Inferring parameters of pyramidal neuron excitability in mouse models of Alzheimer's disease using biophysical modeling and deep learning

Soheil Saghafi[1], Timothy Rumbell[2], Viatcheslav Gurev[2], James Kozloski[2], Francesco Tamagnini[3], Kyle C. A. Wedgwood[4] and Casey O. Diekman[1]

[1]Department of Mathematical Sciences, New Jersey Institute of Technology, University Heights, Newark, 07102, New Jersey, USA.
[2]IBM T.J. Watson Research Center, Yorktown Heights, 10598, NY, USA.
[3]Pharmacology, University of Reading, Reading, UK.
[4]Mathematics and Statistics, University of Exeter, Exeter, UK.

Abstract

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. We introduce a novel parameter inference technique, deep hybrid modeling (DeepHM), that combines 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. Although mechanistic modeling and machine learning methods are by themselves powerful tools for approximating biological systems and making accurate predictions from data, 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. DeepHM addresses these shortcomings by using conditional generative adversarial networks (cGANs) to provide an inverse

mapping of data to mechanistic models that identifies the distributions of mechanistic modeling parameters coherent to the data. Here, we demonstrate that DeepHM accurately infers parameter distributions of the conductance-based model and outperforms a Markov chain Monte Carlo method on several test cases using synthetic data. We then 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 find that the conductances most disrupted by tauopathy, amyloidopathy, and aging are delayed rectifier potassium, transient sodium, and hyperpolarization-activated potassium, respectively.

Keywords: Pyramidal Neuron Excitability, Parameter Inference, Generative Adversarial Network, Population of Models

1 Introduction

Although the underlying cause of Alzheimer's disease (AD) remains poorly understood, it is believed to occur when abnormal amounts of the proteins amyloid beta and tau aggregate in the brain, forming extracellular plaques (amyloidopathy) and neurofibrillary tangles (tauopathy) that result in a progressive loss of neuronal function and dementia [1, 2]. In transgenic mice with amyloidopathy, neurons in the hippocampus—a brain structure critical for memory—exhibit altered intrinsic excitability properties, such as action potentials with reduced peaks and widths [3–5]. Hippocampal neurons in transgenic mice with tauopathy also show altered excitability, but in different properties such as hyperpolarization-activated membrane potential sag and action potential threshold [6, 7].

Ideally, biophysical modeling could be used to gain insights into the mechanisms underlying the disrupted electrophysiological properties of these Alzheimer's mutant mice. However, determining whether or not such a biophysical model and its outputs are coherent with a set of experimental observations is a major challenge since such 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 [8]. On the other hand, neither purely statistical models with tractable likelihoods nor purely data-driven machine learning algorithms offer much insight into underlying biological mechanisms [9, 10]. 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 [9, 11–16]. 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, 17–19]. 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 retraining [20].

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, cryptography, and neuroscience [21–23]. Several extensions of GANs have been developed to address particular tasks [24–26]. To solve SIPs, