).

In biological systems, the tremendous amount of inherent cell-to-cell variability presents a significant challenge to mapping experimental data to underlying cellular mechanisms. It is common to handle this variability by simply averaging over the data and finding a single set of model parameters that best fits the averaged data. The "populations of models" approach allows deterministic models to reflect the inherent variability in biological data through identification of not just the single best parameter set but a population of parameter sets such that the output of the group of models displays the same heterogeneity as the population being modeled [2, 7, 20, 34, 38, 54, 60]. The problem of constructing populations of deterministic models and identifying distributions of model input parameters from stochastic observations from multiple individuals in a population is known as the stochastic inverse problem (SIP). State-of-the-art methods for solving SIPs apply Bayesian inference techniques, including Markov chain Monte Carlo (MCMC) sampling, and are limited to finding a distribution for a single set of observations [11, 34, 49, 55]. To draw inferences about a new target dataset, the SIP would have to be solved again. We have recently proposed an alternative approach to solving SIPs, using generative

adversarial networks (GANs), that enables *amortized inference*— i.e., the trained GAN can be reused on many target datasets without re-solving the SIP [48].

GANs are a deep learning paradigm involving two artificial neural networks that compete with each other in a minimax game. The *generator* network attempts to produce fake samples that are as similar as possible to a distribution of real samples, and the *discriminator* network tries to distinguish fake samples from real samples. Since being introduced in 2014, GANs have garnered significant interest across a wide range of fields including applications in image processing, cybersecurity, and cryptography [21,24]. To solve SIPs, we use a conditional GAN (cGAN) structure [40] where the generator is trained with parameter sets X conditioned on the output features Y of a mechanistic model.

1.3 Structure of the Dissertation

This dissertation is organized as follows. Chapter 2 introduces different types of mechanistic/biophysical models that we will be using throughout this study, namely conductance-based models of neuronal dynamics. In Chapter 3, we briefly cover the optimization, sensitivity analysis, and parameter inference methodologies that are applied throughout this dissertation. More specifically, we provide background material on differential evolution (a global optimization strategy), Sobol indices (a type of global sensitivity analysis), and MCMC. We describe GANs and cGANs and then illustrate our parameter inference methodology using the Rosenbrock function as a toy model at the end of this chapter. In Chapter 4, we show that cGAN outperforms a benchmark MCMC method on a relatively simple parameter inference task. We then validate the ability of cGAN to accurately infer complex parameter distributions through a series of tests with synthetic targets extracted from different biophysical models. In Chapter 5, we apply the trained cGAN for parameter estimation on experimental data as a target and use the inferred parameter distributions in three

different contexts to identify the ionic conductances that: (1) control day/night variation of electrical activity in circadian clock neurons, (2) are affected by aging and mutations associated with Alzheimer's disease in CA1 pyramidal neurons, and (3) are affected by mutations associated with Huntington's disease in a network of medium spiny neurons. We conclude this dissertation with a discussion of alternative methods and future work in Chapter 6. Some of the work in this dissertation on the application of cGANs to CA1 pyramidal neuron excitability has been published as a preprint [56].

CHAPTER 2

CONDUCTANCE-BASED MODELS

This chapter will introduce five different mechanistic models that we will use within the DeepHM framework. All of them are conductance-based models that describe the electrical dynamics of neurons.

2.1 Hodgkin-Huxley model

The conductance-based modeling formalism was introduced by Hodgkin and Huxley in 1952 to explain the ionic mechanisms underlying the initiation and propagation of action potentials in the squid giant axon [25]. Hodgkin-Huxley-type models use an equivalent electric circuit to describe the excitability of the cell membrane. Figure 2.1A represents the equivalent circuit underlying the Hodgkin-Huxley equations [16] showing the cell membrane capacitance C, ionic conductances g_x, where $x \in \{Na, K, L\}$ for sodium, potassium, and leak, respectively, the equilibrium potential or reversal potential E_x, and the applied current as a function of time t, $I_{app}(t)$. Figure 2.1B shows the schematic of the cell membrane potential V (mV) versus time t (ms) during an action potential with three main phases: (1) depolarization of the cell membrane through activation of the voltage-dependent sodium channels, (2) repolarization of the cell membrane through inactivation of the sodium channels and activation of potassium channels, and (3) refractory period during which time the cell membrane is not able to fire or generate another action potential.

At steady state, the inside of the cell is more negatively charged compared to the outside, leading to a hyperpolarized membrane potential V of around -60 or -70 mV. The cell membrane acts as a capacitor C that separates charges, however there are channels in the membrane that conduct certain ions. For sodium ions,

the extracellular concentration is higher than the intracellular concentration, while for potassium ions the opposite is true. Due to these ionic concentration gradients, when sodium channels are open Na^+ ions tend to flow into the cell and depolarize the membrane potential, whereas K^+ ions flow out of the cell and hyperpolarize the membrane potential when potassium channels are open. These ionic currents are described by Ohm's Law, for example, for sodium ions we have $I_{Na} = g_{Na}m^3h(V - E_{Na})$, where I_{Na} is the sodium current, g_{Na} is the maximal conductance of the sodium channel and E_{Na} is the equilibrium potential or reversal potential at which no current would flow. The gating variable m describes whether the sodium channels are open ($m \to 1$) or closed ($m \to 0$), and the other gating variable h models whether the sodium channels are inactivated ($h \to 0$) or de-inactivated ($h \to 1$). Relatively similar equations could explain the ionic current for potassium and leak ions I_K and I_L respectively (see (2.1), which is explained below). Note that, the potassium channel has a single gating variable n, since it does not inactivate. In addition, the leak channel L is passive and thus there are not any gating variables associated with it.

The equations representing the Hodgkin-Huxley model are

$$\begin{aligned}
C\frac{dV}{dt} &= I_{app} - g_{Na}m^3h(V-E_{Na}) - g_K n^4(V-E_K) \\
&\quad -g_L(V-E_L), \\
\frac{dm}{dt} &= \alpha_m(1-m) - \beta_m m = \frac{m_\infty - m}{\tau_m}, \\
\frac{dh}{dt} &= \alpha_h(1-h) - \beta_h h = \frac{h_\infty - h}{\tau_h}, \\
\frac{dn}{dt} &= \alpha_n(1-n) - \beta_n n = \frac{n_\infty - n}{\tau_n}, \\
x_\infty &= \frac{\alpha_x}{\alpha_x+\beta_x}, \quad \tau_x = \frac{1}{\alpha_x+\beta_x}, \quad x \in \{m,h,n\}, \\
\alpha_m &= 0.1\frac{V+40}{1-\exp\left(-\frac{V+40}{10}\right)}, \quad \beta_m = 4\,\exp\left(-\frac{V+65}{18}\right), \\
\alpha_h &= 0.07\,\exp\left(-\frac{V+65}{20}\right), \quad \beta_h = \frac{1}{1+\exp\left(-\frac{V+35}{10}\right)}, \\
\alpha_n &= 0.01\frac{V+55}{1-\exp\left(-\frac{V+55}{10}\right)}, \quad \beta_n = 0.125\,\exp\left(-\frac{V+65}{80}\right).
\end{aligned} \tag{2.1}$$

The parameter values and gating functions are $C = 1\ \mu\text{F/cm}^2$, $g_{Na} = 120\ \text{mS/cm}^2$, $g_K = 36\ \text{mS/cm}^2$, $g_L = 0.3\ \text{mS/cm}^2$, $E_{Na} = 55\ \text{mV}$, $E_K = -77\ \text{mV}$ and $E_L = -54.4\ \text{mV}$.

The current-balance equation in (2.1) originates from Kirchhoff's Current Law (the algebraic sum of all currents entering and exiting a node in a circuit must equal zero) and says that the sum of the capacitive and ionic currents must be equal to the applied current I_{app}. The gating variables m, h and n in (2.1) are based on a two-state channel model with x being the fraction of open channels and $1-x$ the fraction of closed channels, with voltage-dependent rates of closed channels opening α_x and open channels closing β_x, where $x \in \{m,h,n\}$. Note that the gating variable equations can be rewritten in terms of steady-state activation x_∞ and time constant τ_x functions as shown in (2.1). By solving this system of ODEs (2.1), we can obtain the membrane potential over time $V(t)$, as shown in Figure 2.1B. The dynamics of the voltage-dependent gating variables, with m being on a faster time scale than h

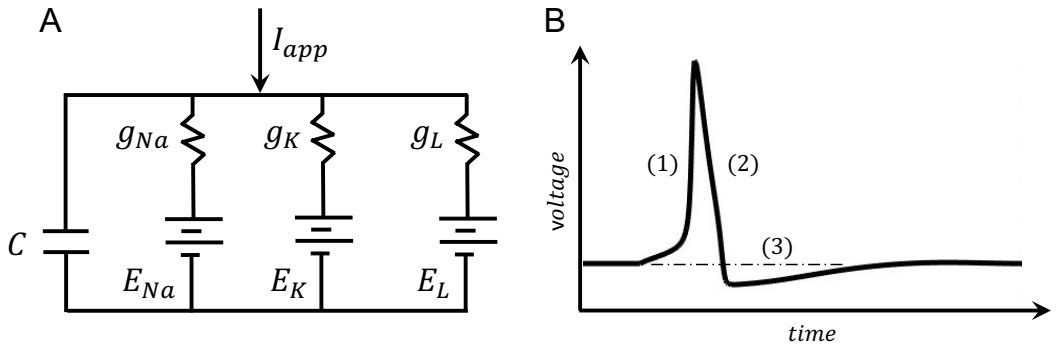

Figure 2.1 (A) Equivalent circuit underlying the Hodgkin-Huxley equations [16] showing the cell membrane capacitance C, ionic conductances g_x, where $x \in \{Na, K, L\}$ for sodium Na, potassium K and leak L, the equilibrium potential or reversal potential E_x, and the applied current $I_{app}(t)$ as a function of time t. (B) Schematic of the cell membrane potential V (mV) versus time t (ms) with an action potential and three main phases: (1) depolarization of the cell membrane through activation of the voltage-dependent sodium channels, (2) repolarization of the cell membrane through inactivation of the sodium channels and activation of potassium channels, and (3) refractory period time during which the cell membrane is not able to fire or generate another action potential.

and n, create positive and negative feedback in the cell membrane. This results in a stable periodic solution that corresponds to the repetitive firing of action potentials. We use the Hodgkin-Huxley model for parameter inference tests on synthetic target data in Section 4.2.

2.2 Morris-Lecar Model

The Morris-Lecar model is a conductance-based model that has become a canonical low-dimensional model for studying neuronal dynamics. Originally developed by Catherine Morris and Harold Lecar in 1981 to model barnacle muscle fibers [43], it consists of three ionic currents: voltage-dependent inward calcium, voltage-dependent outward potassium, and passive leak. The equations for the Morris–Lecar model are as follows:

$$
\begin{aligned}
C_m \frac{dv}{dt} &= I_{app} - g_l(v - E_l) - g_{Ca} m_\infty(v)(v - E_{Ca}) - g_k n(v - E_k), \\
\frac{dn}{dt} &= \phi(n_\infty(v) - n)/\tau_n(v), \\
m_\infty(v) &= \frac{1}{2}[1 + \tanh((v - v_1)/v_2)], \\
n_\infty(v) &= \frac{1}{2}[1 + \tanh((v - v_3)/v_4)], \\
\tau_n(v) &= 1/\cosh((v - v_3)/(2v_4)).
\end{aligned} \tag{2.2}
$$

For different values of the parameters, the Morris-Lecar model can exhibit qualitatively different types of excitability. We will use the Morris-Lecar model and cGAN to generate models with different excitability types in Section 4.1.

2.3 Suprachiasmatic Nucleus Neuron Model

The suprachiasmatic nucleus (SCN) is the central circadian (∼24-hour) clock in mammals. A conductance-based model of SCN neurons is given by the following equations [4]:

$$
C\frac{dV}{dt} = I_{app}(t) - I_{Na} - I_K - I_{Ca} - I_{L_{Na}} - I_{L_K} - I_H - I_A \tag{2.3}
$$

$$
\frac{dq}{dt} = \frac{q_\infty(V) - q}{\tau_q(V)}, \quad q = \{m_i, h_i, n\}
$$

$$
\begin{aligned}
I_{Na} &= g_{Na} m_{Na}^3 h_{Na}(V - E_{Na}), \quad I_K = g_K n^4 (V - E_K) \\
I_{Ca} &= g_{Ca} m_{Ca} h_{Ca}(V - E_{Ca}), \quad I_{L_{Na}} = g_{L_{Na}}(V - E_{Na}) \\
I_{L_K} &= g_{L_K}(V - E_K), \quad I_H = g_H m_H (V - E_H), \quad I_A = g_A m_A^3 h_A (V - E_K) \\
q_\infty(V) &= \frac{1}{2} + \frac{1}{2}\tanh\left(\frac{V - \nu_q}{d\nu_q}\right), \quad \tau_q(V) = \tau_{q0} + \tau_{q1}\left(1 - \tanh^2\left(\frac{V - \nu_q}{d\nu_q}\right)\right)
\end{aligned}
$$

where C is membrane capacitance, V is membrane potential, $I_{app}(t)$ is the applied current, I are ionic currents (transient sodium I_{Na}, non-inactivating potassium I_K,

transient calcium I_{Ca}, sodium leak $I_{L_{Na}}$, potassium leak I_{L_K}, hyperpolarization-activated potassium I_H, and transient potassium I_A, respectively), g are maximal conductances, E are reversal potentials, and q are gating variables with steady-state functions q_∞ and time constants τ_q. We will use the SCN model with experimental target data in Section 5.1.

2.4 CA1 Pyramidal Neuron Model

In mouse models of Alzheimer's disease, pyramidal neurons in the CA1 region of the hippocampus exhibit altered excitability properties [63]. Nowacki et al. [46] developed a conductance-based model of CA1 pyramidal neurons that includes the following ionic currents: two Na^+-currents, one transient (I_{Na_T}) and one persistent (I_{Na_P}); two Ca^{2+}-currents, one T-type (I_{Ca_T}) and one high-voltage activated (I_{Ca_H}); and three K^+-currents, delayed rectifier ($I_{K_{DR}}$), M-type (I_{K_M}), and leak (I_L). The dynamics of the membrane potential V and ionic gating variables x are governed by the following system of ordinary differential equations:

$$C\frac{dV}{dt} = I_{app} - I_{Na_T} - I_{Na_P} - I_{Ca_T} - I_{Ca_H} - I_{K_{DR}} - I_{K_M} - I_L - I_{K_H} \tag{2.4}$$

$$\frac{dx}{dt} = \frac{x_\infty - x}{\tau_x}$$

where:

$$I_{Na_T} = g_{Na_T} m^3_{Na_{T_\infty}} h_{Na_T}(V - E_{Na}), \quad I_{Na_P} = g_{Na_P} m_{Na_{P_\infty}}(V - E_{Na})$$

$$I_{Ca_T} = g_{Ca_T} m^2_{Ca_T} h_{Ca_T}(V - E_{Ca}), \quad I_{Ca_H} = g_{Ca_H} m^2_{Ca_H} h_{Ca_H}(V - E_{Ca})$$

$$I_{K_{DR}} = g_{K_{DR}} m_{K_{DR}} h_{K_{DR}}(V - E_K), \quad I_{K_M} = g_{K_M} m_{K_M}(V - E_K)$$

$$I_L = g_L(V - E_L), \quad I_H = g_H(p\, m_H + (1-p)\, n_H)(V - E_H)$$

and $x \in \{h_{Na_T}, m_{Ca_T}, h_{Ca_T}, m_{Ca_H}, h_{Ca_H}, m_{K_{DR}}, h_{K_{DR}}, m_{K_M}, m_H, n_H\}$.

The ionic currents I are described by Ohm's Law with maximal conductance parameters g and reversal potentials E. The steady-state activation and inactivation functions x_∞ for all gating variables, including m_{NaT} and m_{NaP}, are given in Boltzmann form:

$$x_\infty(V) = \frac{1}{1 + \exp\left(-\frac{V - V_x}{k_x}\right)}.$$

The time constants τ_x for all gating variables are fixed parameters, except for h_{NaT}, for which the time constant is a voltage-dependent function:

$$\tau_{h_{NaT}}(V) = 0.2 + 0.007 \exp\left(\exp\left(-(V - 40.6)/51.4\right)\right).$$

The parameter values for this model are provided in Appendix Table A.1. We use the CA1 model with synthetic target data in Section 4.3 and with experimental target data in Section 5.2.

2.5 Medium Spiny Neuron Network Model

Ponzi et al. [51] modeled the balance of synaptic excitation and inhibition in the striatum by constructing a 2500-cell network of medium spiny neurons (which comprise over 90% of cells in the striatum) using single-compartment Hodgkin-Huxley-type models. They were able to reproduce the interspike interval (ISI) distributions obtained from extracellular spike train recordings by fitting just two network parameters, the amount of net feedforward excitatory drive (g_E), and the strength of recurrent inhibition coming from the medium spiny neuron (MSN) collateral network (g_I). All simulations involve 2500 cells randomly linked through inhibitory synapses with a probability of 0.2, implying that each cell gets input from about 500 other cells. The model is described by the following equations:

$$
\begin{aligned}
C_m \frac{dV}{dt} &= -\sum_x I_x, \qquad I_x = g_x m_x^p h_x^q (V - V_x), \\
\frac{dq_x}{dt} &= \frac{q_x^\infty - q_x}{\tau_{q_x}}, \qquad q_x^\infty = \frac{1}{1 + exp(\frac{V - \theta_{q_x}}{k_{q_x}})}, \quad q \in \{m, h\}, \\
\frac{ds}{dt} &= aH(V)(1 - s) - bs, \qquad \tau_{q_x}^0 + \frac{\tau_{q_x}^1 - \tau_{q_x}^0}{exp(\frac{\phi - V}{\sigma_{q_x}^0}) + exp(\frac{\phi - V}{\sigma_{q_x}^1})}, \\
I_{syn} &= g_{syn} s(V - V_{cl}), \qquad I_{ex} = g_{ex}(V - V_{cat})
\end{aligned}
\tag{2.5}
$$

where $g_{syn} \in [0.001 g_I, 0.001 g_I + 0.001]$ is the maximal synaptic conductance, s is a voltage-dependent synaptic gating variable, and $g_{ex} \in [0.04381, 0.04381 + 0.002 g_E]$ is excitatory drive. $H(v)$ is a Heaviside function applied to the membrane potential of the presynaptic cell and is unity if the presynaptic cell spikes, otherwise it is zero. τ^0 and τ^1 are the minimal and maximal time constants for the gating variable respectively, θ and k are constants for determining the voltage-dependent steady-state value of the gating variable, ϕ, σ^0 and σ^1 are constants determining the time course of voltage-dependent gating time constants. V_{cat} is the cation reversal potential. Parameter values are based on [12]. We will use the MSN network model with synthetic target data in Section 4.4 and with experimental target data in Section 5.3.

CHAPTER 3

METHODS

This chapter provides background information on some of the methodologies that we will use in this dissertation, including differential evolution, Sobol sensitivity analysis, and the Metropolis-Hastings algorithm. We also introduce conditional Generative Adversarial Networks as a novel parameter inference tool within the DeepHM framework.

3.1 Global Optimization - Differential Evolution

Differential evolution is a type of stochastic global optimization and a population-based search technique first introduced by Storn and Price in 1997 [53, 62]. This method is comprised of four different steps, including *Initialization*, *Mutation*, *Crossover*, and *Selection*.

Initialization - a population of individuals, which is also known as the first generation or parents, is often defined by drawing a population randomly from a uniform distribution to explore the objective landscape. In this context, an individual is essentially an ordered set of parameters.

Mutation - a new vector (a.k.a donor vector) is introduced in this step by a linear combination of three random vectors from the current generation, but not the one which we are going to compare and substitute with in the next generation (i.e., target vector). This means at least four members are needed for applying this method.

Crossover - the recombination of the donor vector with the best member of the previous generation with a probability rate brings both exploration and exploitation to the search and helps the system escape local minima.

Selection - during the selection step, we have a new vector, which is a candidate for going to the new generation in place of the target vector if it satisfies certain conditions, and if it has a better score, [1] than the target vector. In this way, after each generation we might have better members, or in the worst-case scenario, all members could be the same as the previous generation without any change, but none of them are worse than the previous generation.

The Differential Evolution Algorithm that we used in this project is known as **classic DE** in [53] (see Figure 3.1). It is also referred to as DE/rand/1/bin which means the base vector is *randomly* selected and only *1* vector difference is added in the mutation part. Finally, the donated parameters in the crossover section follow a *binomial* distribution.

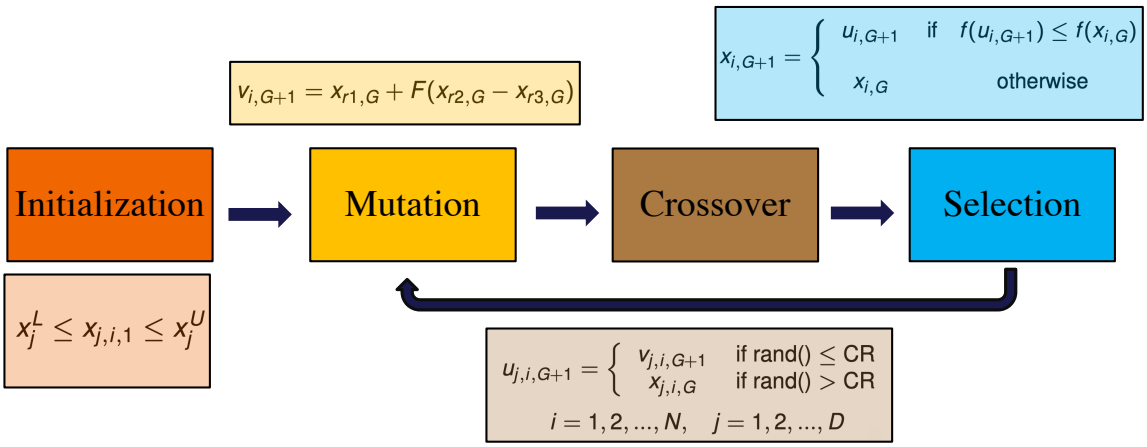

Figure 3.1 Schematic of the Differential Evolution Optimization. There are four different stages: (1) Initialization, (2) Mutation, (3) Crossover and (5) Selection. This algorithm iterates over steps (2) to (4) until it reaches certain criteria.

3.2 Global Sensitivity Analysis - Sobol Indices

Sensitivity Analysis (SA) [23, 28, 32, 61, 65] addresses the question of how much the output of a model changes given a change in the input. There are two main classes of

[1] The score could be the output of the objective function and depends on the minimization or maximization problem. The new candidate will either be rejected or accepted for the next generation.

sensitivity analysis, *local* and *global*. In local SA, the goal is to deal with sensitivity in the neighborhood of a particular parameter value. Taking a derivative is an example of local sensitivity analysis as it only considers the behavior of the system near a single point. Global SA, on the other hand, focuses on the variability of model outputs throughout parameter space. Global SA provides more information about the system and because of that it is often preferred, but if the system is too large the computational cost of this technique can be prohibitive. There are several different global SA approaches, including linear methods, tree-based methods, regionalized sensitivity analysis (also known as Monte Carlo filtering methods), and variance-based techniques [31, 36]. Sobol SA is a variance-based technique that considers the contribution of the input parameters to the variance of the system outputs. In Sobol SA we typically use two measures, the first order index and total effect index. The first order index is the contribution of an individual parameter to the response variance without considering any interactions with the other parameters. The total effect index is the contribution of an individual parameter to the response variance that does consider interactions with the other parameters.

Suppose we have a function of f that maps the vector of variables $X = (X_1, X_2, \cdots, X_p)$ to some quantity of interest. We assume X has some known probability distribution, which corresponds to certain parameters in the model. This probability distribution reflects the level of uncertainty in them. The goal is to determine the sensitivity of f to X, under the assumption that f is square integrable:

$$\begin{aligned} f &: \mathbb{R}^p \to \mathbb{R} \\ X &\mapsto f(X). \end{aligned}$$

If f is square integrable (i.e., $f(x) \in L^2$), we can decompose the above function as the sum of the functions of only $1, 2, \cdots, p$ parameters:

$$f(X) = f_0 + \sum_{i=1}^{p} f_i(X_i) + \sum_{1 \le i < j \le p} f_{i,j}(X_i, X_j) + \cdots + f_{1,2,\cdots,p}(X_1, X_2, \cdots, X_p) \quad (3.1)$$

where:

$$\begin{aligned} f_0 &= \mathbb{E}[f(X)] \\ f_i(X_i) &= \mathbb{E}[f(X) \| X_i] - f_0 \\ f_{i,j}(X_i, X_j) &= \mathbb{E}[f(X) \| X_i, X_j] - f_i(X_i) - f_j(X_j) - f_0 \end{aligned}$$

If $X_1, X_2, \cdots, X_p$ are statistically independent then all f satisfy the orthogonality property, which includes:

$$\begin{aligned} \mathrm{Var}(f(X)) &= \sum_{k=1}^{p} D_k(X_k) + \sum_{1 \le k \le k' \le p} D_{k,k'}(X_k, X_{k'}) + \cdots + D_{1,2,\cdots,p} \\ &= \sum_{u} D_u(X_u), \quad u \subset \{1, 2, \cdots, p\} \end{aligned}$$

where:

$$D_k(X_k) = \mathrm{Var}\left[f_k(X_k)\right], \quad D_{k,k'}(X_k, X_{k'}) = \mathrm{Var}\left[f_{k,k'}(X_k, X_{k'})\right]$$

$$D_0 = \mathrm{Var}\left[f_0\right] = 0, \quad \text{and} \quad u = \{i_1, i_2, \cdots, i_s\}, \quad 1 \le s \le p$$

This means the total variance of the response $f(X)$ can be written as the sum of partial variances. By defining the total variance of the response $f(X)$ that can be attributed to input parameter X_k as the ratio of $\mathrm{Var}\left[\mathbb{E}[f(X) \| X_k]\right] / \mathrm{Var}(f(X))$, the Sobol index for a subset $\mathbf{u}$ (i.e., $u \subset \{1, \cdots, p\}$) can be defined as the ratio between the contribution given by the interaction among the components of $\mathbf{u}$ for the model variance, and the total variance itself. Thus, the Sobol index for a subset $\mathbf{u}$ can be

written as:

$$S_u = \frac{D_u(X_u)}{\mathrm{Var}(f(X))}, \qquad \sum_{u \subset \{1,\cdots,p\}} S_u = \frac{\sum_u D_u(X_u)}{\mathrm{Var}(f(X))} = \frac{Var(f(X))}{\mathrm{Var}(f(X))} = 1. \tag{3.2}$$

As was mentioned before, in Sobol SA, two measures (i.e., the first order index and the total index) are usually computed. The first order index refers to the contribution of any one parameter to the output variance and can be defined as:

$$S_i = \frac{D_i(X_i)}{\mathrm{Var}(f(X))}, \quad i = 1, \cdots, p.$$

The total index refers to the contribution of all subsets with more than one parameter to the output variance, which means:

$$\begin{aligned} S_T &= \sum_{1 \le i \le j \le p} S_{i,j} + \cdots + S_{1,\cdots,p} = \sum_{u \subset \{1,\cdots,p\}} S_u \\ &= 1 - S_i, \quad i \in u. \end{aligned}$$

The first order index (i.e., S_i) describes the impact of X_i individually on the defined output, whereas the total index (i.e., S_T) represents the effect of X_i along with the interaction of other input variables. A high value for either of these two indices suggests that X_i, either alone or in conjunction with other input variables, has a considerable overall impact on the output space.

3.3 Parameter Inference Methodologies

3.3.1 Markov Chain Monte Carlo

The ***Monte Carlo*** method [19, 26] is a stochastic technique that aims to calculate numerical results from many random samples. In other words, if a method uses random numbers to solve a problem, that method is a type of Monte Carlo method. By employing this randomness, it is possible to address some problems that, in theory, may have deterministic solutions that are hard to obtain. In order to draw some

samples from a known distribution, this method repeatedly produces some random samples coming from that distribution. By repeating this process, eventually all the samples come from the desired distribution. One of the main issues with using a Monte Carlo method is that because it is a memoryless process, it can be time consuming to draw samples from a complex distribution. In other words, if one sample comes from a highly probable region of the distribution, there is no guarantee that the next candidate will also come from the same highly probable region. A ***Markov Chain*** is a sequence of events where the probability of the next event depends only on the present [8, 9]. By incorporating a Markov process into the Monte Carlo method, which is known as ***Markov Chain Monte Carlo (MCMC)***, information about the previous step affects the next proposal candidate, and the process of sampling from complex distributions is sped up. The Metropolis Hastings algorithm (Algorithm 1) is a particular MCMC method that performs a random walk through the probability distribution to try to evaluate the regions with higher probability. The main idea behind Metropolis-Hastings is that by running this algorithm for a sufficient number of iterations, the random walk procedure helps the Monte Carlo method to draw samples from the stationary distribution, which is the posterior distribution or the target distribution. The key point here is that based on this algorithm, the acceptance probability that comes from the density of the model output is computed using a Gaussian mixture model that is fit to the target dataset.

3.3.2 Generative Adversarial Networks

Generative Adversarial Networks (GANs) are an example of generative models in machine learning. Since GANs have a deep neural network architecture we can classify them as deep learning models. The application we are interested in here is to build an inverse surrogate model for mapping the output of a mechanistic model into its corresponding region in parameter space. More precisely, the goal

Algorithm 1 Markov Chain Monte Carlo method - Metropolis Hastings

Define $f(x)$ as the target distribution, N as the total number of iterations, x_i as the current value, and $q(x|x_i)$ as the proposal distribution. This proposal distribution can be symmetric or asymmetric in this method. Choose x_0 randomly from its defined range (i.e., $x_0 = x_l + rand(0,1) \times (x_u - x_l)$).

1. **while** Iter $<$ N **do**
2. $x^\star \sim q(x|x_i)$, proposed candidate
3. $\rho = min\left\{1, \frac{f(x^\star)q(x_i|x^\star)}{f(x_i)q(x^\star|x_i)}\right\}$, $u \sim U(0,1)$,
4. **if** $u < \rho$
5. $x_{i+1} = x^\star$
6. **else**
7. $x_{i+1} = x_i$
8. **end**
9. Iter = Iter + 1
10. **end**

is to map the density of observed data (i.e., $\mathcal{P}_Y$) to a *coherent* density α_X of the model input parameter space. A distribution α_X is coherent if: (1) upon sampling from α_X and applying the mechanistic model, the estimated density in output space satisfies $\hat{\mathcal{P}}_Y \sim \mathcal{P}_Y$, and (2) α_X covers all possible solutions in the range described by the prior $\mathcal{P}_X$. In this section we will first introduce standard GANs, and then move on to conditional GANs (cGANs) which are designed to incorporate conditional distributions into GANs.

The basic GAN consists of two artificial neural networks, a generator G and a discriminator D, that compete with each other (Figure 3.2). The Generator tries to produce fake samples that are as close as possible to real samples that come from some distribution, and the Discriminator tries to distinguish real samples from fake samples.

Training the GAN is an iterative process through which G gets better at fooling D, and D gets better at identifying fake samples. The first step in training the GAN is to create a training dataset (referred to as real samples) by simulating the mechanistic model with parameter sets x drawn from a uniform distribution $P_{data}(x)$. Then we initialize the Generator with a base distribution $P_z(z)$ which is a random variable with typically a Gaussian distribution. These parameter sets $G(z) \sim P_G$, referred to as fake samples, are passed to the Discriminator along with the real samples. For a given sample x or $G(z)$, the Discriminator outputs a probability $\hat{y} = D(x)$ or $\hat{y} = D(G(z))$, referred to as a reconstructed label, indicating whether it thinks the sample is real ($\hat{y} > 0.5$) or fake ($\hat{y} < 0.5$). If D is correct (i.e., $\hat{y} = D(x) > 0.5$ or $\hat{y} = D(G(z)) < 0.5$), then the weights and biases (ω, β) of the D network will remain fixed, but (ω, β) of the G network will be adjusted through backpropagation (referred to as fine-tuning in Figure 3.2). If D is incorrect (i.e., $\hat{y} = D(x) < 0.5$ or $\hat{y} = D(G(z)) > 0.5$), then (ω, β) of D are adjusted while (ω, β) of G remain fixed. In practice, convergence of generator and discriminator one at a time not only would be time consuming but also lead to instability due to a vanishing gradient for the generator. Therefore, in this case the weights are adjusted after computing the loss function from the outputs of D and G over each mini-batch. As a result, both the generator and the discriminator are being trained simultaneously, and they are converging gradually.

Derivation of the objective function for the GAN *Cross-entropy* is a measure from the field of information theory which calculates the difference between two probability distributions and is defined by the following equation:

$$L(\hat{y}, y) = [y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \tag{3.3}$$

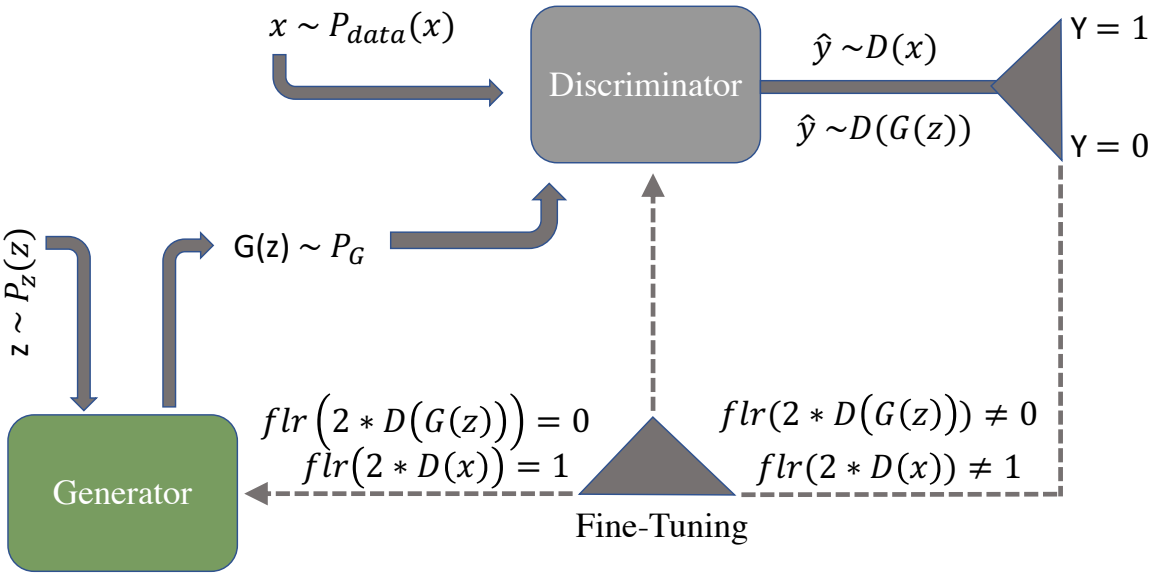

Figure 3.2 Schematic of a generative adversarial network (GAN). The Generator (G) and Discriminator (D) are two neural networks that compete with each other during the training process, with the end result being that G can produce fake samples from a distribution that matches the distribution of the real samples. $D(x)$ and $D(G(z))$ provide the reconstructed label $\hat{y}$ (i.e., the output of the Discriminator network), which represents the probability of the sample being real rather than fake. If $\hat{y}$ is greater than (less than) 0.5, the Discriminator classifies the sample as real (fake). If D is correct (incorrect), then the weights of G (D) are fine-tuned through backpropagation.

where $\hat{y}$ is the reconstructed label $D(x)$ or $D(G(z))$ and y is the actual label for the sample. Therefore, the corresponding label for a real sample (i.e., a sample coming from $P_{data}(x)$) is $y = 1$ and the reconstructed label (i.e., the output of the Discriminator) is $\hat{y} = D(x)$. By substituting these labels for the real samples into the equation (3.3) we can get:

$$L(D(x), 1) = \log(D(x)) \tag{3.4}$$

Likewise the data coming from the Generator has the real label $y = 0$ and the reconstructed label is $\hat{y} = D(G(z))$, therefore, by substituting these expressions for the fake samples into Equation (3.3) we end up with:

$$L(D(G(z)), 0) = \log(1 - D(G(z))). \tag{3.5}$$

Panels A1 and A2 in Figure 3.3 are the visualization of Equations. (3.4) and (3.5), respectively. Since the output of the Discriminator is a probability, in both of these panels we only consider the region between zero and one on the $x-$axis (which represents $D(x)$ or $D(G(z))$). In panel A1, $\log(D(x))$ is an increasing function of $D(x)$. If we have a strong Discriminator, then we expect $D(x) \sim 1$ for a real sample. In panel A2, the $x-$axis represents $D(G(z))$, and the expectation for a strong Discriminator would be $D(G(z)) \sim 0$ for a real sample. As these two points are close to the maximum of both Equations. (3.4) and (3.5), the objective function for the Discriminator is:

$$\max_D \{\log D(x) + \log(1 - D(G(z)))\}. \tag{3.6}$$

The same logic also holds in the case of having a strong Generator, the only difference here is that we are going to minimize Equation (3.5), which corresponds to the right panel of Figure 3.3. Having a very strong Generator means it is able to fool the Discriminator easily. This means the Discriminator will erroneously return a high probability even for a fake sample, i.e., $D(G(z)) \sim 1$. This point is the minimum value of Equation (3.5). Thus, the objective function for the Generator is:

$$\min_G \{\log(1 - D(G(z)))\} = \min_G \{\log D(x) + \log(1 - D(G(z)))\}. \tag{3.7}$$

In order to obtain a single objective function for the GAN, we combine Equations (3.6) and (3.7) and take the expectation over the whole dataset. The resulting objective function for the GAN, which is inspired by the cross-entropy loss, is given by:

$$\min_G \max_D \{\mathbb{E}_{x \sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim P_z(z)}[\log(1 - D(G(z)))]\}. \tag{3.8}$$

In practice, at the beginning of the training process the Generator is not strong enough and the output of the Generator is very different from the training dataset. Thus,

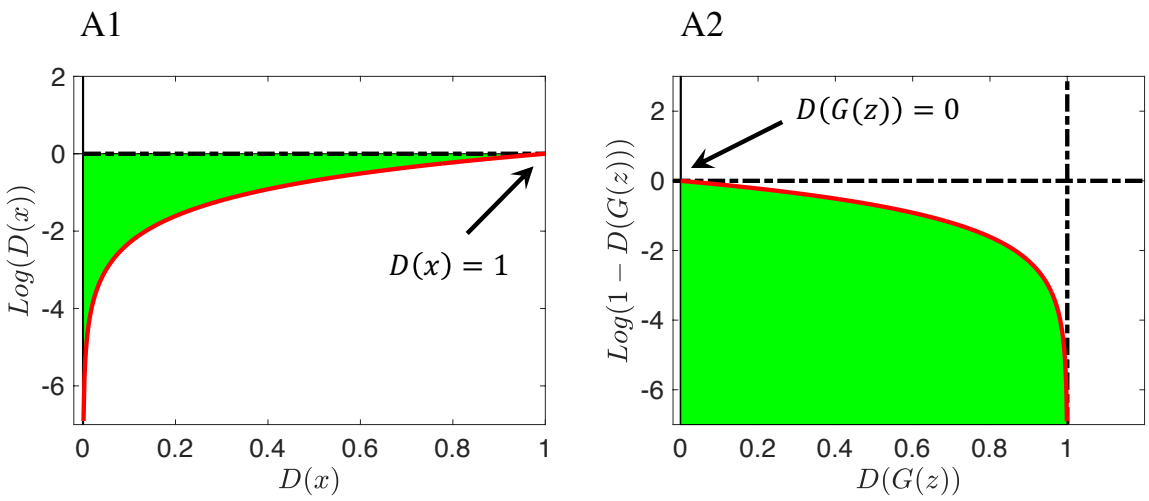

Figure 3.3 Visualization of the components of the GAN objective function (Equation (3.8)). (A1) Discriminator loss function (Equation (3.4)). $D(x)$ is the probability that x came from the real data rather than P_g. (A2) Generator loss function (Equation (3.5)). $D(G(z))$ is the probability that $G(z)$ came from P_g rather than P_{data}. The green shading identifies the possible regions for these two probabilities.

the Discriminator can easily distinguish the real and the fake samples. This causes $\log(1 - D(G(z)))$ to saturate (see Figure 3.3A2), and the gradient does not provide any information as it is almost zero. Therefore, as is mentioned in [21], instead of minimizing $\log(1 - D(G(z)))$ we can minimize the $\log(1 - D(G(z))) - \log(D(G(z)))$. This will help ensure we have a more stable loss function during the training process.

3.3.3 Conditional Generative Adversarial Networks

A standard GAN could be trained to produce samples of parameter sets, samples of feature sets, or even samples of combined parameter-feature sets. However, it is not able to produce samples of parameter sets corresponding to a set of given feature values. To accomplish this, we employ conditional GANs [40], where features extracted from the output of the mechanistic model are passed as a condition to the Generator. The parameter samples produced by the Generator, augmented with the features it was provided as a condition, are then passed to the Discriminator. Ground

truth parameters, with their corresponding features, as a condition, are also passed to the Discriminator. During the training process, the Generator learns how to produce samples in parameter space that are similar to the ground truth parameters for a given set of features.

The overall structure of the cGAN is similar to the basic GAN model. However, the main difference is that the input into both the Generator (G) and Discriminator (D) are augmented by the conditional variable Y as described in Figure 3.4. The objective function of the cGAN is:

$$\begin{cases} \min\limits_{D} & \{-\mathbb{E}_{x\sim P_{data}(x)}[\log D(x|y)] - \mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z|y)))]\} \\ \min\limits_{G} & \{\mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z|y)))] - \mathbb{E}_{z\sim P_z(z)}[\log(D(G(z|y)))]\} \end{cases} \tag{3.9}$$

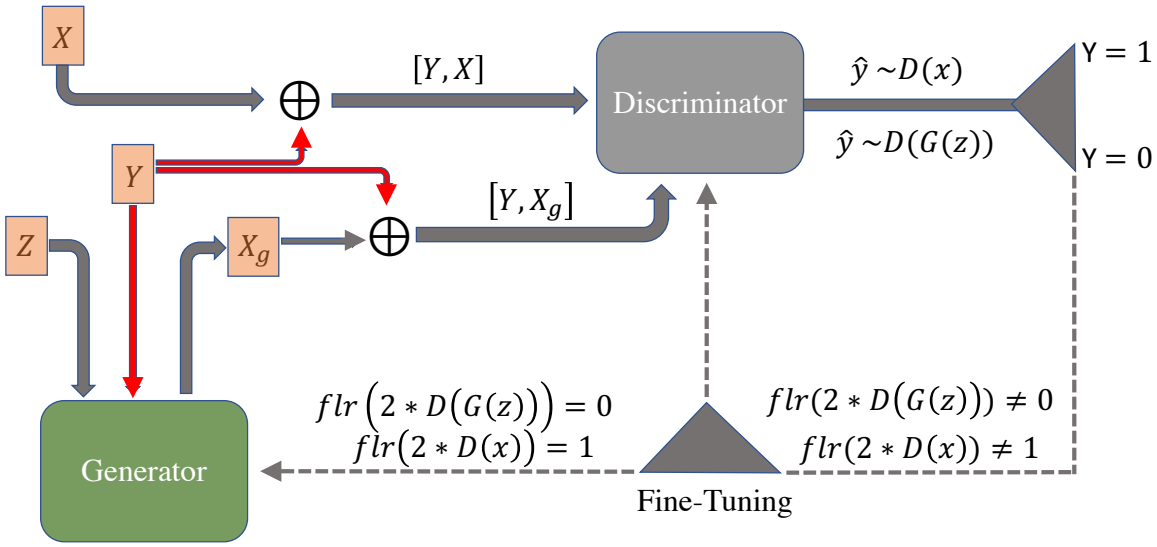

Figure 3.4 Schematic of a conditional GAN (cGAN). Features of the training dataset (Y) are passed as a condition into the Generator (G) - already initialized with a Gaussian distribution (Z) - in order to produce some samples X_g given that condition $[Y, X_g]$. This output, along with real samples X augmented with their output features $[Y, X]$, are passed into the Discriminator (D). If the Discriminator's output is close to zero (one), it means the Discriminator has assigned a low (high) probability of that sample being real.

Illustration of cGAN training process

We used the Rosenbrock function (Equation 3.10) as a toy mechanistic model to illustrate the training process for cGAN:

$$Y = (1 - X_1)^2 + 100(X_2 - X_1^2)^2. \tag{3.10}$$

In this example, we have only two input parameters (X_1 and X_2) and only one feature (Y, the output of the Rosenbrock function). Thus, with a fixed output as a condition, cGAN converges to the ground truth region in the parameter space after a few training epochs. Figure 3.5 provides a visual summary of the training. We used Jensen Shannon Divergence (JSD), a measure of the similarity between two probability distributions, as our stopping criterion:

$$\mathrm{JSD}(p\|q) = \frac{1}{2}\left\{\int p(x)\log\left(\frac{p(x)}{M(x)}\right)dx + \int q(x)\log\left(\frac{q(x)}{M(x)}\right)dx\right\},$$

where

$$M = \frac{p+q}{2}.$$

When the JSD measure for a fixed epoch number approaches zero, then we stop training after that epoch. If one continues training after this stopping point, the training process destabilizes and the JSD measure starts increasing (Figure 3.5 bottom right panel). This phenomenon occurs due to the vanishing gradient problem[2].

[2]During the process of training, all the weights and biases of the neural network are updated through the process of backpropagation, which is a gradient based technique. This gradient sometimes gets smaller and smaller due to the multiplication of small values by each other through the chain rule. This might slow down the neural network's training process and have a negative impact it, which is known as the vanishing gradient problem.

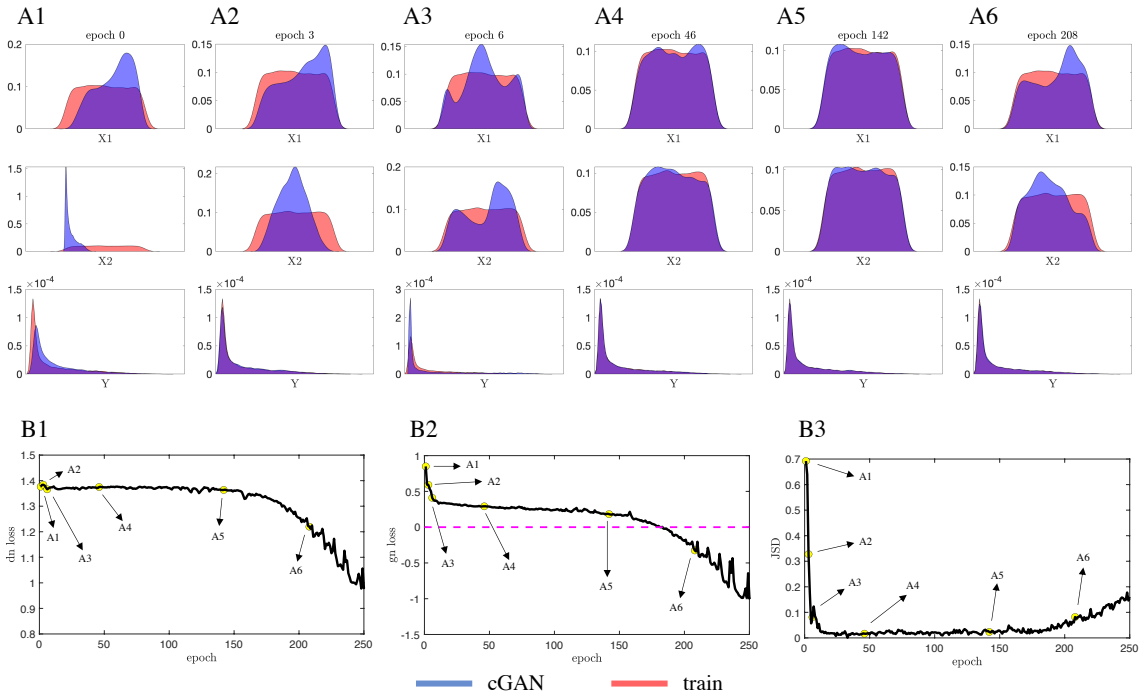

Figure 3.5 Illustration of the cGAN training process on the Rosenbrock function (Equation (3.10)). Over the course of the training process, both the discriminator and the generator improve. The discriminator gets better at distinguishing between real (coming from P_{data}) and fake (coming from P_g) samples, and for a fixed discriminator, the generator gets better at fooling the discriminator. (A1-A6) KDE plots of the cGAN samples (blue) and the training data (red) for parameters $X1$, $X2$, and feature Y at epochs 0, 3, 6, 46, 142, and 208. At epoch 142 (A5), the cGAN distributions are good approximations of the training distributions, but by epoch 208 (A6) the cGAN distributions are no longer good approximations. (B1-B3) The discriminator loss (B1), the generator loss (B2), and the JSD measure between the ground truth parameters versus estimated parameters (B3) as a function of epoch number. The point labeled A5 in panel B3 represents the JSD stopping criterion: once the JSD starts to increase we stop training the cGAN.

CHAPTER 4

PARAMETER INFERENCE ON SYNTHETIC TARGET DATA

In this chapter, we will validate our parameter inference approach by employing it on synthetic target datasets where the ground truth parameter distributions are known. We perform these parameter inference tests with four different mechanistic models corresponding to distinct biological questions about types of excitability, circadian rhythms, and the effect of mutations on single-cell and network activity. We also compare the performance of cGAN to a baseline MCMC method.

4.1 Morris-Lecar Model - Type I and Type II Excitability

The difference between type I and type II excitability is illustrated by the experimental recordings shown in Figure 4.1. A layer 5 pyramidal cell (panel A1) goes from silent at $I = 20$ to repetitive firing at a very slow rate at $I = 40$ to firing at a very high rate at $I = 320$. On the other hand, a brainstem mesV cell (panel A2) goes from silent at $I = 500$ (aside from a few transient spikes at the start of the stimulus) to firing at a high rate at $I = 600$, but then doesn't fire all that much faster at $I = 1000$. Panels B1 and B2 plot the firing frequency vs input (F-I) curve for these two cell types. The main difference is that the F-I curve for the pyramidal cell (panel B1) is approximately continuous with firing arbitrarily low rates, whereas the F-I curve for the mesV cell (panel B2) shows a clear discontinuity when the firing rate jumps up from zero. These properties of the F-I curve are characteristic of type I and type II excitability, respectively.

These different types of excitability can be exhibited by different parameter regimes of the Morris-Lecar model (see Section 2.2). With the parameter values in the first column of Table 4.1 (labeled Hopf), the model exhibits type II excitability; with the parameter values in the second column (labeled SNIC), the model exhibits

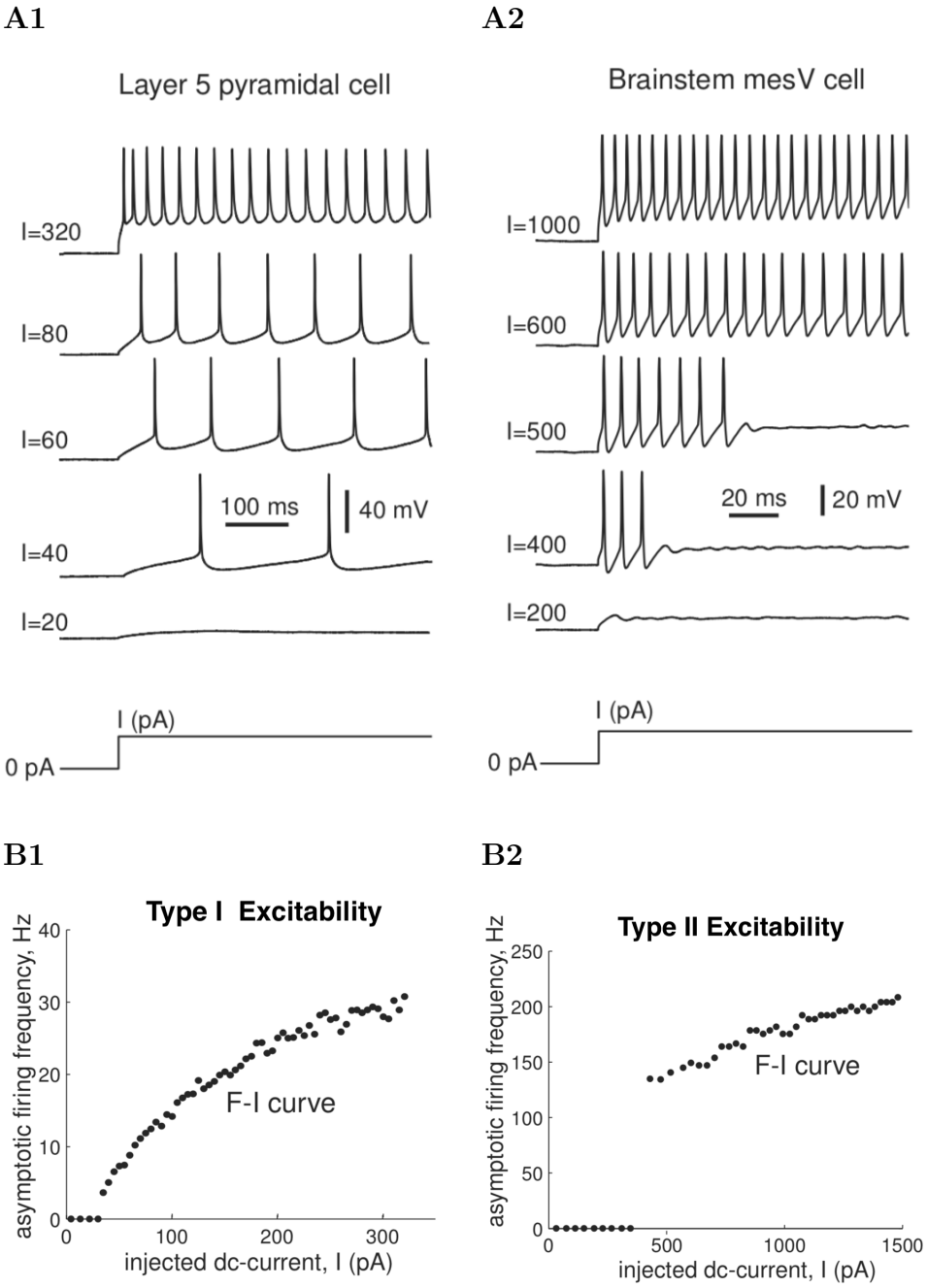

Figure 4.1 Panels A1 and B1 - type I excitability class. Panels A2 and B2 - type II excitability class. The F-I curve in panel B1 is continuous whereas there is a clear discontinuity in panel B2. Figure modified from [29].

type I excitability. Note that one can go from type II to type I behavior by changing just three parameters (i.e., ϕ, v_3 and v_4). The qualitatively different behavior of the model in these parameter regimes is due to the different bifurcations underlying the

transition from silent to spiking, i.e., a Hopf bifurcation results in type II excitability whereas a SNIC (saddle node on invariant circle) bifurcation results in type I.

Table 4.1 Morris-Lecar Parameter Values

	Hopf	SNIC
ϕ	0.04	0.067
g_{Ca}	4	4
V_3	2	12
V_4	30	17.4
g_K	8	8
g_L	2	2
V_1	-1.2	-1.2
V_2	18	18

For all simulations, $C = 20$, $E_{Ca} = 120$, $E_K = -84$ and $E_L = -60$.

4.1.1 Parameter Estimation with cGAN for Type I and Type II Excitability Classes

We will use the Morris-Lecar model with randomly distributed parameters to create a synthetic dataset. We will train a cGAN on this dataset to learn the parameter distributions that correspond to type I and type II excitability. The benefit of this approach is that the trained cGAN, since it is a generative model, can then be used to generate samples (parameter sets) that produce each type of excitability in order to populate a conductance-based network model with parameter heterogeneity and the desired fraction of type I vs type II behavior.

To create the synthetic dataset we first need to choose random parameter values. Here, for each parameter we simply used a uniform distribution over an interval with

a lower bound of 0 and an upper bound of two times the mean of the values shown in Table 4.1 (except for V_1, which has an upper bound of 0 and a lower bound of two times its mean value). Alternatively, we could use Latin Hypercube Sampling to get samples more representative of the multidimensional parameter space. For each parameter set, we then simulate the Morris-Lecar model over a range of I_{app} values and compute an F-I curve. Some models do not fire at all, so these samples are classified as silent. For models that do fire, if the F-I curve is sufficiently discontinuous we classify the sample as type I; otherwise it is classified as type II. To make this classification accurately, we ran at least 1000 simulations for each sample, first searching over a coarse grid of I_{app} values and then a refined grid near the minimal I_{app} value for which the model begins to spike; see Algorithm 2 for a detailed description of our method.

The excitability class (type I, type II, or silent), which we refer to as a "feature" or "label", is appended to the parameter set for each sample using one-hot encoding in order to create the synthetic dataset that is used to train the cGAN.

4.1.2 Case 1: cGAN Results Varying 3 Parameters

In the first case we consider, only three parameters of the Morris-Lecar model (ϕ, V_3 and V_4) are chosen randomly while the other 5 parameters are held constant. The rationale for considering these three parameters alone is that it is already known (as shown in Table 4.1) that varying these three is sufficient to produce both types of excitability.

The training dataset is shown in Figure 4.2A. Along the diagonal are kernel density estimate (KDE) plots showing the distribution of each parameter for type I (red) and Type II (blue) samples. Off the diagonal are scatter plots showing pairwise relationships between the parameters and excitability type. Once the cGAN is trained, we now have an inverse surrogate model. This means we can provide a label to the cGAN, e.g., type I, and ask it to provide parameter sets that will produce type I behavior when fed to the Morris-Lecar model. Here, the trained cGAN was

Algorithm 2 Method for Classifying Samples as Type I or Type II

1. Create a large number of random samples, $\mathcal{P}$, based on a uniform distribution for each parameter $x \in [x_{low}, x_{high}]$ where $x_{low} = x_{mean} - (100\% \{x_{mean}\})$ and $x_{high} = x_{mean} + (100\% \{x_{mean}\})$ where $x_{mean} = (x \in Hopf + x \in SNIC)/2$

for $iteration_1$ **in** length($\mathcal{P}$) **do**

2. Find the first I_{app} value over a coarse grid for which the model starts firing for Morris-Lecar($\mathcal{P}(iteration_1)$) and save that index into I_{First}. This search is pretty fast but not accurate, as initially we are just trying to get the approximate location of the first spike.

3. Create new finer search interval, linspace($I_{First} - 1$, I_{First}, $ITERATION$), where $ITERATION$ is a high number. In our case 1000 seems to be accurate enough. Although increasing this number will increase the accuracy, the simulation time will increase dramatically. Thus we need to find a trade off between accuracy and simulation time.

for $iteration_2$ **in** range($ITERATION$) **do**

$\mathcal{F} \leftarrow$ FR(Morris $-$ Lecar($\mathcal{P}(iteration_1)_{I_{app_{(iteration_2)}}}$)) (FR is the firing rate)

end for loop

4. $Discontinuity \leftarrow$ Diff ($\mathcal{F}$)

if max($Discontinuity$) $>$ 1 **do**

Classify $\mathcal{P}(iteration_1)$ as **type II**

else

Classify $\mathcal{P}(iteration_1)$ as **type I**

end if loop

end for loop

asked to generate 4,000 type I parameter sets and 4,000 type II parameter sets. It was able to do so quite accurately for type I excitability, with 98.37% of the cGAN type I parameter sets actually having type I excitability when passed through the Morris-Lecar model. The accuracy was quite high for type II cGAN samples as well, at 96.82%. Figure 4.2B shows KDE and scatter plots for the samples produced by the trained cGAN for type I (orange) and type II (purple). Comparing panels A and B, we see that the parameter distributions produced by the cGAN match very well the parameter distributions in the training dataset. For example, in both cases there is quite a bit of overlap in the parameter ϕ for type I and type II, with type I skewed to the right. Furthermore, for V_3, type II is skewed left and type I is skewed to the right, whereas for V_4 type I is skewed left and type II is skewed right. The excellent agreement between the samples produced by the cGAN and the training dataset is even more evident in Figure 4.3, where both the training and cGAN distributions for type I (panel A) and type II (panel B) are plotted on the same set of axes.

In Figure 4.4, we show the distribution of all classes (i.e., type I, type II and silent for both training (panel A) and cGAN samples (panel B) together. The main point here is that by providing the silent class information in our training dataset, cGAN is not only able to produce some samples from their corresponding region in parameter space with high accuracy, but also avoid going into that region when asked to provide samples for type I and type II excitability classes.

4.1.3 Case 2: cGAN Results Varying 8 Parameters

This case is similar to the previous case but instead of varying just three parameters we randomly vary all eight parameters shown in Table 4.1. The main purpose of this case is to see if the cGAN is still successful inferring parameter distributions in a higher-dimensional space, and also to learn if other parameters besides ϕ, V_3,

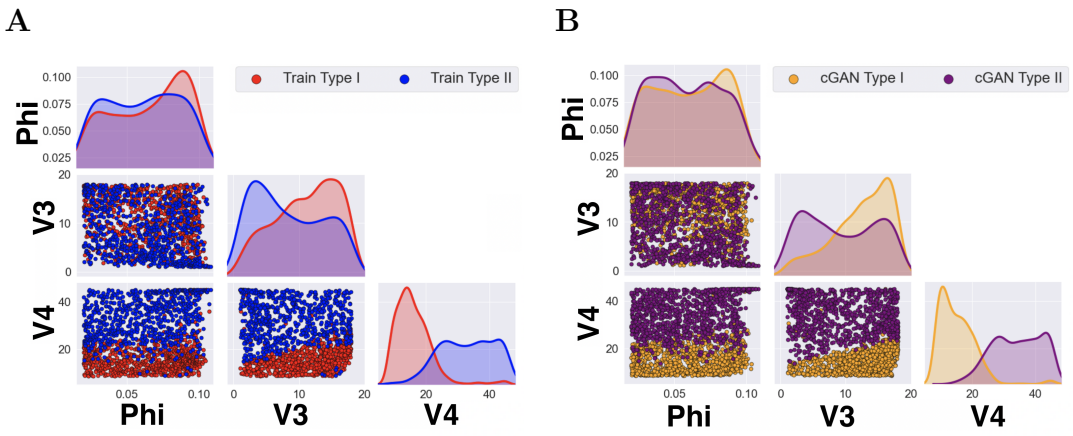

Figure 4.2 cGAN inference of type I and type II classes varying three parameters of the Morris-Lecar model. (A) Training dataset containing 40,000 different parameter sets, out of this number 12,979 of them are type I class (red) and 26,300 are type II (blue). The 721 training samples that are silent are not shown. (B) 4,000 samples produced by the trained cGAN samples for type I (orange) and type II (purple) excitability.

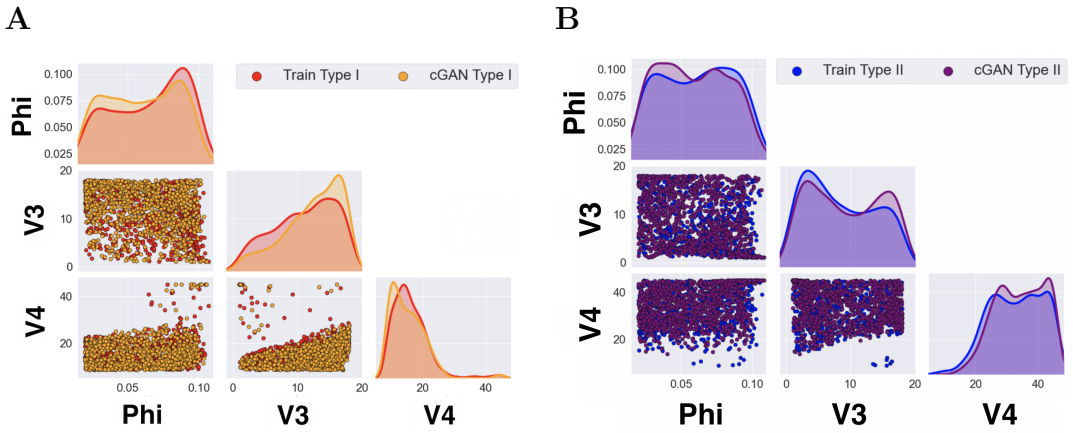

Figure 4.3 Replotting cGAN inference of type I and type II classes varying three parameters of the Morris-Lecar model. Same data as **Figure 4.2**, but with type I training and cGAN samples (panel A) and type II training and cGAN samples (panel B) plotted on top of each other.

and V_4 can change the type of excitability exhibited by the Morris-Lecar model. The results are shown in Figure 4.5. The parameter distributions for type I (red), type II (blue), and silent (green) are shown for the training dataset in panel A and for the cGAN samples in panel B. We see that the distribution of V_2 is skewed to

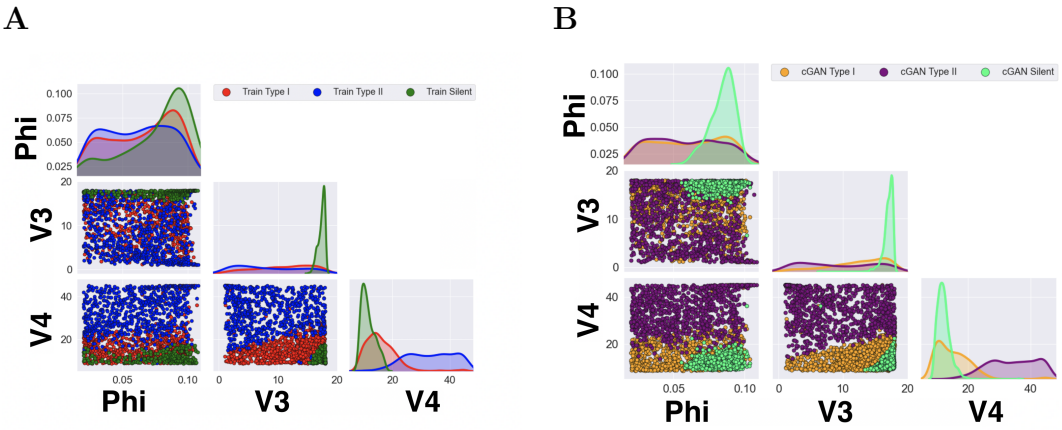

Figure 4.4 cGAN inference of type I, type II, and silent classes varying three parameters of Morris-Lecar model. Same data as **Figure 4.2**, but now the training samples for silent cells (green, panel A) and the cGAN samples for silent cells (light green, panel B) are also shown.

the left for type I and to the right for type II in the training dataset, and this asymmetry is captured well among the cGAN samples. However, the distributions of other parameters, such as V_4, in the cGAN samples do not match up as well with the training data distributions. Also, the classification accuracy of the cGAN samples for the 8-parameter case was lower than what it was for the 3-parameter case—here we have 85.3% accuracy for type I and 80.6% for type II. The lower accuracy here is not surprising, since we have increased the dimension of the parameter space from 3 to 8 but we have not increased the number of training samples accordingly. Thus, the samples are distributed more coarsely throughout parameter space which negatively affects the training of the cGAN. To resolve this issue, we could increase the number of samples. It is also possible that the distribution of V_4 in the cGAN samples is an example of mode collapse due to the vanishing gradient problem. In other words, sometimes during the process of training, the generator only produces limited varieties of samples (mode collapse) which leads to having a very strong discriminator and makes the generator's gradient vanish. Thus, the generator won't be able to learn anything new in each iteration and will converge to a distribution covering a

small portion of samples. If so, we could try to resolve the mode collapse issue by using spectral normalization (a novel weight normalization technique to stabilize the training of the discriminator [41]). Finally, we note that for the 8-parameter case we used a balanced distribution in feature space (10,000 samples of each excitability class) which results in a non-uniform distribution in parameter space. Perhaps better results could be obtained using a uniform distribution in parameter space.

4.2 Suprachiasmatic Nucleus Neurons - Day/Night Variation

Circadian rhythms are oscillations in physiology and behavior with a period of ∼24 hours, such as the sleep-wake cycle. There are two key properties that have to be satisfied for a rhythm to be considered circadian, the first one is that it needs to persist without any external time cues (i.e., it is internally generated) and the second one is that it can synchronize to external periodic stimuli. These rhythms are controlled by a group of ∼20,000 neurons in the hypothalamus called the suprachiasmatic nucleus (SCN). SCN neurons possess an internal molecular clock that produces 24-h oscillations in gene expression and membrane excitability. Conductance-based models of SCN excitability have been developed in prior work [5,14,15,17]. Most of what we know about SCN electrical activity comes from studies of mice and other nocturnal (night-active) rodents. This hinders the translation of this knowledge to humans, a diurnal (day-active) species. Recently, Mino Belle and his colleagues at the University of Manchester [47] made the first single-cell recordings from SCN neurons of a diurnal rodent, *Rhabdomys pumilio*. Overall, these recordings revealed that diurnal SCN neurons exhibit the same type of day/night rhythms in resting membrane potential (RMP) and firing rate as nocturnal SCN neurons—they have a more depolarized RMP and fire at a higher rate during the day than at night (see Figure 4.6). Thus, we are motivated to assess what ionic conductances are responsible for the day/night differences observed in Rhabdomys SCN electrical activity.

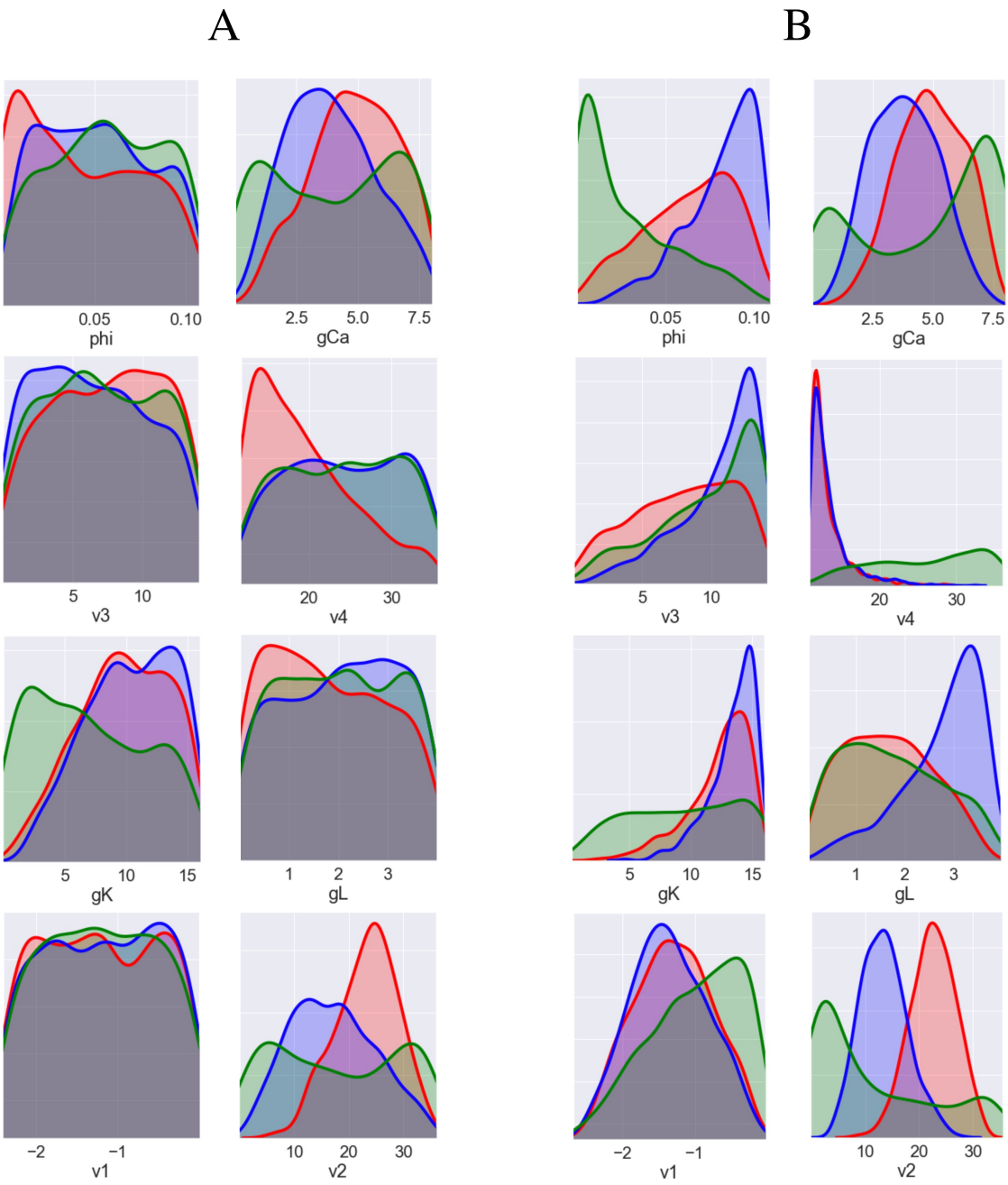

Figure 4.5 cGAN inference of type I (red), type II (blue), and silent (green) classes varying eight parameters of Morris-Lecar model. Column A: KDE plots for the training dataset containing 30,000 different parameter sets, of which 10,000 are type I, 10,000 are type II, and 10,000 are silent. Column B: KDE plots for 4,000 cGAN samples produced for each excitability class.

To validate the ability of the cGAN framework to perform this type of inference, we created a synthetic circadian datasets by simulating the Hodgkin-Huxley (HH)

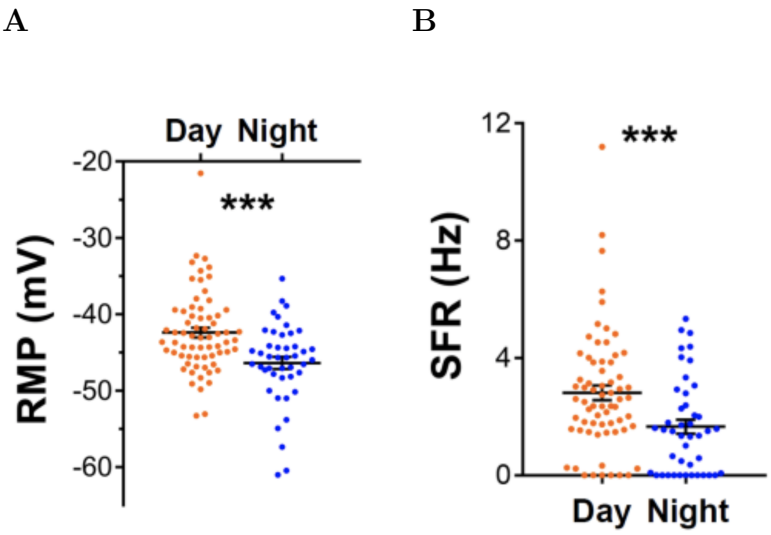

Figure 4.6 Day/night changes in the spontaneous electrical activity of Rhabdomys pumilio SCN neurons. (A) resting membrane potential (RMP), (B) spontaneous firing rate (SFR). Figure modified from [47].

model (see Section 2.1) with some of the parameters systematically varied to produce a bimodal distribution in parameter space to represent an artificial "day/night" difference. Then we trained a cGAN to see if it can build an inverse surrogate model to infer the underlying parameter distributions based on two features, firing rate and RMP.

Since it is known that sodium leak currents play a role in regulating the day/night activity of SCN neurons [17], our first step was to modify the HH model separating out the leak current I_L into a sodium leak current I_{NaL} and a potassium leak current I_{KL}. To do this, we solve the following set of two equations with two unknowns to obtain unique solutions for the default values of g_{NaL} and g_{K_L}:

$$I_L = g_L(V - E_L) \Longrightarrow I_L = I_{Na_L} + I_{K_L}$$

where:

$$I_{Na_L} = g_{Na_L}(V - E_{Na}) \quad and \quad I_{K_L} = g_{K_L}(V - E_K)$$

$$\begin{cases} g_L(V - E_L) & = g_{Na_L}(V - E_{Na}) + g_{K_L}(V - E_K) \\ g_L & = g_{Na_L} + g_{K_L} \end{cases} \tag{4.1}$$

There are now four maximal conductances in our modified HH model: g_{Na}, g_K, g_{Na_L}, and g_{K_L}. We will assume that g_{Na} and g_K do not have day/night variation but that g_{Na_L} and g_{K_L} do. Specifically, we assume that g_{Na_L} is higher during the day and lower at night, whereas g_{K_L} is lower during the day and higher at night. To create a synthetic training dataset, we draw parameter values from uniform distributions. For both the day and night simulations, the lower and upper bounds for g_{Na} and g_K are $\pm 100\%$ of their default value. For the day simulations, the lower bound of g_{Na_L} is its default value and the upper bound is $+100\%$ of that value; for g_{K_L} the lower bound is 0 and the upper bound is its default value. On the other hand, for the night simulations, the lower bound of g_{K_L} is its default value and the upper bound is $+100\%$ of that value; for g_{Na_L} the lower bound is 0 and the upper bound is its default value.

The next step is to pass these parameters to the HH model and extract the features for 2500 "day" training samples and 2500 "night" training samples. Figure 4.7A shows the distributions of parameter values, firing rate, and RMP for our synthetic training dataset. We see that, as we hoped, the day samples (orange) have a higher firing rate and a more depolarized RMP than the night samples (green). Next, we trained the cGAN, and then asked it provide 1000 "day" samples and 1000 "night" samples. Figure 4.7B shows that the distributions of the cGAN day samples (purple) and the cGAN night samples (blue) agree very well with the distributions for the training data. This indicates that the cGAN was able to find the correct region of parameter space corresponding to day and night neuronal activity through the mapping process, and gives us confidence that a cGAN will provide useful information

when trained on the single-cell SCN model and provided experimental data from Rhabdomys SCN neurons in Section 5.1.

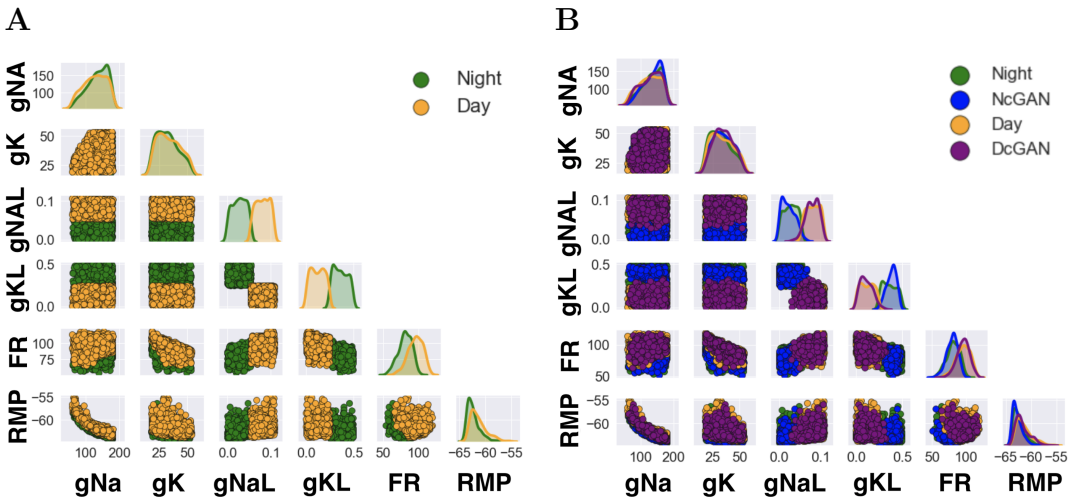

Figure 4.7 cGAN results for Hodgkin-Huxley model with bimodal parameter distribution representing day/night circadian variation. (A) Parameter-feature training dataset which is created by passing day (orange) and night (green) parameter sets into the HH model and extracting their corresponding feature values, firing rate (FR), and resting membrane potential (RMP). (B) Samples produced by the trained cGAN and passed to the HH model for day (purple) and night (blue).

4.3 CA1 Pyramidal Neuron Model - Disease Effect

4.3.1 Overview

Alzheimer's disease (AD) is believed to occur when abnormal amounts of the proteins amyloid beta and tau aggregate in the brain, resulting in a progressive loss of neuronal function. Hippocampal neurons in transgenic mice with amyloidopathy or tauopathy exhibit altered intrinsic excitability properties. Our goal here is to combine deep learning with biophysical modeling to map experimental data recorded from hippocampal CA1 neurons in transgenic AD mice and age-matched wild-type littermate controls to the parameter space of a conductance-based CA1 model. In this section, we demonstrate that cGANs can accurately infer parameter distributions of

the conductance-based model and outperforms a Markov chain Monte Carlo method on several test cases using synthetic data.

4.3.2 Experimental Data and Feature Extraction

The experimental data we use consists of patch-clamp recordings made from hippocampal CA1 neurons associated with two previous studies involving mouse models of Alzheimer's disease. In Tamagnini et al. [63], CA1 current-clamp recordings were obtained from transgenic PDAPP mice exhibiting amyloidopathy. In unpublished data, Tamagnini et al. obtained CA1 current-clamp recordings from transgenic rTg4510 mice exhibiting tauopathy. In this dissertation, we use voltage traces from $n = 30$ cells of 24-month-old PDAPP mice (and $n = 19$ cells from their age-matched WT littermate controls) and $n = 26$ cells of 12-month-old rTg4510 mice (and $n = 26$ cells from their age-matched WT littermate controls).

To characterize the excitability of these cells, we focused on the properties of the first action potential (AP) elicited in response to a square depolarizing current pulse (300 pA, 500 ms; Figure 4.8A) and on the electrotonic properties of the plasma membrane measured upon the membrane potential exponential decay in response to a square hyperpolarizing current pulse (-100 pA, 500 ms; Figure 4.8B). To account for the biasing effect of cell-to-cell variability of the membrane potential over the excitability properties, all recordings were made from a starting membrane potential of $V_m = -80$ mV. This V_m value was obtained via the constant injection of a biasing current. To summarize the behavior of these voltage traces, we defined 9 features associated with the APs and 4 features associated with the membrane hyperpolarization.

The AP features are illustrated in Figures 4.9A-B and are defined as follows:

1. *AP threshold:* voltage at 10 percent of the AP max positive rate of rise (feature 6)
2. *AP peak:* maximum value of the voltage trace

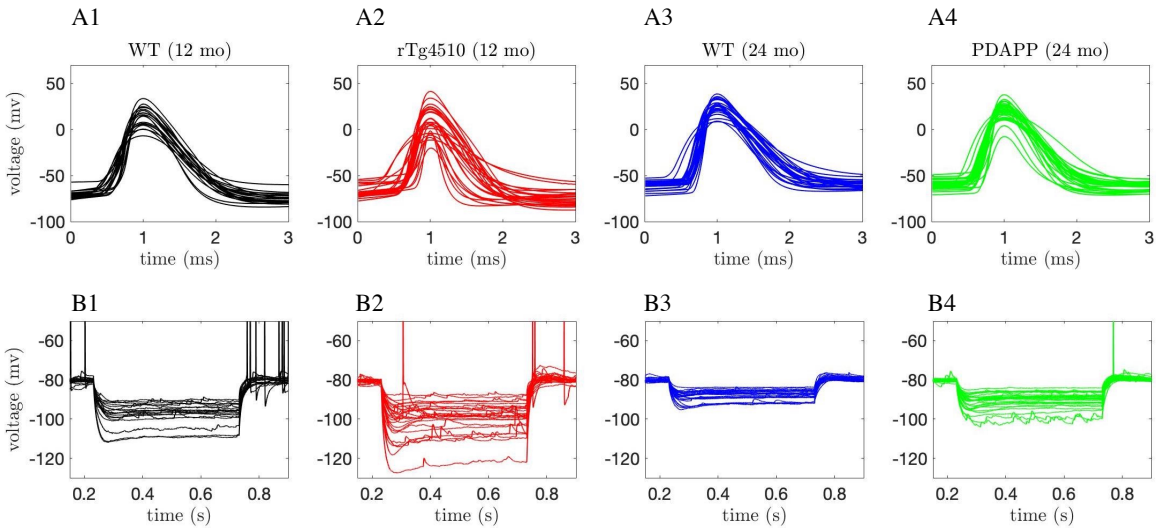

Figure 4.8 Experimental recordings. Waveforms of the first action potential in response to depolarizing current pulses (A1-A4) and voltage traces in response to hyperpolarizing current pulses (B1-B4) injected into CA1 pyramidal neurons for four different categories of mice: Wild-type (WT) 12-month-old mice (black traces, A1-B1), tau mutant (rTg4510) 12-month old mice (red traces, A2-B2), WT 24-month old mice (blue traces, A3-B3), and amyloid beta mutant (PDAPP) 24-month-old mice (green traces, A4-B4).

3. *AP trough:* minimum value of the voltage in the 2 ms time interval after the AP peak

4. *AP width:* duration of time that the voltage is above the AP voltage at max positive rate of rise (feature 7)

5. *AP min voltage before the pulse:* minimum voltage in the 1 ms interval before the AP peak

6. *AP max positive rate of rise:* maximum value of dV/dt in the 3 ms time interval around the AP peak (i.e., 1 ms before and 2 ms after the peak)

7. *AP voltage at max positive rate of rise:* voltage value at the AP max positive rate of rise

8. *AP max negative rate of rise:* minimum value of dV/dt in the 3 ms time interval around the AP peak (i.e., 1 ms before and 2 ms after the peak)

9. *AP voltage at max negative rate of rise:* voltage value at the AP max negative rate of rise.

The membrane hyperpolarization features are illustrated in Figure 4.9C and are defined as follows:

10. *HP A* - voltage at negative peak and baseline differences

11. *HP B* - voltage at exponential fit[1] and baseline differences

12. *HP C* - voltage at steady state and baseline differences

13. *HP D* - voltage at rebound and baseline differences.

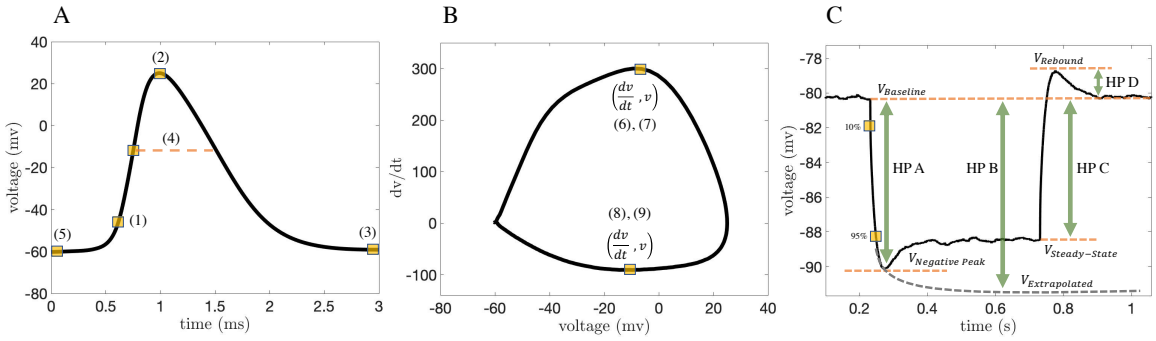

Figure 4.9 Schematic of feature extraction. (A-B) Action potential features: (1) AP threshold, (2) AP peak, (3) AP trough, (4) AP width, (5) AP min voltage before the pulse, (6) AP max positive rate of rise, (7) AP voltage at max positive rate of rise, (8) AP max negative rate of rise, (9) AP voltage at max negative rate of rise. (C) Hyperpolarization features: (10) HP A - voltage at negative peak and baseline differences, (11) HP B - voltage at exponential fit and baseline differences, (12) HP C - voltage at steady state and baseline differences and (13) HP D - voltage at rebound and baseline differences.

We note that these features were chosen to try to capture as much of the behavior of the voltage traces in as few features as possible. Increasing the dimensionality of the feature space can reduce the accuracy of cGAN training if the additional features are not sufficiently informative.

We then calculated these features for the voltage traces from PDAPP, rTg4510, and WT mice (see Figure 4.10 for the AP features, and Figure 4.11 for the hyperpolarization features). Despite the large amount of variability within each category, for some features clear differences are observed across categories. For example, AP peak appears to be reduced in PDAPP mice compared to their WT

[1]The exponential curve is fitted to the 10% into 95% of the falling phase of the voltage trace during the hyperpolarization pulse

controls (Figure 4.10 top middle panel) and AP width appears to be reduced in rTg4510 mice compared to their WT controls (Figure 4.10 middle left panel).

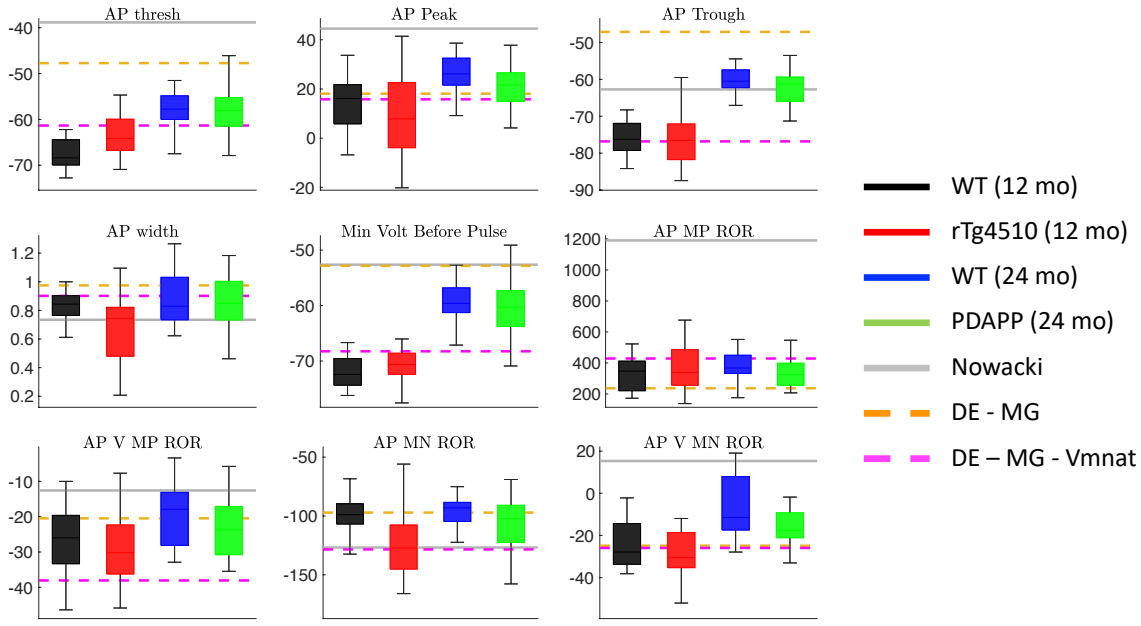

Figure 4.10 Action potential features in experimental data and initial models. Box and whisker plots of the action potential (AP) features extracted from the experimental data and the biophysical CA1 model with three different parameter sets: (1) the parameters in the original Nowacki et al. [46] paper (solid gray lines), (2) parameters obtained using differential evolution, optimizing only the maximal conductance parameters (DE-MG, dashed orange lines), and (3) parameters obtained using differential evolution, optimizing the maximal conductances and the half-activation voltage of the transient sodium current (DE-MG-Vmnat, dashed magenta lines).

4.3.3 Optimal Parameters - Differential Evolution

First, we simulated the pyramidal neuron model (see Section 2.4) with the parameter values provided in Nowacki et al. [46] (see Appendix Table A.1) and calculated feature values based on the model output (i.e., the simulated voltage trace). For certain features, the model's feature value is outside the range of the feature values observed in the experimental data (solid gray lines in Figures 4.10 and 4.11). For example, the AP threshold and AP peak in the model are significantly more depolarized than the

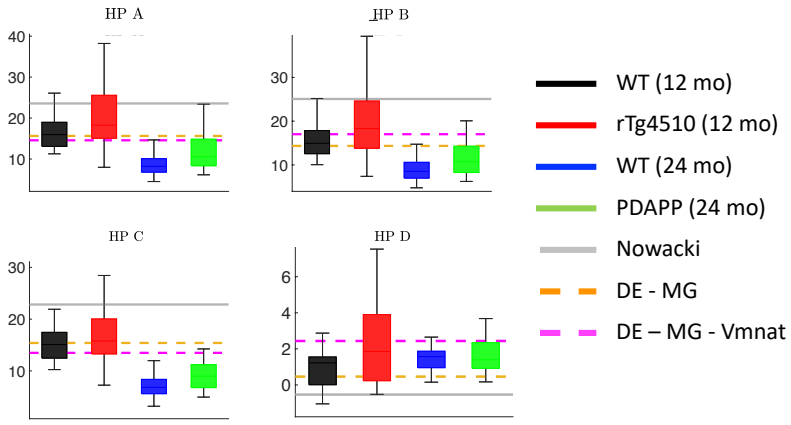

Figure 4.11 Membrane hyperpolarization features in experimental data and initial models. Box and whisker plots of the membrane hyperpolarization features extracted from the experimental data and the biophysical CA1 model with three different parameter sets: (1) the parameters in the original Nowacki et al. paper ([46], solid gray lines), (2) parameters obtained using differential evolution, optimizing only the maximal conductance parameters (DE-MG, dashed orange lines), and (3) parameters obtained using differential evolution, optimizing the maximal conductances and the half-activation voltage of the transient sodium current (DE-MG-Vmnat, dashed magenta lines).

AP thresholds and peaks seen in these CA1 neurons (Figure 4.10 top left and top middle panels).

Thus, we used stochastic global optimization to find model parameters that produce model output with feature values consistent with the experimentally observed feature values. Specifically, we used differential evolution (DE), a population-based search technique first introduced by Storn and Price [53, 62]. The objective function for the DE algorithm was to minimize the sum of squares error between the simulated voltage trace and the average voltage trace for each category (PDAPP, rTg4510, WT 12 and 24 month) across all four categories. More details on our implementation of the DE algorithm are provided in Section 3.1.

Initially, we chose to hold all the reversal potentials and gating variable parameters at their original Nowacki et al. values, so the only free parameters for DE to optimize were the 8 maximal conductances. The model with optimized

maximal conductances produced model output with feature values more consistent with the range of the feature values in the experimental data (dashed orange lines in Figures 4.10 and 4.11). However, this model's AP threshold was still significantly more depolarized than in the data (Figure 4.10 top left panel).

We used a variance-based sensitivity analysis (Sobol's method) to determine which model parameters affect the model's AP threshold, and found that the half-activation of the transient sodium current (V_{mNaT}) has the largest effect (see Section 3.2 for a description of our sensitivity analysis procedure). We then ran DE again, this time with V_{mNaT} as a free parameter in addition to the maximal conductances. The model with optimized V_{mNaT} produced model output with feature values within the range of the feature values in the experimental data, including the AP threshold (dashed magenta lines in Figures 4.10 and 4.11). Furthermore, the action potential and membrane hyperpolarization voltage traces produced by this model agree well with the experimental voltage traces themselves, as the model output appears to lie in the middle of the four voltage traces obtained by averaging the traces within each of the four categories (Figure 4.12).

We refer to the parameter set obtained through DE with V_{mNaT} and maximal conductances as free parameters as the "default" model parameters. We will use these parameters to set parameter bounds when creating the training dataset for cGAN.

4.3.4 cGAN Training on Biophysical Model and Validation on Synthetic Target Data

Recall that our main goal is to use cGAN to map voltage traces recorded from WT and Alzheimer's mutant mice to the parameter space of our CA1 model. To enable cGAN to learn the mapping from electrophysiological features to the CA1 model parameter space, we will create a training dataset consisting of features calculated from CA1 model simulations with randomly chosen parameter

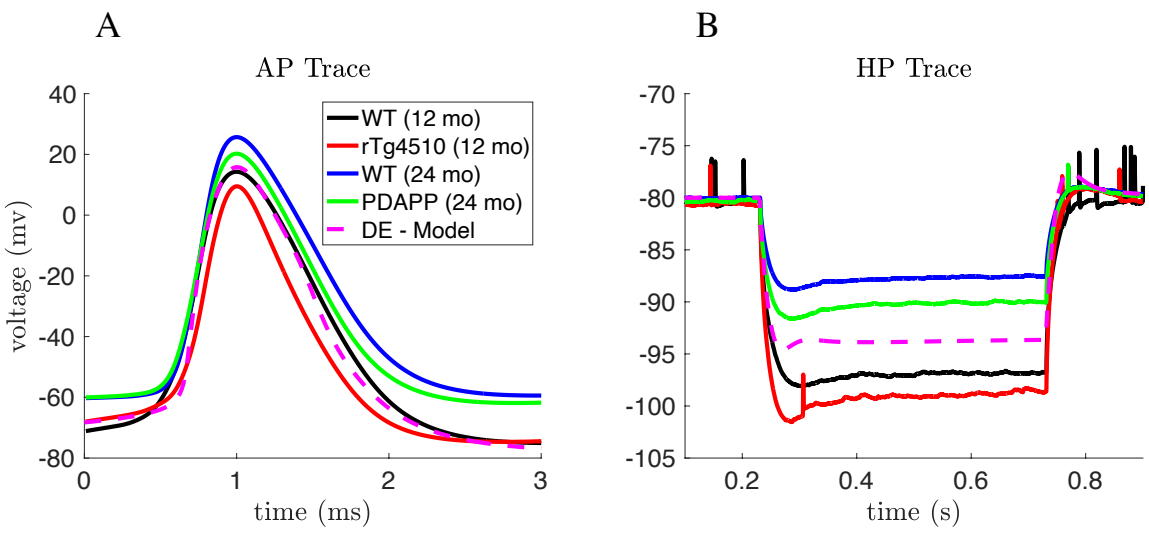

Figure 4.12 Average AP and hyperpolarization voltage traces from experimental data and optimized model. The DE-Model shown here is the same model that was labeled DE-MG-Vmnat in **Figures 4.11** and **4.10**, and was obtained by minimizing the least square error between the DE-Model output and the average AP and hyperpolarization traces across all four categories. (A) Mean of the AP waveforms in the experimental data for each category, and the AP waveform simulated using the optimized DE model (magenta). (B) Mean of the membrane hyperpolarization traces in the experimental data for each category, and the hyperpolarization trace simulated using the optimized DE model (magenta).

values. Since we are primarily interested in how the maximal conductances of the various ion channels present in CA1 pyramidal neurons are affected by aging and amyloidopathy/tauopathy, we will only vary the maximal conductance parameters in our training dataset and keep the gating variable kinetic parameters and the reversal potentials fixed at their default values. However, it may be that some of the maximal conductances do not have a large effect on the output features of interest. To explore this possibility, before creating a training dataset, we first conducted Sobol sensitivity analysis (see Section 3.2) to see how each of the 8 maximal conductance parameters affect the features. We found that 3 of these conductances, g_{NaP}, g_{CaT}, and g_L have a small effect on the features compared to the other 5 conductances (Figure 4.13). From a biological standpoint, these 3 conductances are unlikely to play a major role in determining the AP features for the following reasons: (1) persistent sodium current

(I_{NaP}) is likely to be much smaller than transient sodium current (I_{NaT}), (2) T-type calcium (I_{CaT}) is likely to be much slower than I_{NaT}, and (3) the leak current (I_L) primarily affects resting membrane potential rather than AP dynamics. Therefore, when we create the training dataset, we keep those 3 conductances fixed at their default values, and only vary 5 conductances: g_{NaT}, g_{CaH}, g_{KDR}, g_{KM}, and g_H.

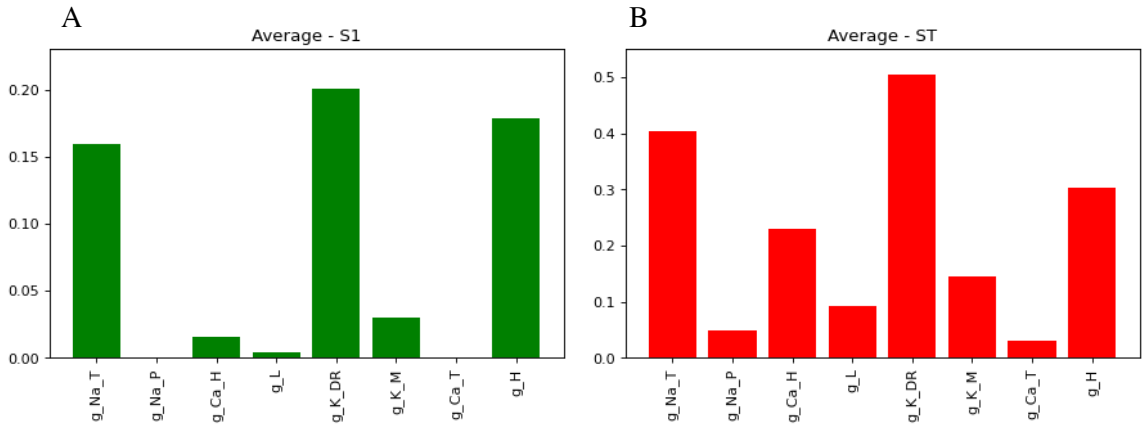

Figure 4.13 Dimensionality reduction using variance-based (Sobol) sensitivity analysis. The height of the bars represent how sensitive the AP and hyperpolarization features are to each parameter. (A) A1: average first-order index across feature space. (B) ST: average total-effect index across feature space.

For the training dataset, we performed 3 million simulations of the CA1 model with these 5 conductances drawn from uniform distributions with upper and lower bounds at ±100% of their default values. We then calculated the feature values for these simulations, and trained the cGAN with the parameters X conditioned on the features Y. Once the cGAN was trained, we passed the features for a subset of the training dataset (10,000 simulations) to the trained cGAN and asked it to produce samples (i.e., parameter sets) for those features. We then simulated the CA1 model with the parameter sets from the cGAN and calculated the features from these simulations. To compare the distributions of features from the training dataset and from the cGAN samples, we plotted Kernel Density Estimates (KDEs) for each feature and scatter plots for each pair of features (Figure 4.14A). These plots

show that the cGAN samples produced features that were very consistent with the features in the training data. We also plotted KDEs and pairwise contour plots for each parameter, which show that the distributions of parameters produced by the cGAN are similar to the parameter distributions in the training dataset (Fig. 4.14B). This visual conclusion was confirmed by calculating the Jensen Shannon Divergence. The JSD between the cGAN samples and training data on the joint distribution of parameters and features was approximately zero (1.11×10^{-14}).

4.3.5 Comparison with Markov Chain Monte Carlo Method on Synthetic Target Data

Although the results shown in Figure 4.14B are encouraging, it is important to test the performance of the cGAN on data that were not part of the training dataset. To create synthetic target data to use for validation, we constructed 100 random parameter sets with each parameter p drawn from a normal distribution with mean μ_p and standard deviation $\mu_p/8$, where μ_p is the default value of that parameter. If the randomly drawn value was negative or was larger than the upper bound for that parameter in the training set, then the value was set to zero or the upper bound, respectively.

We simulated these 100 parameter sets with the CA1 model, calculated the features, and passed the features to the trained cGAN to generate 100 cGAN parameter samples. We then simulated these cGAN parameter samples with the CA1 model, calculated the features, and compared the target and cGAN feature distributions. KDE plots for each feature, as well as 2D KDE plots for each pair of features, show that the cGAN feature distributions are similar to the target feature distributions (Figures 4.15 and 4.16A lower triangles). Furthermore, KDE plots for each parameter and each pair of parameters show that the cGAN parameter distributions are similar to the parameter distributions used to generate

the target data (Figure 4.16B lower triangle). To confirm these visual conclusions, we performed two-sample Kolmogorov-Smirnov (KS) tests to compare the cGAN and target distributions in feature and parameter space. In all cases but one (the voltage at the maximum positive rate of rise feature), the KS test failed to reject the null hypothesis (with a p-value threshold of 0.01) that these two sets of samples come from the same probability distribution, suggesting that the cGAN distributions are indeed similar to the target distributions.

We then performed the same procedure using a Markov chain Monte Carlo (MCMC) method instead of cGAN to infer parameters from the target data. The details of our MCMC implementation are provided in Section 3.3.1. We chose MCMC as the benchmark method to compare the performance of cGAN to because most state-of-the-art methods for solving stochastic inverse problems are based on MCMC [10, 34]. We passed the same 100 target data features to the MCMC algorithm as we did the cGAN, and then simulated the parameters produced by MCMC with the CA1 model. The feature distributions produced by the MCMC parameters, and the distributions of the MCMC parameters themselves, differ from their respective target distributions (Figures 4.15 and 4.16 upper triangles). Furthermore, we performed KS tests between the MCMC parameters and features and the target parameters and features, and in all cases the null hypothesis that these samples come from the same probability distribution was rejected, suggesting that the MCMC distributions are indeed different from the target distributions.

4.3.6 Parameter Inference Tests on Synthetic Target Data

To further test the ability of cGAN to accurately infer biophysical model parameters from feature data, we generated synthetic target datasets with a variety of underlying parameter structures. These structures were chosen to reflect the possible scenarios one may encounter when working with data from 2 different categories (e.g., data from WT versus mutant mice, or data from young versus old mice). For the CA1

model, we are interested in 5 parameters. Suppose that in the mutant mice, only 1 of these parameters (e.g., g_{NaT}) is altered compared to WT, and the other 4 parameters are not. To simulate this scenario, we construct two groups of target data. For each of the 4 parameters that are not hypothesized to be altered by the mutation (i.e., g_{CaH}, g_{KDR}, g_{KM}, and g_H), we draw 100 values from the same normal distribution $\mathcal{N}(\mu_p, (\mu_p/8)^2)$ for each group. For the parameter that is altered by the mutation (g_{NaT}), we draw 100 values from $\mathcal{N}(0.5\mu_p, (\mu_p/8)^2)$ for Group 1 and 100 values from $\mathcal{N}(1.5\mu_p, (\mu_p/8)^2)$ for Group 2. For each group, we then: (1) simulate these parameter sets using the CA1 model and calculate the features, (2) pass the features to the trained cGAN as target data to obtain cGAN samples (parameter sets), (3) simulate the cGAN parameter sets using the CA1 model and calculate the features, and (4) compare the cGAN feature and parameter distributions between Group 1 (G1) and Group 2 (G2) and to their respective targets. The G1 and G2 target distributions of some AP features are quite different from each other (Appendix Figure A.8 lower triangle), whereas the G1 and G2 membrane hyperpolarization feature target distributions are similar (Figure A.9 lower triangle), illustrating that the value of g_{NaT} affects some features more than others. Nonetheless, for all features the cGAN samples reproduce the target distributions well across both G1 and G2. Figure 4.17 (lower triangle) shows that the cGAN was able to accurately infer the distributions of all 5 parameters as well. Importantly, the cGAN-inferred distributions for g_{NaT} are distinct between Groups 1 and 2, whereas for the other 4 parameters the cGAN-inferred distributions are similar for G1 and G2.

We also used KS tests to assess the cGAN performance. First, we ran KS tests on the target data from G1 and G2. For the parameters, the null hypothesis that the G1 and G2 target samples come from the same distribution is rejected for g_{NaT}, but is not rejected for the other 4 parameters, as one would hope since this was the true structure used to create the target data. For the feature distributions, the KS tests

reject the null for 10 out of the 13 features. When we ran the KS tests on the cGAN G1 and G2 distributions, we get the exact same results for both the parameters and features as we did for the target data. This consistency in KS test results suggests that cGAN is able to correctly identify the structure of parameter variations between two groups of samples based on their features. Additionally, we ran KS tests to compare the cGAN distributions for G1 to the target data for G1, and the cGAN distributions for G2 to the target data for G2. For G1, all of the KS tests (for both parameters and features) failed to reject the null, again indicating the cGAN distributions are similar to the target distributions. For G2, all of the KS tests failed to reject the null except for one feature (the voltage at the maximum positive rate of rise). We then repeated this simulation and testing procedure 4 more times with the other possible choices for having a single parameter distinguish G1 and G2. For the G1 vs. G2 KS tests, the cGAN sample tests returned the same result as the target data tests 70 out of 72 times (Appendix Figure A.2 top panels). For the cGAN sample vs. target data KS tests, the null was rejected 0 out of 72 times for G1 (Figure A.2 bottom left panel) and 2 out of 72 times for G2 (Figure A.2 bottom right panel).

Next, we investigated scenarios with more than one parameter distinguishing G1 and G2. For example, we considered the case where g_{NaT} is not altered by the mutation, but the other 4 parameters all are altered by the mutation (i.e., $g_{NaT} \sim \mathcal{N}(\mu_p, (\mu_p/8)^2)$ in both G1 and G2, but g_{CaH}, g_{KDr}, g_{KM}, and g_H are all distributed $\mathcal{N}(0.5\mu_p, (\mu_p/8)^2)$ in G1 and $\mathcal{N}(1.5\mu_p, (\mu_p/8)^2)$ in G2). The KDE and scatter plots for the AP features (Figure A.8 upper triangle), membrane hyperpolarization features (Figure A.9 upper triangle), and parameters (Figure 4.17 upper triangle) indicate that the cGAN samples are consistent with the target data distributions. We also simulated the 4 other cases where each of the other 4 parameters was the only one not altered by the mutation. For the G1 vs. G2 KS tests, the cGAN sample tests returned the same result as the target data tests 89 out of 90 times (Figure A.5 top

panels upper portion). For the cGAN sample vs. target data KS tests, the null was rejected 0 out of 90 times for G1 and 3 out of 90 times for G2 (Figure A.5 top panels, bottom left and bottom right portions, respectively).

There are 35 other ways that exactly 4 out of the 5 parameters could be altered by the mutation. In Figure 4.17 (upper triangle), the other 4 parameters all had lower means in G1 than in G2. Instead, up to three of these parameters could have higher means in G1 than in G2 (if all 4 had higher means, it would be equivalent to the case we already simulated just with the G1 and G2 labels swapped). We simulated and tested these additional parameter structures. For the G1 vs. G2 KS tests, the cGAN sample tests returned the same result as the target data tests 624 out of 630 times (Figure A.5 bottom 7 panels upper portions). For the cGAN sample vs. target data KS tests, the null was rejected 15 out of 630 times for G1 and 38 out of 630 times for G2 (Figure A.5 bottom 7 panels lower portions).

So far, we have discussed scenarios where either exactly 1 parameter was affected by the mutation or exactly 4 parameters were affected by the mutation. Here, we consider the remaining scenarios of exactly 2, 3, or 5 parameters being affected. The number of possible cases for each scenario is given by

$$\binom{5}{k} \times 2^{k-1}, \quad k = 1, \cdots, 5 \tag{4.2}$$

where k is the number of parameters affected by the mutation. Thus, for $k = 2$, we have 20 different cases. For the G1 vs. G2 KS tests, the cGAN sample tests returned the same result as the target data tests 353 out of 360 times (Figure A.3 upper portions of panels). For the cGAN sample vs. target data KS tests, the null was rejected 3 out of 360 times for G1 and 15 out of 360 times for G2 (Figure A3 lower portions of panels). For $k = 3$, there are 40 different cases. For the G1 vs. G2 KS tests, the cGAN sample tests returned the same result as the target data tests 711 out of 720 times (Figure A.4 upper portions). For the cGAN sample vs. target data

KS tests, the null was rejected 9 out of 720 times for G1 and 38 out of 720 times for G2 (Figure A4 lower portions). Finally, for $k = 5$, we have 16 different cases. For the G1 vs. G2 KS tests, the cGAN sample tests returned the same result as the target data tests 288 out of 288 times (Figure A.6 upper portions). For the cGAN sample vs. target data KS tests, the null was rejected 9 out of 288 times for G1 and 13 out of 288 times for G2 (Figure A.6 lower portions). The results of the KS tests for all of the 5 choose k cases are summarized in Figure A.7.

In summary, these results on synthetic target data demonstrate that cGAN is capable of accurately identifying complex parameter variation structures from subtle differences in the features of CA1 model simulations. This gives us the confidence to apply the cGAN method to CA1 experimental data in Section 5.2.

4.4 Medium Spiny Neuron Network Model - Disease Effect

In this section we will show the results of using cGANs for parameter estimation on a mechanistic model of the MSN network (introduced in Section 2.5). Ponzi et al. [51] used their network model and the data from both wild-type (WT) and Hungtington's Disease (HD) mutant mice to generate insights into network pathologies associated with HD symptoms. In order to test the feasibility of using DeepHM for network models, we reanalyzed some of their data and applied the cGAN architecture. As the data provided in their paper is not well distributed between the lower and upper bounds of each parameter, we used Latin Hyper Cube sampling to help distribute the samples evenly in the parameter space. We made 170 new datasets by passing these parameters into the MSN-network mechanistic model. From the simulated spike trains for each neuron in the network (see Figure 4.18), we extracted four features for each of the samples: **mean interspike interval (ISI)**, **Local Coefficient of Variation (CV) of the ISIs**, **Skew of the ISIs**, and **Firing Rate**. The dataset comprising different parameter sets and their corresponding features formed

the synthetic dataset we used to train the cGAN. The trained cGAN then takes the 4D feature values as input and produces samples from parameter distributions that generate those feature values as output. We checked the trained network quality at each training epoch by probing the network with a feature set that was generated by simulating the MSN network model at a known point in parameter space, and measured the divergence between those target parameter values and the distribution of parameter values output by the cGAN. We found minimal divergence at training epoch 1000 (which was the last epoch in our training), and obtained parameter samples that were very close to the original parameter values (see Figure 4.19 and 4.20), despite a relatively coarse sampling of only 170 points in the original 2D parameter space.

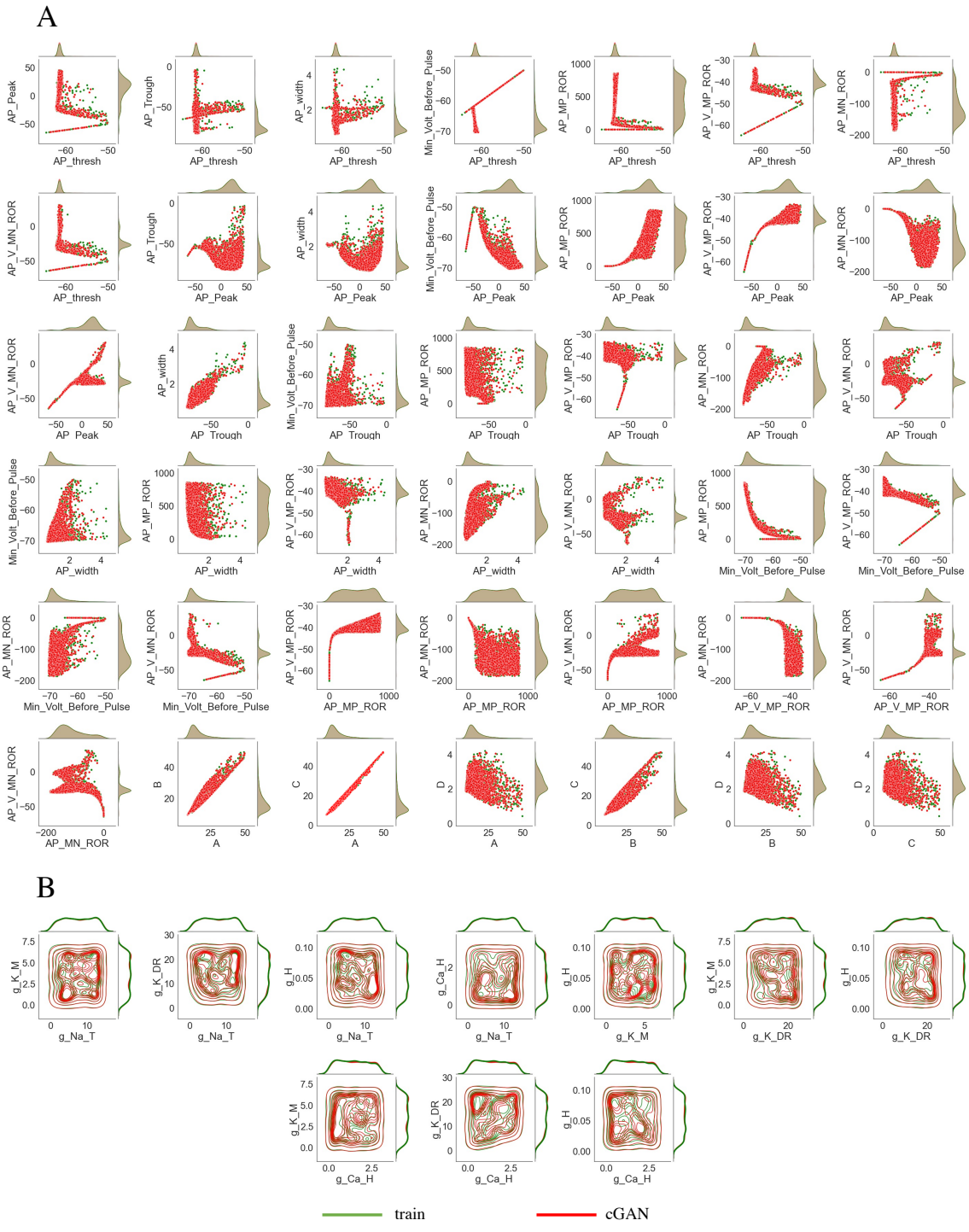

Figure 4.14 Comparison of cGAN samples and training dataset. (A) Feature space: scatterplots (center of panel) and KDE plots (top and right of each panel) with cGAN samples in red and the training dataset in green. (B) Parameter space: contour plots (center of panel) and KDE plots (top and right of each panel) with cGAN samples in red and the training dataset in green. In both (A) and (B), the KDE plots for the cGAN samples are nearly identical to the KDE plots for the training dataset (and have almost zero JSD measure).

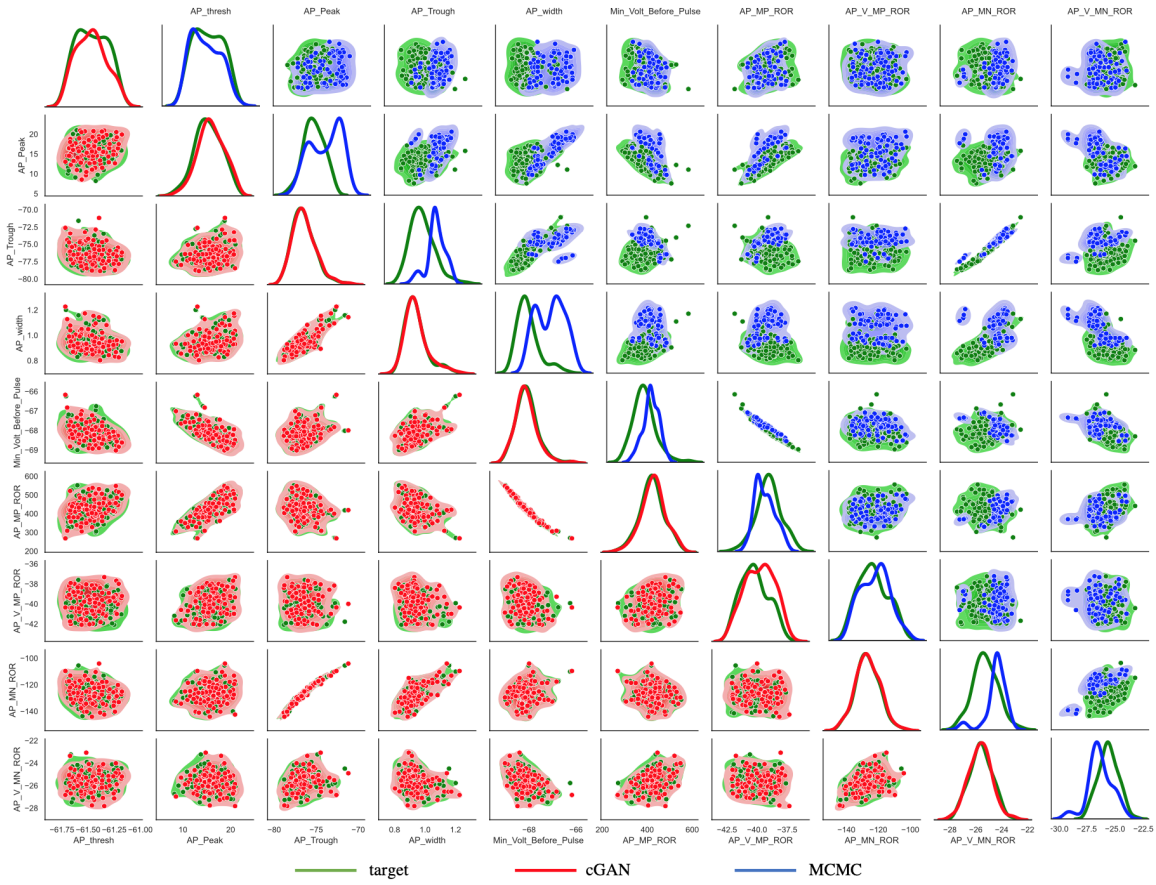

Figure 4.15 Performance of cGAN and MCMC on synthetic target data - AP features. *Lower main diagonal and lower triangle* - KDE and scatter plots of the cGAN samples (red) versus target (green). *Upper main diagonal and upper triangle* - KDE and scatter plots of the MCMC samples (blue) versus target (green).

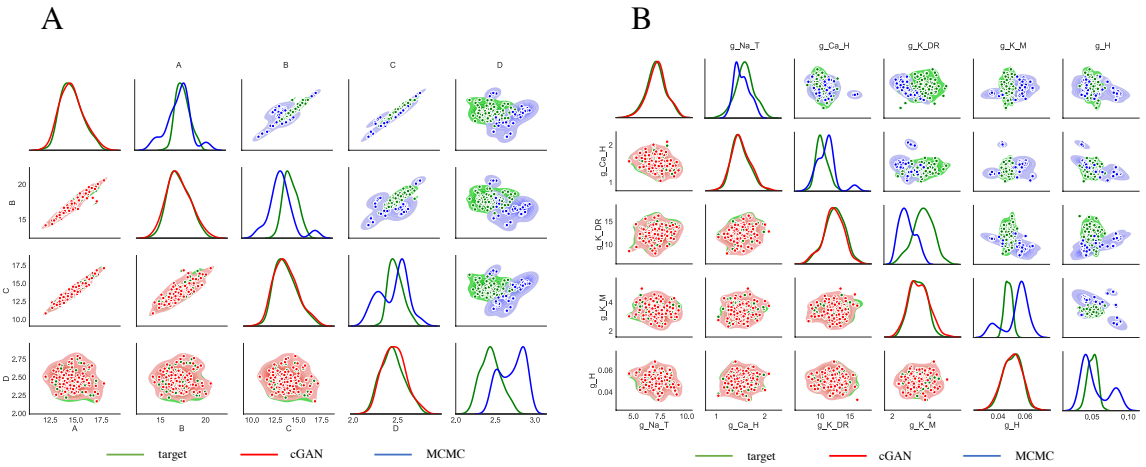

Figure 4.16 Performance of cGAN and MCMC on synthetic target data - HP features and parameter space. *Lower main diagonal and lower triangle* - KDE and scatter plots of the cGAN samples (red) versus target (green). *Upper main diagonal and upper triangle* - KDE and scatter plots of the MCMC samples (blue) versus target (green). (A) HP features. (B) Parameter space.

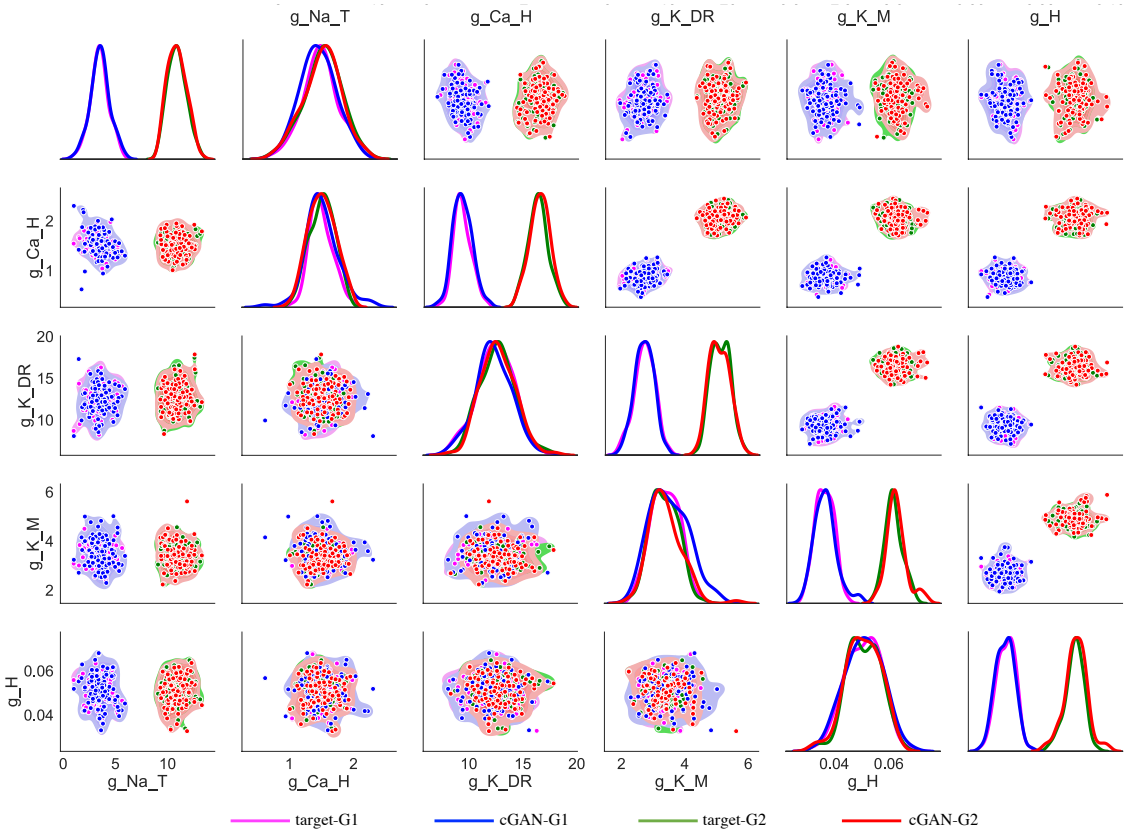

Figure 4.17 Performance of cGAN in parameter space on synthetic targets from two groups with distinct parameter structures. KDE plots (main diagonals) and scatter plots (lower and upper triangles) for Group 1 (G1) target data (magenta), Group 2 (G2) target data (green), cGAN samples for G1 (blue) and cGAN samples for G2 (red). *Lower main diagonal and lower triangle* - only one parameter (g_{NaT}) is distributed differently in the G1 target data than in the G2 target data, and the other four parameters have the same distribution in the G1 and G2 target data. *Upper main diagonal and upper triangle* - Four parameters (all parameters except g_{NaT}) are distributed differently in the G1 target data than in the G2 target data.

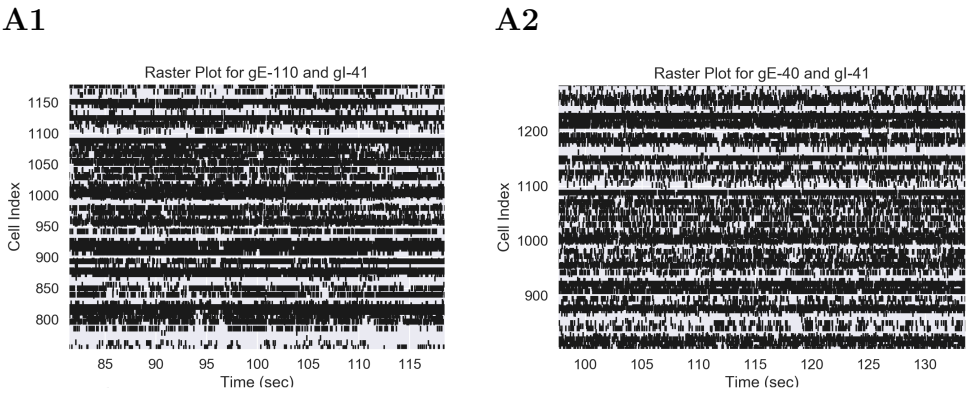

Figure 4.18 Spike raster plots for two different parameter sets of the MSN network model. The y-axis range is cell index (from 1 to 2500 as we have this number of cells in the MSN network), and the x-axis is time. Each row represents the spike train data for that specific cell, with black tick marks indicating the time of a spike.

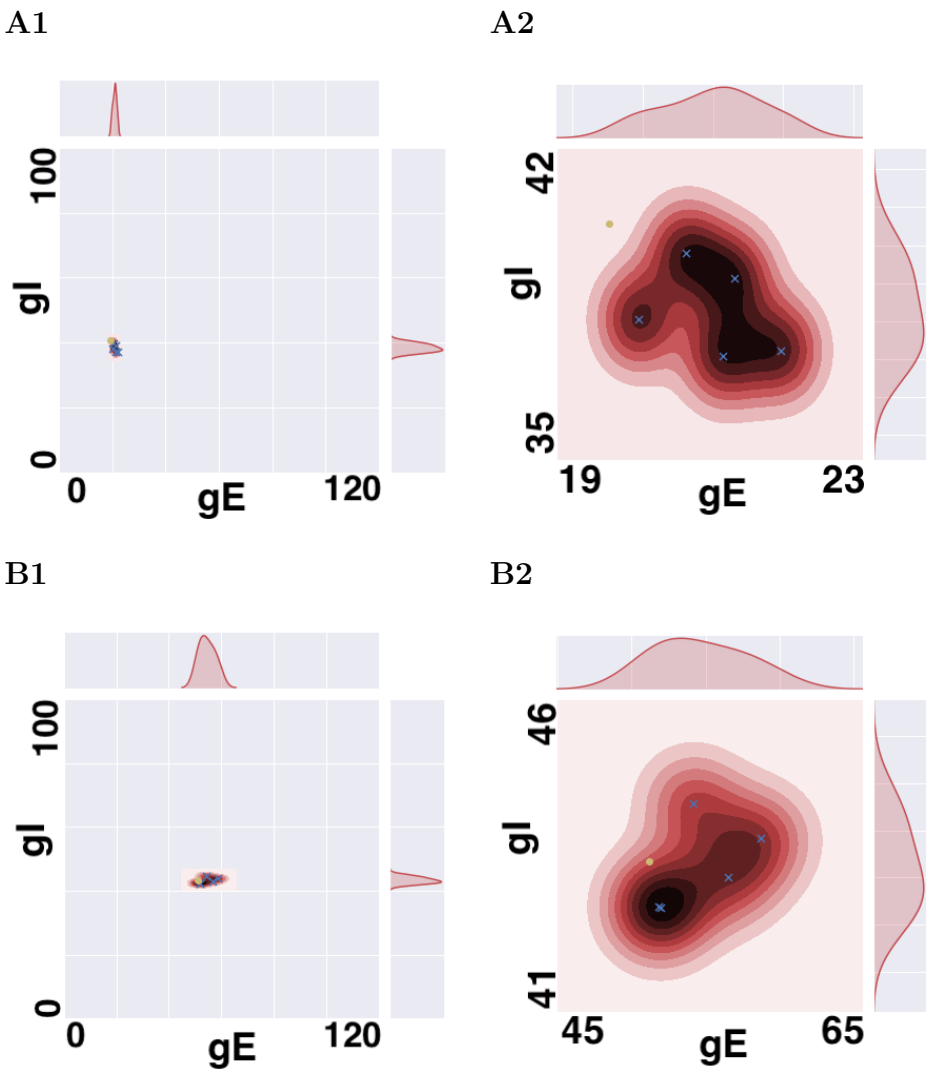

Figure 4.19 Panels A1 and B1 are plots of the joint parameter distribution for two different test datasets. The red area in each panel represents the sampling region coming from the final distribution of the generator based on their target vector (gE and gI parameters are known and are specified in each panel with a yellow circle). Panel A2 and B2 are magnifications of the left panels.

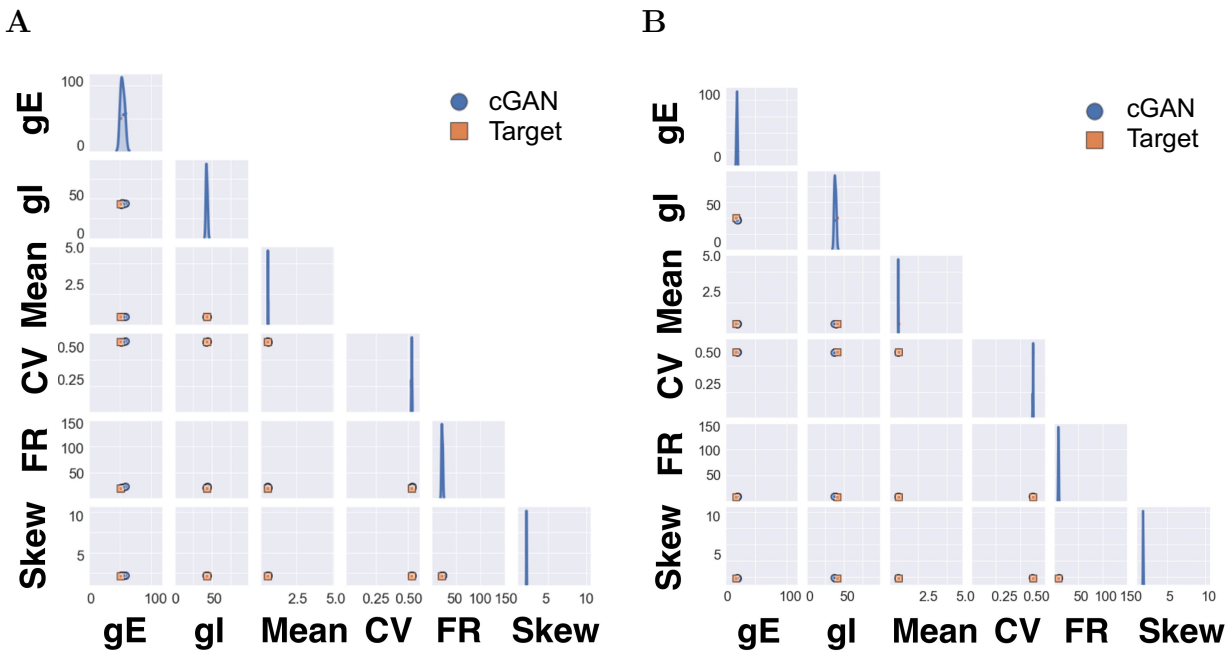

Figure 4.20 cGAN results for simulated MSN test data. KDE plots along the diagonal and scatter plots in the lower triangle. Sampled features in blue and target features in orange corresponding to the same panels in **Figure 4.19**. All samples created by the generator (i.e., blue points) are in a close neighborhood of the target features (i.e., orange points). This means the cGAN was able to map the target features correctly into the parameter space.

CHAPTER 5

PARAMETER INFERENCE ON EXPERIMENTAL TARGET DATA

The ability of the DeepHM framework to accurately identify complex parameter distributions from synthetic target data as demonstrated in Chapter 4 motivates us to apply the approach to experimental target data. In this chapter, we use cGANs to infer the distributions of ionic conductances underlying neuronal activity in three different biological contexts: (1) day/night variation of firing rate and RMP in SCN neurons, (2) altered excitability due to aging and mutations associated with Alzheimer's disease in CA1 pyramidal neurons, and (3) disrupted synaptic connections in the MSN network associated with Huntington's disease.

5.1 Suprachiasmatic Nucleus Neurons - Day/Night Variation

As described in Section 4.2, the SCN is a group of ∼20,000 neurons in the hypothalamus that serves as the central circadian pacemaker coordinating daily rhythms in physiology and behavior. In order to evaluate which ionic conductances are in charge of the variations responsible for the differences between day and night in the electrical activity of the Rhabdomys SCN, we will use the DeepHM methodology to map Rhabdomys SCN electrical activity data during the day and at night to the parameter space of our conductance-based SCN model (Section 2.3). These parameter distributions will enable us to infer which conductances are controlling the day/night variation.

Before analyzing the estimated parameter distributions generated by cGAN and their corresponding feature distributions, one may wonder how closely the voltage traces of the push forward of these estimated parameters into the SCN mechanistic model match their corresponding voltage traces obtained from real mice. Addressing this question is crucial because it reveals whether the number of defined features, and

the definition of the features themselves, were sufficient to capture the structure of the voltage trace. More specifically, if the feature distribution of the push forward parameters in the mechanistic model is very similar to the target distribution but there are some discrepancies in the voltage trace's shape, then more feature values must be defined in order to capture the shape of the target voltage traces as closely as possible. Figures 5.1A and 5.1B compare the voltage traces of the estimated cGAN samples pushed forward into the SCN model (on the left) versus the recorded data from the real mice (on the right). Four different examples of cells recorded during the night (panel A) and during the day (panel B) are provided.

The fact that the trained cGAN was able to converge into each cell's proper region in the parameter space indicates that the definition and the quantity of features were specified correctly based on these results. In Panel C, a swarm plot is used to display all of the extracted features (i.e., FR and RMP) from the real data as well as the cGAN samples that were pushed forward into the SCN model. The features of each cell were precisely captured by cGAN after converging into their proper region in the parameter space. In order to demonstrate the existence of nonlinearity and heterogeneity across various ionic conductances [45], we compared the parameter distribution predicted by cGAN to the feature distribution of real targets acquired from the experiment.

Figure 5.2A shows the KDE plots of the estimated cGAN parameter distribution for different target features (extracted from real mice, see Figure 5.2B). In other words, the trained cGAN created a population of samples back in the parameter space according to the given features all of which were extracted from the recorded voltage trace of the real mice. To verify the accuracy of the cGAN, the JSD measurement between the distribution of the features extracted from the real data and the distribution of the features from the push forward estimated samples into the SCN mechanistic model (see Figure 5.2C) was very close to zero. According

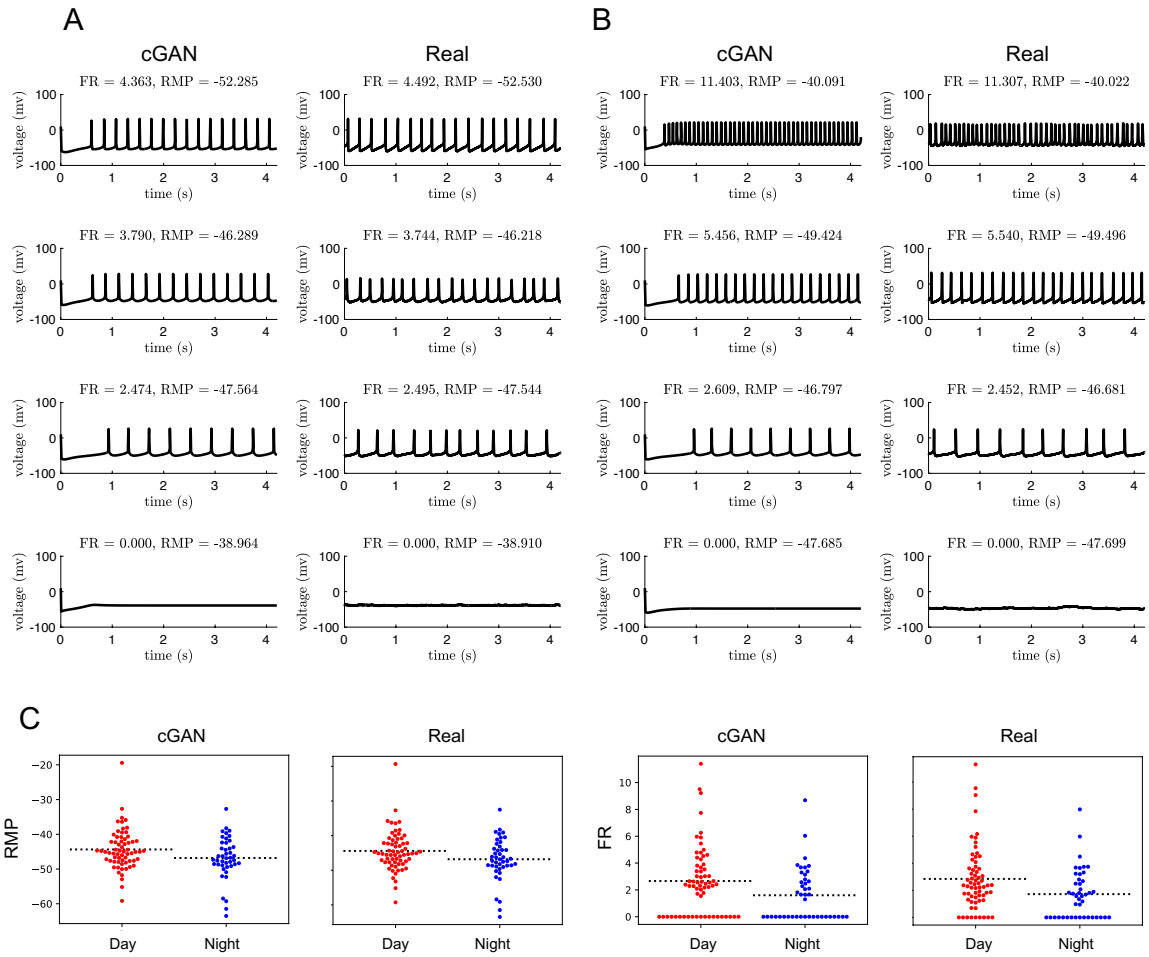

Figure 5.1 Day and night variation - voltage traces and swarm plots. (A) Voltage traces of the push forward cGAN estimated parameters into the SCN model (left panels) corresponding to the extracted features of the experimental targets (right panels) for real data recorded during the night. (B) Voltage traces of the push forward cGAN estimated parameters into the SCN model (left panels) corresponding to the extracted features of the experimental targets (right panels) for real data recorded during the day. (C) Swarm plots of extracted features of the push forward cGAN estimated parameters into the SCN model on the left in comparison with their corresponding target features on the right.

to Figure 5.2A, the most impactful parameters which play a role in capturing the differences between the day and night variation in the feature space are g_K and g_A. Although it is known that sodium leak currents play a role in regulating the day/night activity of SCN neurons [66] in mice, based on this figure it seems both $g_{L_{Na}}$ and g_{L_K} did not vary between day and night.

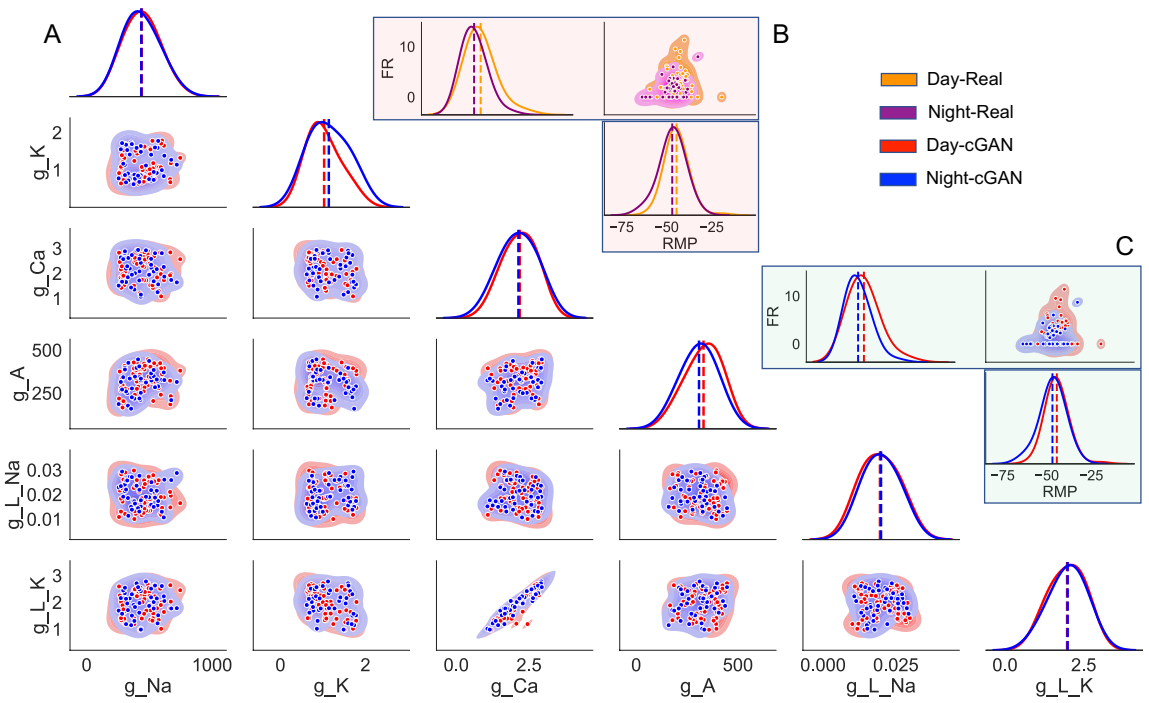

Figure 5.2 Day and night variation - distributions of the cGAN samples corresponding to the experimental targets in both parameter and feature space. (A) *Diagonal* - KDE plots of the parameter distributions, *off diagonal* - shaded contour plots with scatter plots on the top. (B) *Diagonal* - KDE plots of the feature distributions of the target features, *off diagonal* - shaded contour plots with scatter plots on the top. (C) *Diagonal* - KDE plots of the feature distributions of the push forward estimated parameters into the SCN model, *off diagonal* - shaded contour plots with scatter plots on the top.

To investigate the role of sodium leak currents in regulating the day/night variation in more detail, we split the target into two sub-targets; the first one represents all cells with zero firing rate (see Figure 5.3) as a feature and the other one is all cells with non-zero firing rate (see Figure 5.4). This approach will enable us to determine whether sodium leak played a role in either of the two sub-targets. Figure 5.3A – diagonal and lower triangle – represents the KDE and shaded contour plots with scatter plots on the top of cGAN samples for all zero firing rate target features. According to this figure, there is a higher impact of g_{Na}, $g_{L_{Na}}$ and g_{L_K} on day/night variation. As it is clear, these three parameters come with a higher mean for night in comparison with day which comes with the lower mean value.

More interestingly, the story is different when the target has non-zero firing rate. Figure 5.4A – diagonal and lower triangle – represents, g_{Ca}, g_A, $g_{L_{Na}}$ and g_{L_K} are the key parameters in this day/night variation where they come with a higher mean value for day in comparison with night which comes with the lower mean value.

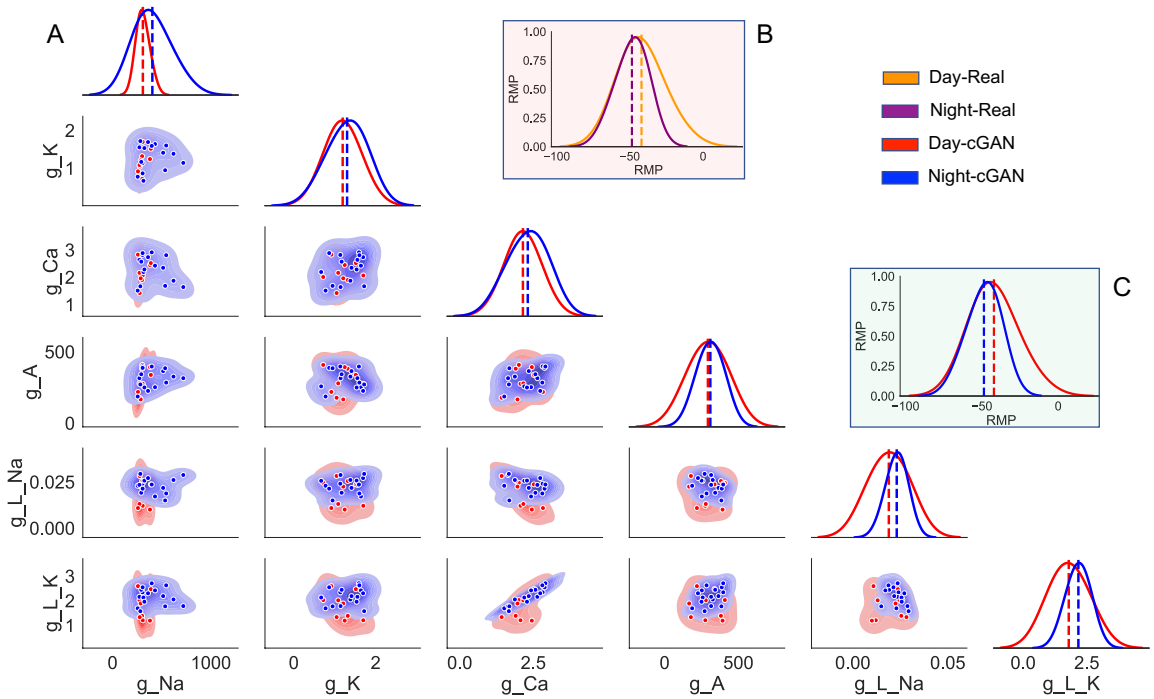

Figure 5.3 Day and night variation - distributions of the cGAN samples corresponding to the experimental targets with zero firing rate in both parameter and feature space. (A) *Diagonal* - KDE plots of the parameter distributions, *off diagonal* - shaded contour plots with scatter plots on the top. (B) *Diagonal* - KDE plots of the feature distributions of the target features, *off diagonal* - shaded contour plots with scatter plots on the top. (C) *Diagonal* - KDE plots of the feature distributions of the push forward estimated parameters into the SCN model, *off diagonal* - shaded contour plots with scatter plots on the top.

5.2 CA1 Pyramidal Neuron Model - Age and Disease Effect

5.2.1 Overview

In this part, we use DeepHM to estimate parameter distributions corresponding to the experimental data and infer which ion channels are altered in the Alzheimer's mouse models compared to their wildtype controls at 12 and 24 months. We

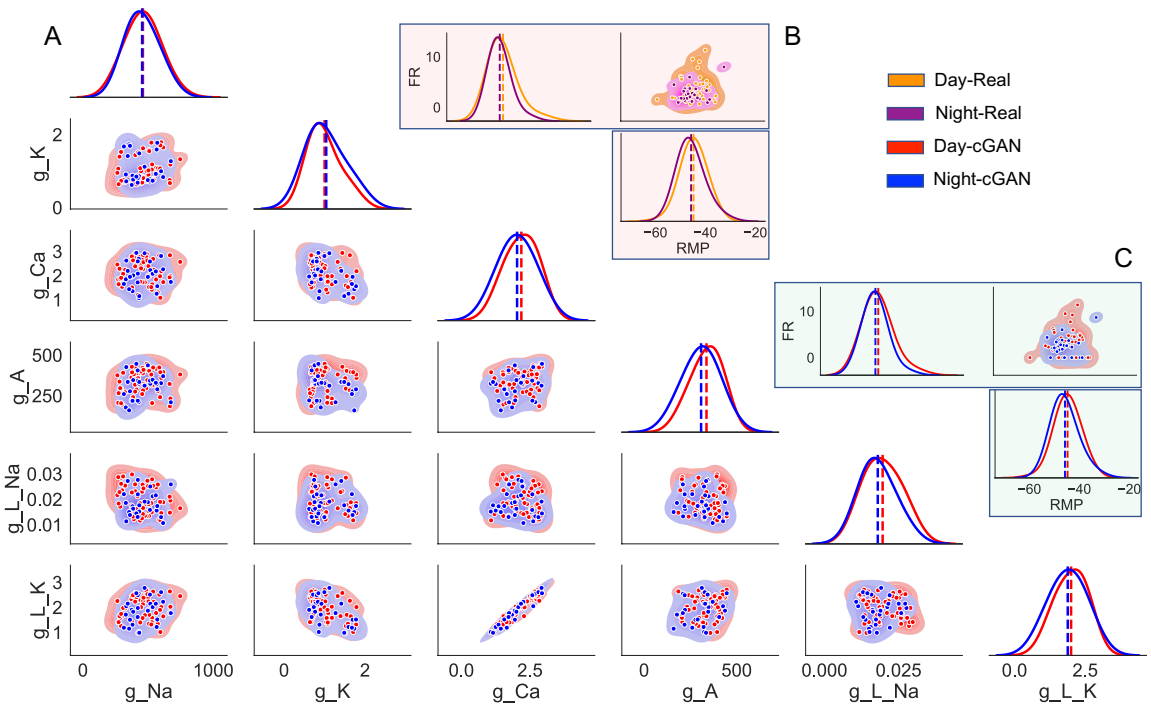

Figure 5.4 Day and night variation - distributions of the cGAN samples corresponding to the experimental targets with non-zero firing rate in both parameter and feature space. (A) *Diagonal* - KDE plots of the parameter distributions, *off diagonal* - shaded contour plots with scatter plots on the top. (B) *Diagonal* - KDE plots of the feature distributions of the target features, *off diagonal* - shaded contour plots with scatter plots on the top. (C) *Diagonal* - KDE plots of the feature distributions of the push forward estimated parameters into the SCN model, *off diagonal* - shaded contour plots with scatter plots on the top.

find that the conductances most disrupted by tauopathy, amyloidopathy, and aging are **delayed rectifier potassium**, **transient sodium**, and **hyperpolarization-activated potassium**, respectively.

5.2.2 Parameter Inference on Alzheimer's Mouse Model Data

Now that we have established cGAN as a tool for mapping observed traces to unobserved/unmeasured parameter values, we turn our attention back to experimental data (Figures 4.8, 4.10, 4.11) and seek to answer the following questions: Which maximal conductances are responsible for the differences observed in feature

space between (1) the wild type versus the mutant mice (i.e., the disease effect), and (2) the 12-month old mice versus the 24-month old mice (i.e., the age effect)?

To answer these questions, we will pass the features for each cell to the cGAN to obtain a population of models for each individual cell. For some cells, certain feature values fall outside the range of the values for that feature used in our training dataset. This can lead to inaccurate cGAN samples; thus, if a cell has a feature value outside that range we replaced that value with the median value of that feature across the training dataset.

We obtained 100 cGAN samples for each cell, and then pushed those parameters forward through the mechanistic model. Figure 5.5A shows that the mean AP and hyperpolarization traces produced by the cGAN samples agree well with the mean AP and hyperpolarization traces from the experimental recordings in each of the 4 categories. For example, we can see from the voltage traces that the AP peak feature exhibits the same trend in the cGAN samples and experimental data, with WT 24 month having the highest mean AP peak, followed by PDAPP, WT 12 month, and rTG4510, respectively. To give a sense of how the variability of the cGAN samples compares to the experimental recordings, Figure 5.5B shows the mean $\pm$ standard deviation of the AP and hyperpolarization traces. Overall, the amount of variability in the cGAN samples appears comparable to the amount of variability in the experimental data across the four categories, with the exception of the hyperpolarization traces for WT 24 month where there is less variability in the cGAN samples than in the data. Furthermore, box-and-whisker plots for the output of the cGAN samples in feature space also agree well with the feature distributions in the experimental data (compare Figure A.10 to Figure 4.10, and Figure A.11 to Figure 4.11).

Seeing that the cGAN samples produce appropriate behavior in feature space, we move on to assessing the distributions of the cGAN samples in parameter space

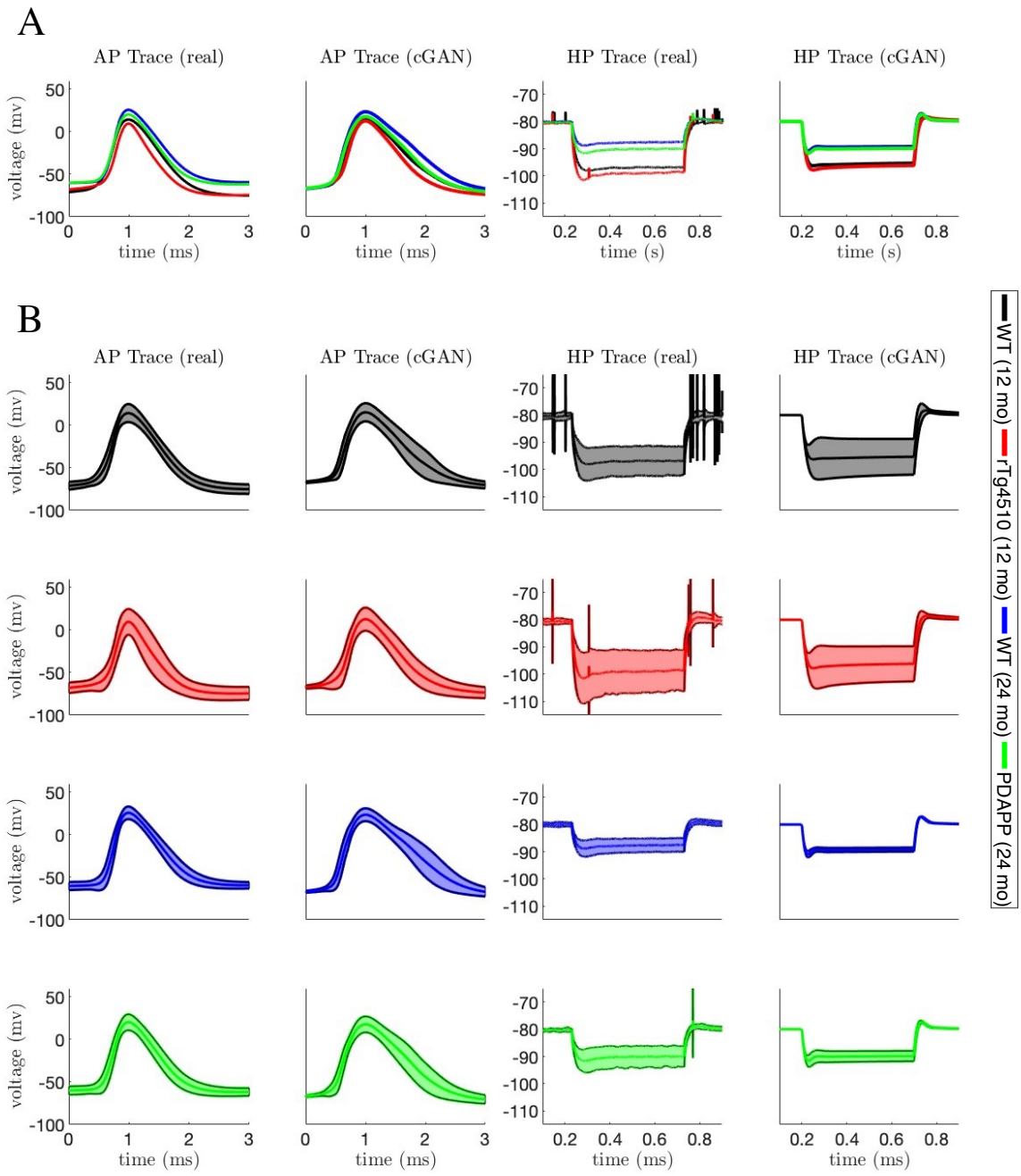

Figure 5.5 *AP and membrane hyperpolarization traces from cGAN samples with experimental targets.* (A) Mean AP and membrane hyperpolarization traces from each experimental data category (1st and 3rd panels) and mean AP and membrane hyperpolarization traces from simulated the mechanistic model with 100 cGAN parameter samples for each cell in each category (2nd and 4th panels). (B) Same as A1, but shading shows the mean ± standard deviation for each category.

in order to answer the questions posed at the beginning of this section. First, we compare the cGAN samples for rTg4510 and their age-matched controls (WT 12 month). Based on KDE plots for each parameter (lower main diagonal of Figure 5.6), we see that for 3 of the parameters (g_{NaT}, g_{CaH}, g_H), the WT 12 month and rTg4510 distributions are centered around the same values. However, the distribution for g_{KDR} is shifted to the right in rTg4510 relative to WT 12 month, whereas the distribution of g_{KM} is shifted to the left in rTg4510 relative to WT 12 month. The KDE plots comparing PDAPP and their age-matched controls (WT 24 month) show a similar trend, with the g_{KDR} and g_{KM} distributions shifted to the right and left, respectively, for PDAPP relative to WT (upper main diagonal of Figure 5.6). For PDAPP, the distribution of g_{NaT} is also shifted to the left relative to WT. From these observations, we hypothesize that for the mouse model of tauopathy, it is the conductances g_{KDR} and g_{KM} that are responsible for the altered excitability properties. For the mouse model of amyloidopathy, we hypothesize that these conductances plus g_{NaT} play a role in the altered excitability.

Having considered the disease effect, we now move on to assessing the age effect. First, we compare the cGAN samples for WT 12 month and WT 24 month. Based on KDE plots for each parameter (lower main diagonal of Figure 5.7), we see the most striking differences for g_H, with the distribution for the older mice shifted to the right relative to the distribution for the younger mice.

The parameter g_{KM} also shows a rightward shift in the older mice. On the other hand, the distribution for g_{KDR} is shifted to the left in the older mice. The 12 to 24 month WT comparison is the most appropriate one for assessing an age effect, since the only difference between these two groups of mice is their age. However, for the sake of completeness we also compared the 12 month mutant (rTg4510) to the 24 month mutant (PDAPP). Remarkably, the 3 shifts in the parameter distributions that we observed with age in the WT mice were all preserved in the mutant mice, despite

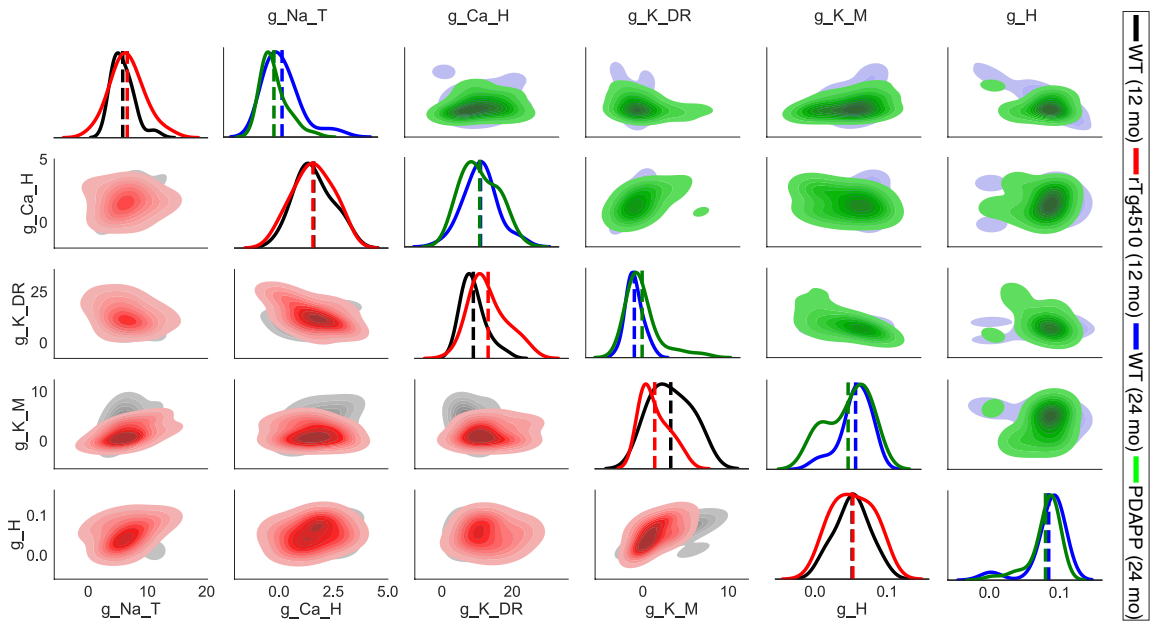

Figure 5.6 Disease effect - parameter distributions from cGAN samples with experimental targets. *Lower main diagonal and lower triangle* - KDE and shaded contour plots of cGAN samples for 12-month-old WT (black) and 12-month-old tau mutant (rTg4510, red) mice. *Upper main diagonal and upper triangle* - KDE and shaded contour plots of cGAN samples for 24-month-old (blue) and 24-month-old amyloid beta mutant (PDAPP, green) mice.

the fact that the 12 and 24 month mutants have different mutations (tauopathy and amyloidopathy, respectively). Specifically, the g_H and g_{KM} distributions are shifted to the right in the older mutants relative to the younger mutants, while g_{KDR} is shifted to the left in the older mutants (upper main diagonal of Figure 5.7). These results lead us to hypothesize that these three conductances underlie the changes in excitability properties observed with aging.

5.3 Medium Spiny Neuron Network Model - Disease Effect

In Section 4.4, we covered the performance of cGAN on a couple of examples based on synthetic target simulated by our mechanistic model. In other words, both training and test features came from model simulations. In these cases the cGAN performed with high accuracy even though samples were distributed coarsely in the parameter

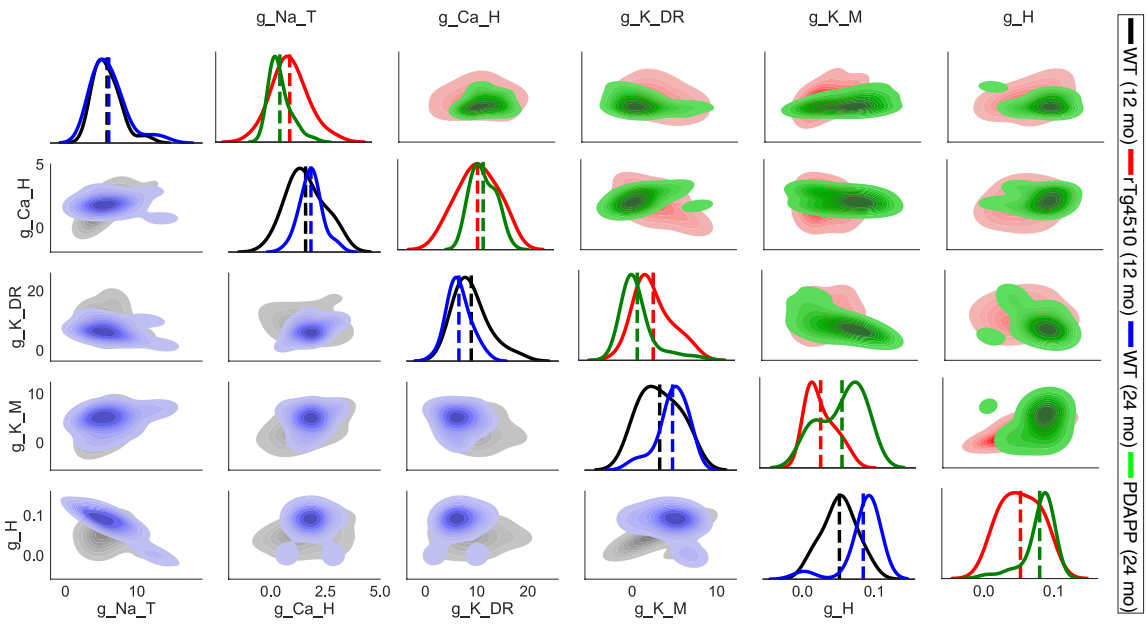

Figure 5.7 Age effect - parameter distributions from cGAN samples with experimental targets. *Lower main diagonal and lower triangle* - KDE and shaded contour plots of cGAN samples for 12-month-old (black) and 24-month-old (blue) WT mice. *Upper main diagonal and upper triangle* - KDE and shaded contour plots of cGAN samples for 12-month-old (rTg4510, red) and 24-month-old (PDAPP, green) mutant mice.

space. At this point, an important question is how this performance might change when using experimental (rather than simulated) data to obtain the test features. Thus, we next extracted features from time series data recorded from mice. Figure 5.8 shows the results of cGAN parameter inference for real mouse data. In panel A, the accuracy of cGAN samples are high, which means cGAN was able to map the target features correctly into parameter space. However, in panel B, cGAN failed to map the target features into the parameter space.

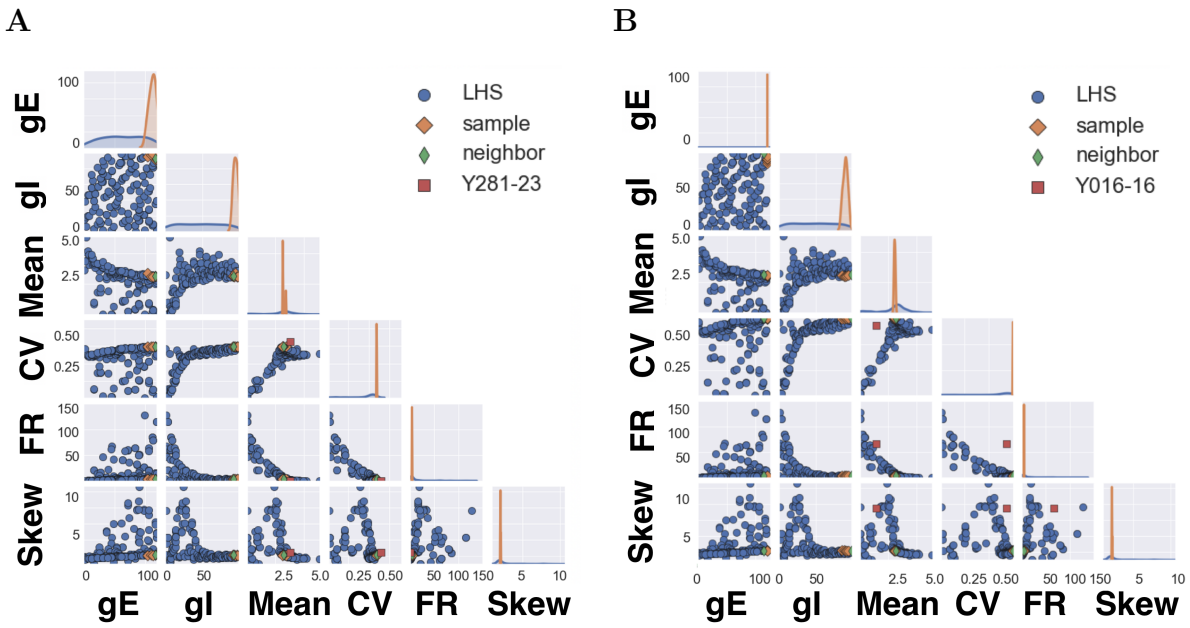

Figure 5.8 cGAN results for experimental MSN test data (i.e., Y281-23 and Y016-16). KDE plots on diagonal and scatter plots in lower triangle. (A) High accuracy in cGAN parameter inference model - Latin Hypercube Sampled (LHS) training dataset covers the range of target in the feature space - cGAN samples (orange diamonds) are in a close neighborhood of the target features (red squares). (B) Low accuracy in cGAN parameter inference model - Latin Hypercube Sampled (LHS) training dataset do not cover the range of target in the feature space - target feature is away from the region of features in the training dataset. In both panels green diamonds represents the closest features among the training dataset to the target features.

The main reason for the poor performance of the cGAN shown in panel B is the lack of information about that area of feature space in the simulated dataset that the cGAN was trained on. In general, if the target features that cGAN wants to map in parameter space have been seen before in our synthetic dataset, then the accuracy of cGAN will be high. On the other hand, poor performance of cGAN is inevitable if the target features are outside of the region of features seen in our training dataset. One easy way to solve this issue is by increasing the number of samples in our training dataset, but this is difficult to do for models that are very time consuming to simulate. Indeed, the MSN network model considered here is very expensive to simulate with each taking more than an hour to complete. Therefore, we need a strategy to find the

best region of parameter space to sample, thereby reducing the number of simulations needed to get as close as possible to the region of target features.

CHAPTER 6

CONCLUSION AND OUTLOOK

6.1 Overview

In this dissertation, we looked at different biophysical models at both single-cell and network levels and tried to address certain unanswered questions regarding the biophysical mechanisms with a novel parameter inference technique. The conclusions and main results and conclusions discussed throughout the dissertation are summarized in this chapter. We then end by listing some potential follow-up studies that apply the findings from this work to additional interesting challenges.

6.2 Summary of Results

The last decade has seen a rise in the application of population-based modeling in the neuronal and cardiac electrophysiology domains [33, 38, 44, 50, 54, 58, 60]. The development of methods for selecting and producing virtual patient populations that accurately reflect the statistics of clinical populations has also received a lot of attention in fields such as quantitative systems pharmacology [2, 11, 18, 30, 55].

In this dissertation, we have introduced and employed a deep hybrid modeling (DeepHM) framework [48, 49] featuring conditional generative adversarial networks (cGANs) that can be categorized as a population of models technique. More specifically, we considered a stochastic inverse problem (SIP), where the experimental data comes from multiple individuals across a population. Our method utilized recent advances in deep learning to generate model parameter sets that produced a population of deterministic mechanistic models with outputs that are consistent with the experimental population data. A distinct but related problem is simulation-based inference (SBI), where experimental data are acquired from a single individual. In SBI, stochastic mechanistic model is used to infer the set of model parameters most

likely to have generated the data distribution observed from the individual. While deep learning methods such as neural density estimation with normalizing flows have been used in SBI problems [20, 37], to our knowledge our work is among the first to apply a deep learning approach to an SIP.

We compared the performance of cGAN and a standard Bayesian inference Markov chain Monte Carlo (MCMC) method [52] on a parameter inference task with synthetic target data where the ground truth was known. The cGAN outperformed MCMC on this task, and showed it is capable of producing a population of models that captures the type of variability that is often present in biological data.

The cGAN was able to accurately detect a variety of complex differences in the distribution of parameters across different synthetic target data such as type-I and type-II excitability targets in the Morris-Lecar model (Section 4.1), day/night variation in a modified Hodgkin-Huxley model (Section 4.2), and across two groups of synthetic target data in a CA1 pyramidal neuron model (Section 4.3). Therefore, based on this success with synthetic target data, we employed the cGAN approach to infer the parameter distributions in experimental target data across two groups of SCN recordings (one group recorded during the day, and the other during the night) and four groups of CA1 pyramidal neurons (WT and mutant mice at two different ages). From these distributions, we drew conclusions about the biophysical mechanisms (i.e., the ionic conductances) underlying the differences in the observed excitability properties of day versus night, WT versus mutant, and younger versus older mice. Overall, our results illustrate the value of mapping experimental data back to the parameter space of a mechanistic model.

6.3 Future Work

In future research, the predictions we made about the role of certain conductances in either day/night variation in the SCN or in Alzheimer's disease and aging phenotypes

in CA1 can be tested experimentally. In addition, we can expand our analysis of the SCN to the network level. In the SCN, the main neurotransmitter coupling neurons to each other is GABA. Although GABA is an excitatory neurotransmitter early in development, throughout most areas of the adult brain it acts as an inhibitory neurotransmitter. The SCN is an exception to this rule, as GABA exhibits both excitatory and inhibitory actions in the adult SCN. The polarity of GABA synapses depends on the concentration of chloride in the post-synaptic cell. There is data suggesting that the expression of chloride transporters in SCN neurons is under circadian control. Thus, a major open question in the circadian field is whether or not GABA polarity changes across the day/night cycle, contributing to day/night rhythms in overall SCN activity. To test this hypothesis, one can conduct simulations of an SCN network model and then use DeepHM to map the features back to parameter space. The mechanistic model parameters of most interest to us are the excitatory and inhibitory synaptic conductances, g_{syn-E} and g_{syn-I}. Thus, this problem has a very similar setup to the MSN network explored in Section 5.3. The SCN network simulations will take a long time to complete; in fact, we predict that we won't be able to supply the cGAN with enough training data to function properly. In order to deal with this issue, one can develop a new GAN approach where it trains iteratively to identify the most optimal regions of parameter space and minimize the number of simulations it needs to perform.

Finally, since DeepHM can produce populations of cell models that accurately reflect the heterogeneous responses of real cells, this framework could prove useful in virtual drug design applications. This future direction is inspired by recent work where a population of models approach was used to identify a set of ion channel drug targets that optimally convert Huntington's disease cellular phenotypes to healthy phenotypes simultaneously across multiple measures [1].

APPENDIX

SUPPLEMENTARY INFORMATION

A.1 Supplementary Results

A.1.1 Synthetic Target Methodology

As we are dealing with five parameters in our mechanistic model, we design different target data scenarios with five choose k parameters distinguishing two different groups of target data.

5 choose 0: In this scenario, since $k = 0$, we only have one target group. All five parameters are drawn from a normal distribution with a mean μ and standard deviation $\mu/8$, where μ is the value of that parameter in the optimized DE model parameter set (DE-MG-Vmnat). There is only one case to consider in this scenario.

5 choose 1: In this scenario, since $k = 1$, we choose one of the five parameters to draw from a different distribution for the Group 1 (G1) and Group 2 (G2) target datasets. For that parameter, we draw the G1 samples from a normal distribution with a mean of 0.5μ and the G2 samples from a normal distribution with a mean of 1.5μ. For both groups the normal distribution has a standard deviation of $\mu/8$, where again μ is the value of that parameter in the optimized DE model. There are five different cases in this scenario, since there are five parameters that can be chosen to distinguish G1 and G2.

5 choose k: Here we consider the scenarios $k \in \{2, 3, 4, 5\}$. The number of cases for each value of k can be computed from Equation (A.1). The 2^{k-1} term reflects the number of different ways k parameters can be different between G1 and G2. For example, suppose $k = 3$, and the 3 parameters chosen to be distributed differently between G1 and G2 are *g-Na-T*, *g-Ca-H*, and *g-K-DR*. There are 2^2 different possibilities: (1) all 3 parameters are low in G1 and high in G2 – denoted

L-H, L-H, L-H; (2) *g-Na-T* is high in G1 and *g-Ca-H*, *g-K-DR* are low in G1 – denoted H-L, L-H, L-H; (3) *g-Na-T* low in G1, *g-Ca-H* high in G1, and *g-K-DR* low in G1 – denoted L-H, H-L, L-H; and (4) *g-Na-T*, *g-Ca-H* low in G1 and *g-K-DR* high in G1 – denoted L-H, L-H, H-L. We do not have to simulate possibilities such as H-L, H-L, H-L or L-H, H-L, H-L, since they are equivalent to possibilities (1) and (2) listed above, respectively, just with the labels swapped for G1 and G2 swapped.

$$\binom{n}{k} \times 2^{k-1}, \quad k = 1, \cdots, 5 \tag{A.1}$$

A.1.2 Kolmogorov Smirnov Tests

We performed Kolmogorov Smirnov tests (KS-tests) to compare the cGAN samples to the ground truth target samples for all the different 5 choose k synthetic target scenarios. The null hypothesis for these tests is that the two samples are from the same probability distribution. The plots in Figures A.1-A.6 represent the results of these tests, where black indicates the null hypothesis is rejected (p-value ≤ 0.01) and peach indicates the null hypothesis is not rejected (p-value > 0.01).

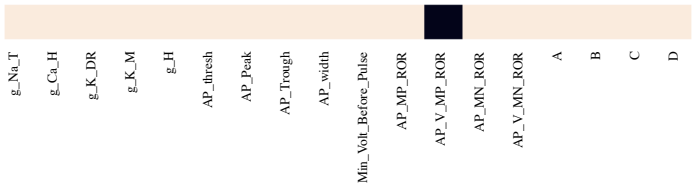

Figure A.1 5 choose 0 - KS tests for cGAN samples versus target data samples. Peach color indicates failure to reject the null hypothesis that the cGAN and synthetic target data samples are from the same distribution.

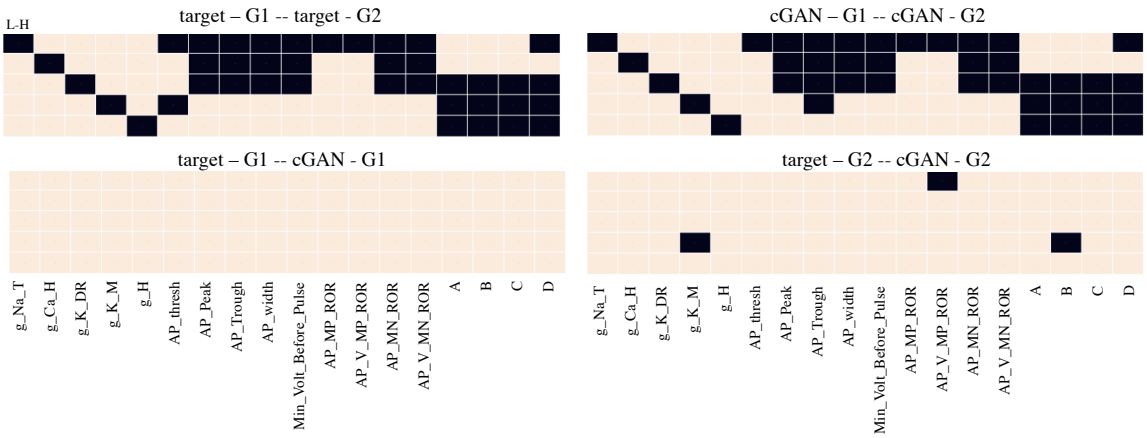

Figure A.2 5 choose 1 - KS tests. *Top left* - KS tests on two groups of targets (i.e., G1 (G2) with low - L (high - H) values of the parameter). *Top right* - KS test on cGAN samples corresponding to the two groups of target data (i.e. cGAN-G1 (cGAN-G2) with low - L (high - H) values of the parameter). *Bottom left* - KS test of cGAN-G1 versus target-G1. *Bottom right* - KS test of cGAN-G2 versus target-G2.

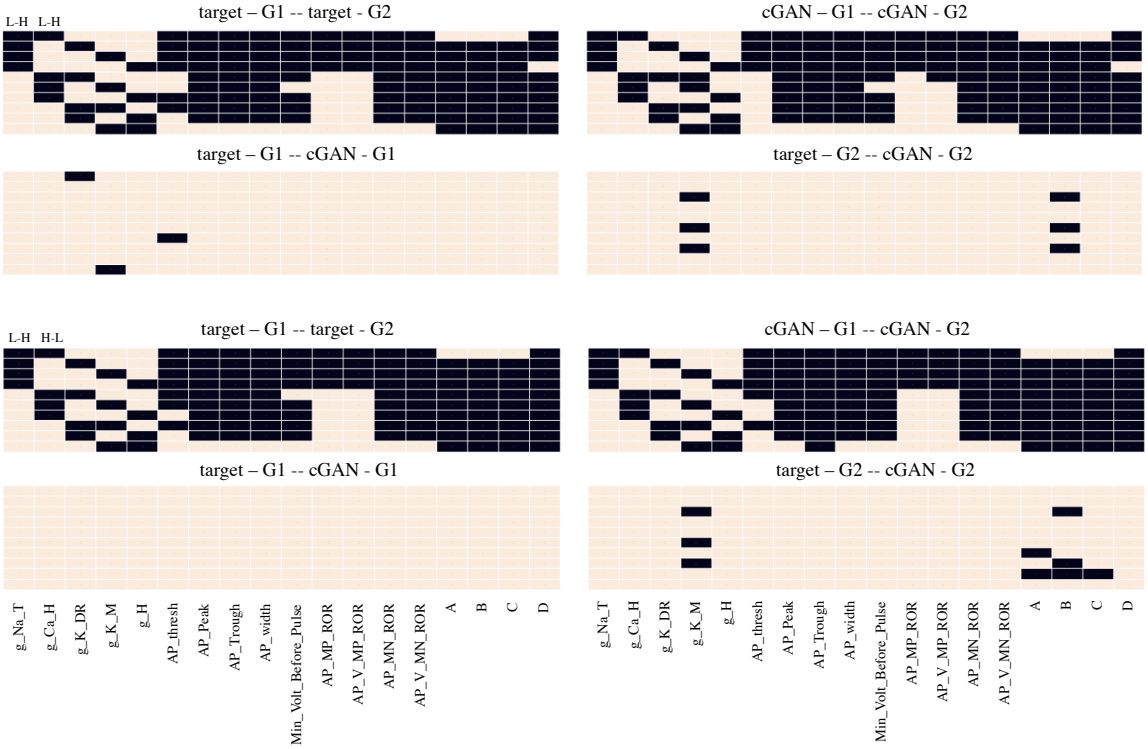

Figure A.3 5 choose 2 - KS tests. Panels are arranged in a similar fashion as Figure A.2. Top: L-H, L-H. Bottom: L-H, H-L.

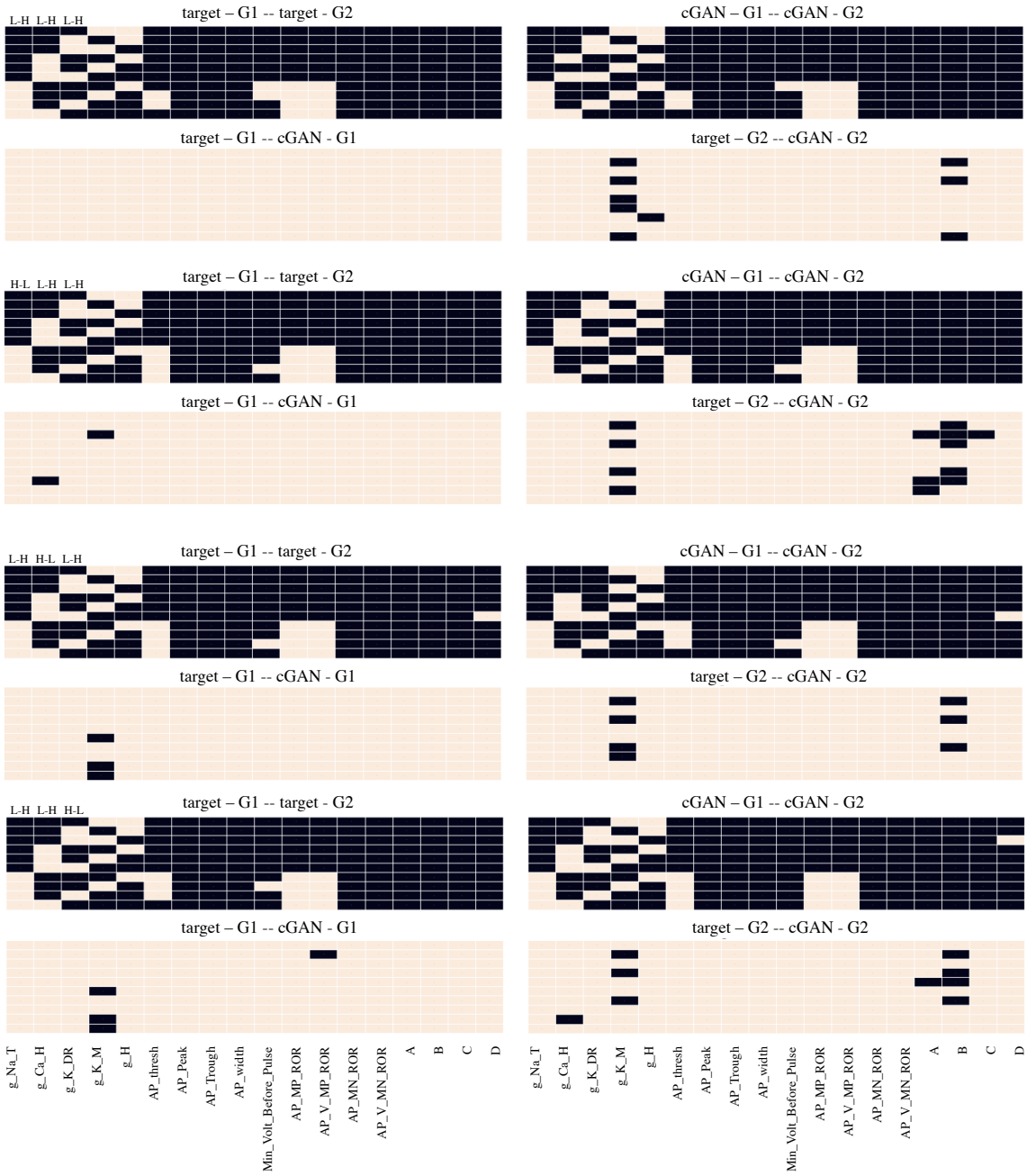

Figure A.4 5 choose 3 - KS tests. Panels are arranged in a similar fashion as Figure A.2. From top to bottom: *(1) L-H L-H L-H*, *(2) H-L L-H L-H*, *(3) L-H H-L L-H*, *(4) L-H L-H H-L*.

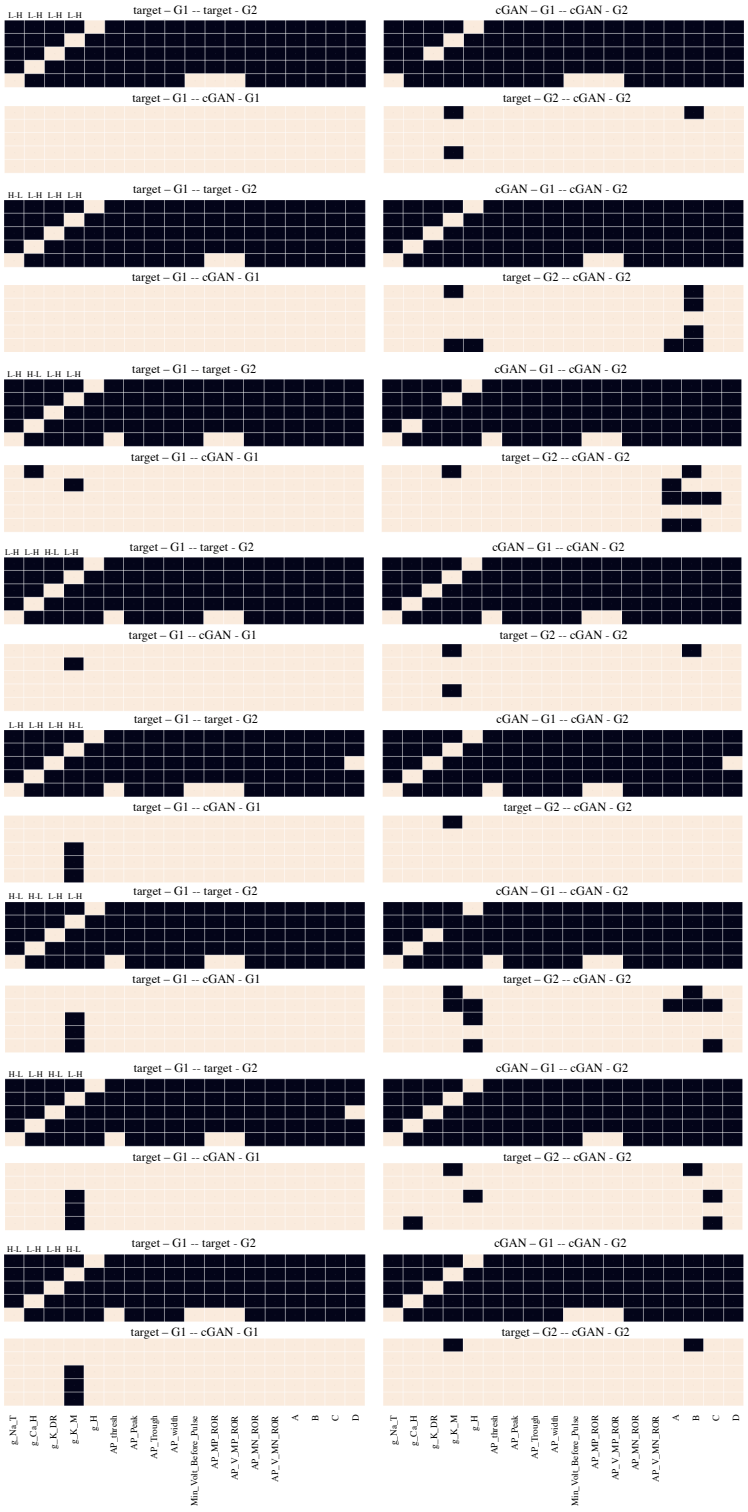

Figure A.5 5 choose 4 - KS tests. Panels are arranged in a similar fashion as Figure A.2. From top to bottom: *(1) L-H L-H L-H L-H*, *(2) H-L L-H L-H L-H*, *(3) L-H H-L L-H L-H*, *(4) L-H L-H H-L L-H*, *(5) L-H L-H L-H H-L*, *(6) H-L H-L L-H L-H*, *(7) H-L L-H H-L L-H*, *(8) H-L L-H L-H H-L*.

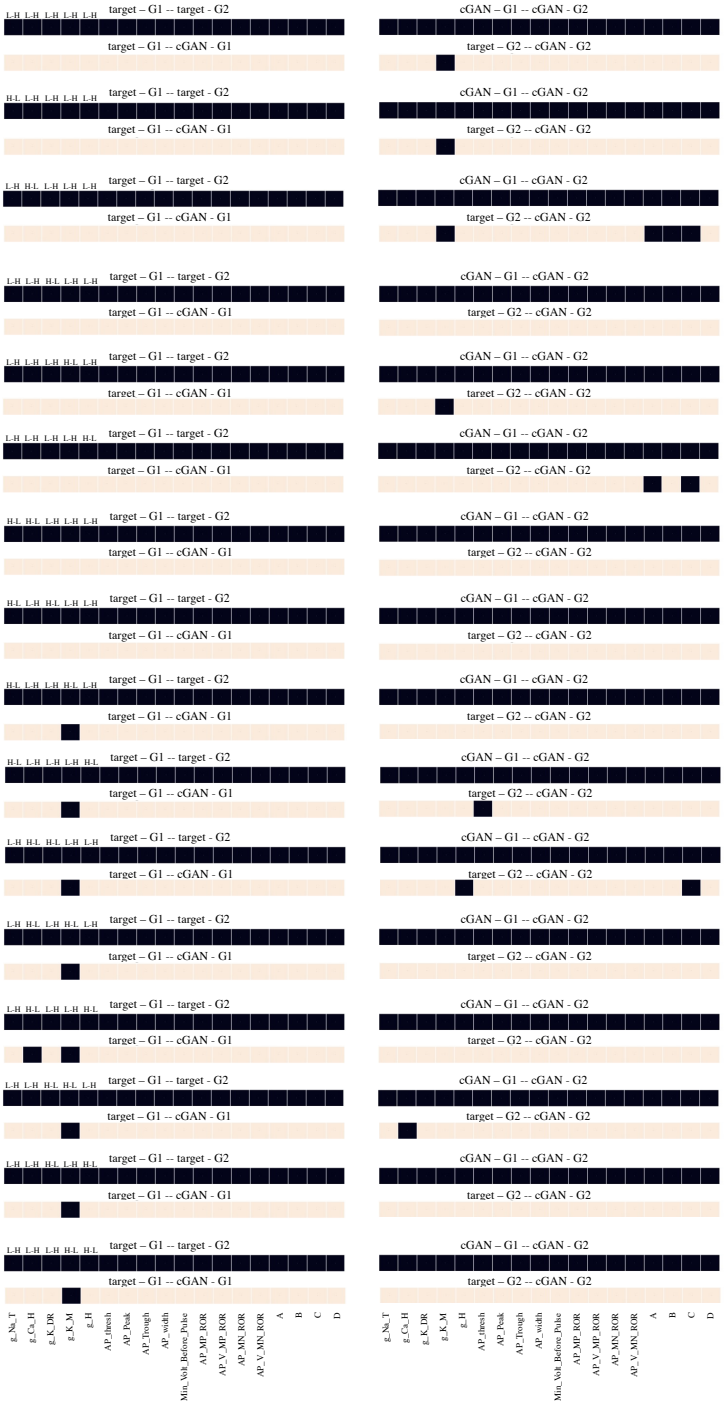

Figure A.6 5 choose 5 - KS tests. - Panels are arranged in a similar fashion as Figure A.2. From top to bottom: *(1) L-H L-H L-H L-H L-H*, *(2) H-L L-H L-H L-H L-H*, *(3) L-H H-L L-H L-H L-H*, *(4) L-H L-H H-L L-H L-H*, *(5) L-H L-H L-H H-L L-H*, *(6) L-H L-H L-H H-L L-H H-L*, *(7) H-L H-L L-H L-H L-H*, *(8) H-L L-H H-L L-H L-H*, *(9) H-L L-H L-H H-L L-H*, *(10) H-L L-H L-H L-H H-L*, *(11) L-H H-L H-L L-H L-H*, *(12) L-H H-L L-H H-L L-H*, *(13) L-H H-L L-H L-H H-L*, *(14) L-H L-H H-L H-L L-H*, *(15) L-H L-H H-L L-H H-L*, *(16) L-H L-H L-H H-L H-L*.

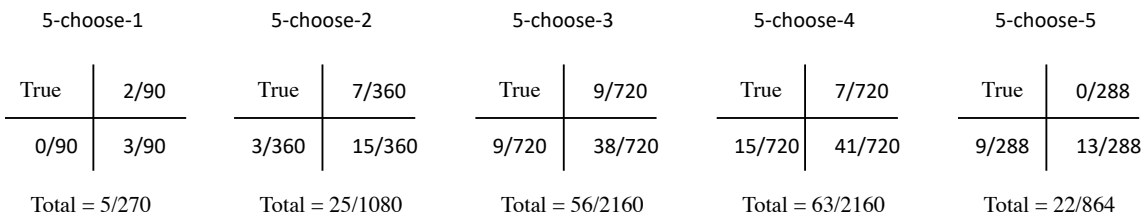

Figure A.7 Summary of KS test results for all 5 choose k synthetic target data cases. The demoninators are the total number of tests, and the numerators are the number of those tests for which the null hypothesis was rejected. Top left quadrants: These KS tests are taken to be the ground truth as they compared target G1 versus target G2. Top right quadrants: These KS tests compared cGAN-G1 versus cGAN-G2. Bottom left quadrants: These KS tests compared target-G1 versus cGAN-G1. Bottom left quadrants: These KS tests compared target-G2 versus cGAN-G2.

A.1.3 Output of cGAN Samples in Feature Space for Synthetic Target Data

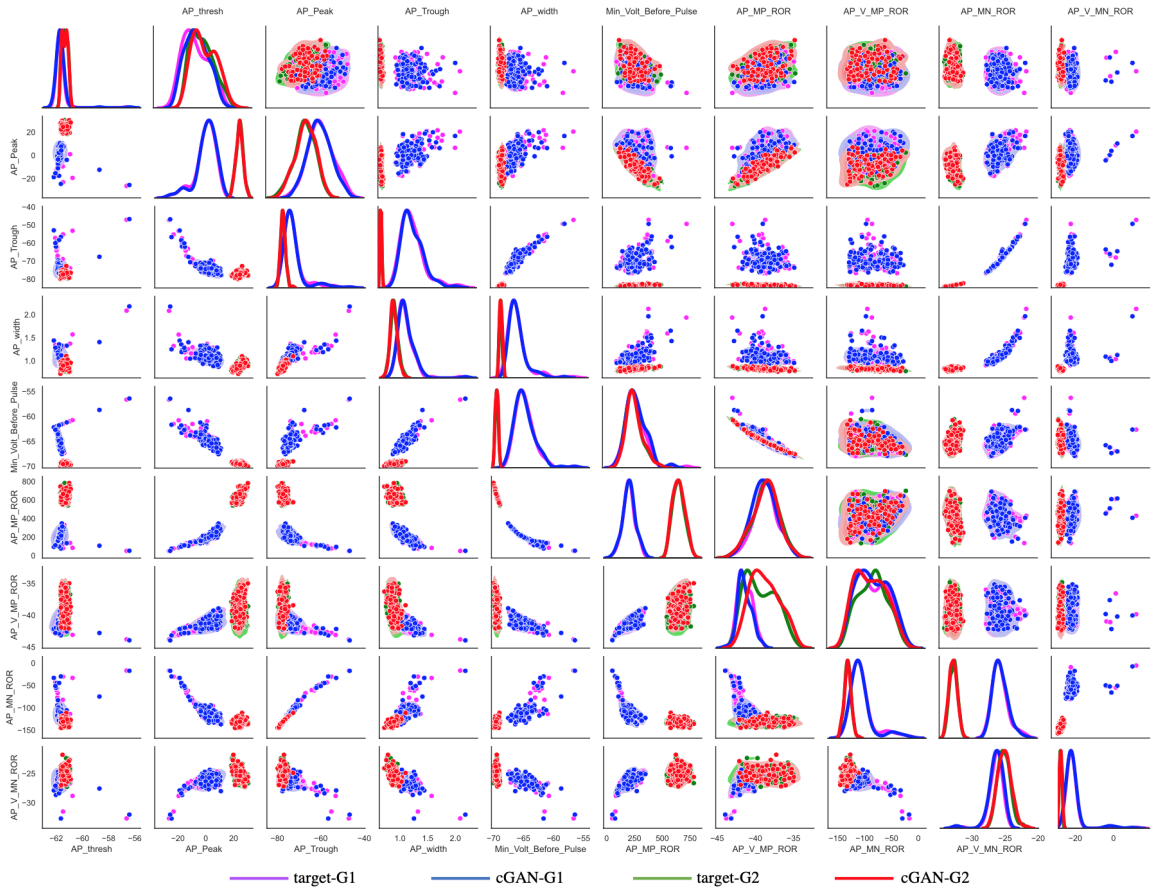

Figure A.8 Performance of cGAN on synthetic targets from two groups with distinct parameter structures - AP features. KDE plots (main diagonals) and scatter plots (lower and upper triangles) for Group 1 (G1) target data (magenta), Group 2 (G2) target data (green), cGAN samples for G1 (blue) and cGAN samples for G2 (red). *Lower main diagonal and lower triangle* - only one parameter (g_{NaT}) is distributed differently in the G1 target data than in the G2 target data, and the other four parameters have the same distribution in the G1 and G2 target data. We refer to this scenario as "5 choose 1" in Appendix A.1.1. *Upper main diagonal and upper triangle* - Four parameters (all parameters except g_{NaT}) are distributed differently in the G1 target data than in the G2 target data. We refer to this scenario as "5 choose 4" in Appendix A.1.1.

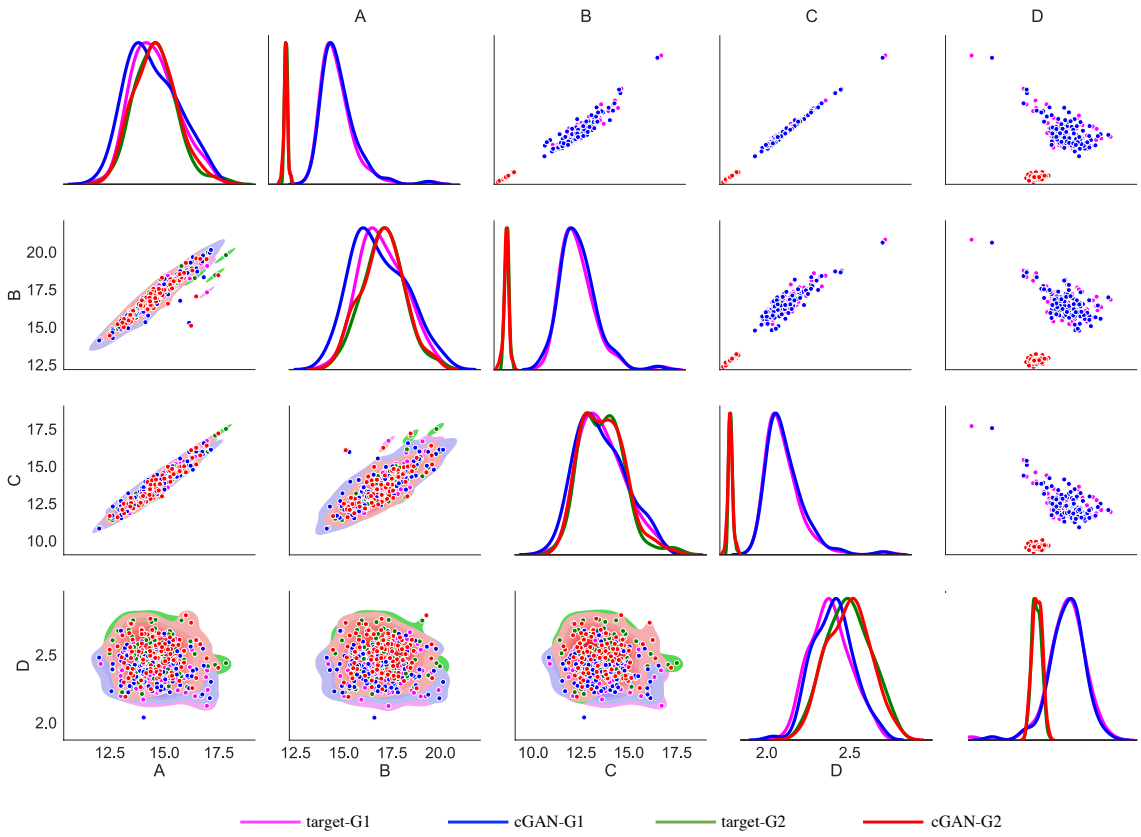

Figure A.9 Performance of cGAN on synthetic targets from two groups with distinct parameter structures - HP features. KDE plots (main diagonals) and scatter plots (lower and upper triangles) for Group 1 (G1) target data (magenta), Group 2 (G2) target data (green), cGAN samples for G1 (blue) and cGAN samples for G2 (red). *Lower main diagonal and lower triangle* - only one parameter (g_{NaT}) is distributed differently in the G1 target data than in the G2 target data, and the other four parameters have the same distribution in the G1 and G2 target data. *Upper main diagonal and upper triangle* - Four parameters (all parameters except g_{NaT}) are distributed differently in the G1 target data than in the G2 target data.

A.1.4 Output of cGAN Samples in Feature Space for Experimental Target Data

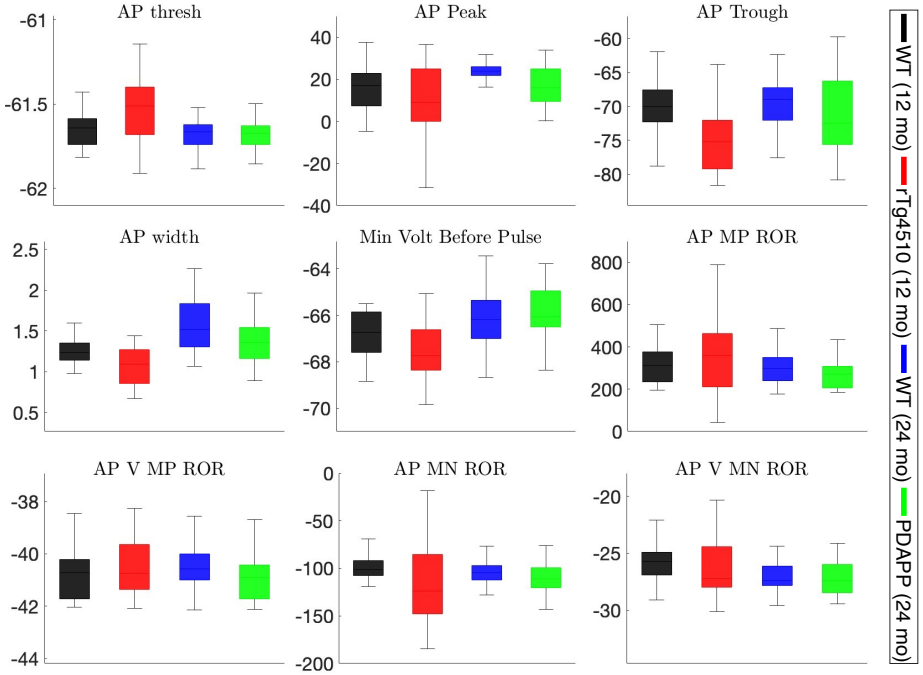

Figure A.10 Box and whisker plots of the action potential (AP) features extracted from the mechanistic model voltage traces obtained by pushing forward the cGAN parameter samples with experimental target data.

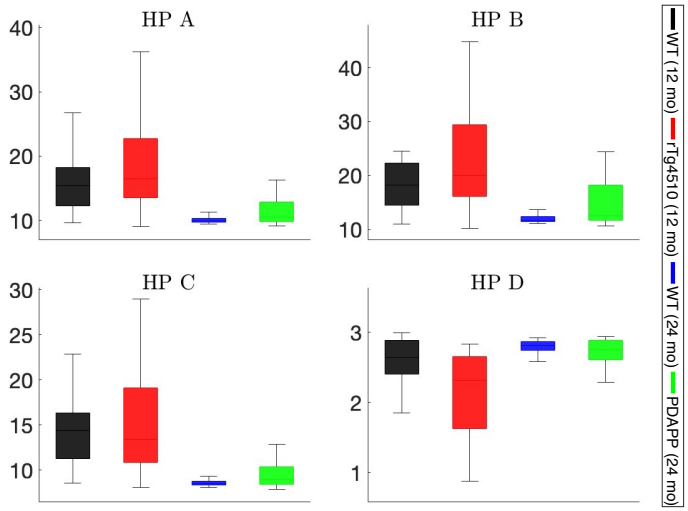

Figure A.11 Box and whisker plots of the membrane hyperpolarization (HP) features extracted from the mechanistic model voltage traces obtained by pushing forward the cGAN parameter samples with experimental target data.

A.2 Mechanistic Model Parameters

Table A.1 Parameters of CA1 Pyramidal Neuron Model

Parameter	Value	Units	Parameter	Value	Units
C_m	1	μF/cm^2	V_{m_H}	-102	mV
E_{Na}	60	mV	V_{n_H}	-102	mV
E_{Ca}	90	mV	$k_{m_{Na_T}}$	5	mV
E_K	-85	mV	$k_{h_{Na_T}}$	-7	mV
E_H	-30	mV	$k_{m_{Na_P}}$	3	mV
E_L	-65	mV	$k_{m_{Ca_T}}$	5	mV
g_{Na_T}	65.0 (7.2603)	μS/cm^2	$k_{h_{Ca_T}}$	-8.5	mV
g_{Na_P}	0.1 (0.0423)	μS/cm^2	$k_{m_{Ca_H}}$	5	mV
g_{Ca_T}	0.6 (0.067)	μS/cm^2	$k_{h_{Ca_H}}$	-7	mV
g_{Ca_H}	0.74 (1.5208)	μS/cm^2	$k_{m_{K_{DR}}}$	11.4	mV
$g_{K_{DR}}$	9.5 (12.505)	μS/cm^2	$k_{h_{K_{DR}}}$	-9.7	mV
g_{K_M}	0.8 (3.3837)	μS/cm^2	$k_{m_{K_M}}$	10	mV
g_H	0.05 (0.0503)	μS/cm^2	k_{m_H}	-13	mV
g_L	0.02 (0.0035)	μS/cm^2	k_{n_H}	-6	mV
$V_{m_{Na_T}}$	-37 (-60.00)	mV	$\tau_{m_{Ca_T}}$	2	ms
$V_{h_{Na_T}}$	-75	mV	$\tau_{h_{Ca_T}}$	32	ms
$V_{m_{Na_P}}$	-47	mV	$\tau_{m_{Ca_H}}$	0.08	ms
$V_{m_{Ca_T}}$	-54	mV	$\tau_{h_{Ca_H}}$	300	ms
$V_{h_{Ca_T}}$	-65	mV	$\tau_{m_{K_{DR}}}$	1	ms
$V_{m_{Ca_H}}$	-15	mV	$\tau_{h_{K_{DR}}}$	1400	ms
$V_{h_{Ca_H}}$	-60	mV	$\tau_{m_{K_M}}$	75	ms
$V_{m_{K_{DR}}}$	-5.8	mV	τ_{m_H}	15	ms
$V_{h_{K_{DR}}}$	-68	mV	τ_{n_H}	210	ms
$V_{m_{KM}}$	-30	mV	p	0.85	-

Note: For most parameters, we used the values given in Nowacki et al. [46] paper. Parameter values for the hyperpolarization-activated potassium current (I_H) are taken from Booth et al. [6]. Optimized parameter values for the maximal conductances and transient sodium half-activation that we obtained through differential evolution (DE-MG-Vmnat) are shown in parentheses.

REFERENCES

[1] S. L. Allam, T. H. Rumbell, T. Hoang-Trong, J. Parikh, and J. R. Kozloski. Neuronal population models reveal specific linear conductance controllers sufficient to rescue preclinical disease phenotypes. *iScience*, 24(11):103279, 2021.

[2] R. Allen, T. R. Rieger, and C. J. Musante. Efficient generation and selection of virtual populations in quantitative systems pharmacology models. *CPT: Pharmacometrics and Systems Pharmacology*, 5(3):140–146, 2016.

[3] E. Alpaydin. *Introduction to machine learning*. Cambridge, MA: MIT Press, 2020.

[4] Beatriz Bano-Otalora, Matthew J Moye, Timothy Brown, Robert J Lucas, Casey O Diekman, and Mino DC Belle. Daily electrical activity in the master circadian clock of a diurnal mammal. *Elife*, 10:e68179, 2021.

[5] M. D. Belle, C. O. Diekman, D. B. Forger, and H. D. Piggins. Daily electrical silencing in the mammalian circadian clock. *Science*, 326(5950):281–284, 2009.

[6] C. A. Booth, J. Witton, J. Nowacki, K. Tsaneva-Atanasova, M. W. Jones, A. D. Randall, and J. T. Brown. Altered intrinsic pyramidal neuron properties and pathway-specific synaptic dysfunction underlie aberrant hippocampal network function in a mouse model of tauopathy. *Journal of Neuroscience*, 36(2):350–363, 2016.

[7] O. J. Britton, A. Bueno-Orovio, K. Van Ammel, H. R. Lu, R. Towart, D. J. Gallacher, and B. Rodriguez. Experimentally calibrated population of models predicts and explains intersubject variability in cardiac cellular electrophysiology. *Proceedings of the National Academy of Sciences*, 110(23):E2098–E2105, 2013.

[8] S. Brooks. Markov chain Monte Carlo method and its application. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47(1):69–100, 1998.

[9] S. Brooks, A. Gelman, G. Jones, and X. Meng. *Handbook of Markov chain Monte Carlo*. CRC Press, Boca Raton: FL, 2011.

[10] T. Butler, J. Jakeman, and T. Wildey. Combining push-forward measures and Bayes' rule to construct consistent solutions to stochastic inverse problems. *SIAM Journal on Scientific Computing*, 40(2):A984–A1011, 2018.

[11] Y. Cheng, C. J. Thalhauser, S. Smithline, J. Pagidala, M. Miladinov, H. E. Vezina, M. Gupta, T. A. Leil, and B. J. Schmidt. QSP toolbox: Computational implementation of integrated workflow components for deploying multi-scale mechanistic models. *The American Association of Pharmaceutical Scientists Journal*, 19(4):1002–1016, 2017.

[12] V. L. Corbit, T. C. Whalen, K. T. Zitelli, S. Y. Crilly, J. E. Rubin, and A. H. Gittis. Pallidostriatal projections promote β oscillations in a dopamine-depleted biophysical network model. *Journal of Neuroscience*, 36(20):5556–5571, 2016.

[13] K. Cranmer, J. Brehmer, and G. Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020.

[14] C. O. Diekman, M. D. Belle, R. P. Irwin, C. N. Allen, H. D. Piggins, and D. B. Forger. Causes and consequences of hyperexcitation in central clock neurons. *PLoS Computational Biology*, 9(8):e1003196, 2013.

[15] C. O. Diekman and D. B. Forger. Clustering predicted by an electrophysiological model of the suprachiasmatic nucleus. *Journal of Biological Rhythms*, 24(4):322–333, 2009.

[16] G. B. Ermentrout and D. H. Terman. *Mathematical foundations of neuroscience*, volume 35. New York, NY: Springer Science and Business Media, 2010.

[17] M. Flourakis, E. Kula-Eversole, A. L. Hutchison, T. H. Han, K. Aranda, D. L. Moose, K. P. White, A. R. Dinner, B. C. Lear, D. Ren, C. O. Diekman, I. M. Raman, and R. Allada. A conserved bicycle model for circadian clock control of membrane excitability. *Cell*, 162(4):836–848, 2015.

[18] K. Gadkar, N. Budha, A. Baruch, J. Davis, P. Fielder, and S. Ramanujan. A mechanistic systems pharmacology model for prediction of LDL cholesterol lowering by PCSK9 antagonism in human dyslipidemic populations. *CPT: Pharmacometrics & Systems Pharmacology*, 3(11):1–9, 2014.

[19] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, Boca Raton: FL, 1995.

[20] P. J. Gonçalves, J. Lueckmann, M. Deistler, M. Nonnenmacher, K. Öcal, G. Bassetto, C. Chintaluri, W. F. Podlaski, S. A. Haddad, T. P. Vogels, D. S. Greenberg, and J. H. Macke. Training deep neural density estimators to identify mechanistic models of neural dynamics. *eLife*, 9:e56261, 2020.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[22] G. Harshvardhan, M. K. Gourisaria, M. Pandey, and S. S. Rautaray. A comprehensive survey and analysis of generative models in machine learning. *Computer Science Review*, 38:100285, 2020.

[23] J. Herman and W. Usher. SALib: An open-source Python library for sensitivity analysis. *Journal of Open Source Software*, 2(9):97, 2017.

[24] S. Hitawala. Comparative study on generative adversarial networks. *arXiv preprint arXiv:1801.04271*, 2018.

[25] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500, 1952.

[26] P. D. Hoff. *A first course in Bayesian statistical methods*, volume 580. New York: Springer, 2009.

[27] J. N. Hooker. *Hybrid modeling*. New York, NY: Springer, 2011.

[28] T. Iwanaga, W. Usher, and J. Herman. Toward SALib 2.0: Advancing the accessibility and interpretability of global sensitivity analyses. *Socio-Environmental Systems Modelling*, 4:18155–18155, 2022.

[29] E. M. Izhikevich. *Dynamical systems in neuroscience*. Cambridge, MA: MIT Press, 2007.

[30] M. Jamei, G. L. Dickinson, and A. Rostami-Hodjegan. A framework for assessing inter-individual variability in pharmacokinetics using virtual human populations and integrating general knowledge of physical chemistry, biology, anatomy, physiology and genetics: a tale of 'bottom-up' vs 'top-down' recognition of covariates. *Drug Metabolism and Pharmacokinetics*, 24(1):53–75, 2009.

[31] M. Jaxa-Rozen and J. Kwakkel. Tree-based ensemble methods for sensitivity analysis of environmental models: A performance comparison with sobol and morris techniques. *Environmental Modelling & Software*, 107:245–266, 2018.

[32] K. Konakli and B. Sudret. Global sensitivity analysis using low-rank tensor approximations. *Reliability Engineering & System Safety*, 156:64–83, 2016.

[33] M. C. Lancaster and E. Sobie. Improved prediction of drug-induced Torsades de Pointes through simulations of dynamics and machine learning algorithms. *Clinical Pharmacology & Therapeutics*, 100(4):371–379, 2016.

[34] B. A. Lawson, C. C. Drovandi, N. Cusimano, P. Burrage, B. Rodriguez, and K. Burrage. Unlocking data sets by calibrating populations of models to data density: A study in atrial electrophysiology. *Science Advances*, 4(1):e1701676, 2018.

[35] D. Lee, A. Jayaraman, and J. S. Kwon. Development of a hybrid model for a partially known intracellular signaling pathway through correction term estimation and neural network modeling. *PLoS Computational Biology*, 16(12):e1008472, 2020.

[36] B. J. Lence and A. Takyi. Data requirements for seasonal discharge programs: an application of a regionalized sensitivity analysis. *Water Resources Research*, 28(7):1781–1789, 1992.

[37] J. Lueckmann, G. Bassetto, T. Karaletsos, and J. H. Macke. Likelihood-free inference with emulator networks. In *Symposium on Advances in Approximate Bayesian Inference*, pages 32–53. Proceedings of Machine Learning Research, 2019.

[38] E. Marder and A. L. Taylor. Multiple models to capture the variability in biological neurons and networks. *Nature Neuroscience*, 14(2):133–138, 2011.

[39] I. Matei, J. de Kleer, A. Feldman, R. Rai, and S. Chowdhury. Hybrid modeling: Applications in real-time diagnosis. *arXiv preprint arXiv:2003.02671*, 2020.

[40] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[41] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

[42] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of machine learning*. Cambride, MA: MIT Press, 2018.

[43] C. Morris and H. Lecar. Voltage oscillations in the barnacle giant muscle fiber. *Biophysical Journal*, 35(1):193–213, 1981.

[44] A Muszkiewicz, A Bueno-Orovio, X Liu, B Casadei, and B Rodriguez. Constructing human atrial electrophysiological models mimicking a patient-specific cell group. In *Computing in Cardiology 2014*, pages 761–764, 2014.

[45] Harshith Nagaraj and Rishikesh Narayanan. Plasticity manifolds and degeneracy govern circadian oscillations of neuronal intrinsic properties in the suprachiasmatic nucleus. *iScience*, 2023.

[46] J. Nowacki, H. M. Osinga, J. T. Brown, A. D. Randall, and K. Tsaneva-Atanasova. A unified model of CA1/3 pyramidal cells: an investigation into excitability. *Progress in Biophysics and Molecular Biology*, 105(1-2):34–48, 2011.

[47] B. B. Otalora, M. J. Moye, T. M. Brown, R. J. Lucas, C. O. Diekman, and M. D. Belle. Daily electrical activity in the master circadian clock of a diurnal mammal. *bioRxiv*, 2020.

[48] J. Parikh, J. Kozloski, and V. Gurev. Integration of AI and mechanistic modeling in generative adversarial networks for stochastic inverse problems. *arXiv preprint arXiv:2009.08267*, 2020.

[49] J. Parikh, T. Rumbell, X. Butova, T. Myachina, J. C. Acero, S. Khamzin, O. Solovyova, J. Kozloski, A. Khokhlova, and V. Gurev. Generative adversarial networks for construction of virtual populations of mechanistic models: simulations to study omecamtiv mecarbil action. *Journal of Pharmacokinetics and Pharmacodynamics*, 49(1):51–64, 2022.

[50] E. Passini, A. Mincholé, R. Coppini, E. Cerbai, B. Rodriguez, S. Severi, and A. Bueno-Orovio. Mechanisms of pro-arrhythmic abnormalities in ventricular repolarisation and anti-arrhythmic therapies in human hypertrophic cardiomyopathy. *Journal of Molecular and Cellular Cardiology*, 96:72–81, 2016.

[51] A. Ponzi, S. J. Barton, K. D. Bunner, C. Rangel-Barajas, E. S. Zhang, B. R. Miller, G. V. Rebec, and J. Kozloski. Striatal network modeling in huntington's disease. *PLoS Computational Biology*, 16(4):e1007648, 2020.

[52] D. Poole and A. E. Raftery. Inference for deterministic simulation models: the Bayesian melding approach. *Journal of the American Statistical Association*, 95(452):1244–1255, 2000.

[53] K. Price, R. M. Storn, and J. A. Lampinen. *Differential evolution: A practical approach to global optimization*. Springer Science & Business Media, 53851, Lappeenranta, 2006.

[54] A. A. Prinz, D. Bucher, and E. Marder. Similar network activity from disparate circuit parameters. *Nature Neuroscience*, 7(12):1345–1352, 2004.

[55] T. R. Rieger, R. J. Allen, L. Bystricky, Y. Chen, G. W. Colopy, Y. Cui, A. Gonzalez, Y. Liu, R. White, R. Everett, H.T. Banks, and C. J. Musante. Improving the generation and selection of virtual populations in quantitative systems pharmacology models. *Progress in Biophysics and Molecular Biology*, 139:15–22, 2018.

[56] S. Saghafi, T. Rumbell, V. Gurev, J. Kozloski, F. Tamagnini, K. C. A. Wedgwood, and C. O. Diekman. Inferring parameters of pyramidal neuron excitability in mouse models of Alzheimer's disease using biophysical modeling and deep learning. *bioRxiv*, pages 2023–04, 2023.

[57] S. Saghafi and P. Sanaei. Dynamic Entrainment: A deep learning and data-driven process approach for synchronization in the Hodgkin-Huxley model. *bioRxiv*, pages 2023–04, 2023.

[58] C. Sánchez, A. Bueno-Orovio, E. Wettwer, S. Loose, J. Simon, U. Ravens, E. Pueyo, and B. Rodriguez. Inter-subject variability in human atrial action potential in sinus rhythm versus chronic atrial fibrillation. *PLoS One*, 9(8):e105897, 2014.

[59] B. Schölkopf and A. J. Smola. *A short introduction to learning with kernels*. Berlin, Germany: Springer, 2003.

[60] E. A. Sobie. Parameter sensitivity analysis in electrophysiological models using multivariable regression. *Biophysical Journal*, 96(4):1264–1274, 2009.

[61] I. M. Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Computers in Simulation*, 55(1-3):271–280, 2001.

[62] R. Storn and K. Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.

[63] F. Tamagnini, J. Novelia, T. L. Kerrigan, J. T. Brown, K. Tsaneva-Atanasova, and A. D. Randall. Altered intrinsic excitability of hippocampal CA1 pyramidal neurons in aged PDAPP mice. *Frontiers in Cellular Neuroscience*, 9:372, 2015.

[64] L. Theis, A. v. Oord, and M. Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

[65] M. Tosin, A. Côrtes, and A. Cunha. A Tutorial on Sobol Global Sensitivity Analysis Applied to Biological Models. *Networks in Systems Biology*, pages 93–118, 2020.

[66] N Yang. *NALCN-Encoded Sodium Leak Currents in the Regulation of Daily Rhythms in the Excitability of Neurons in the Suprachiasmatic Nucleus*. PhD thesis, Washington University in St. Louis, 2023.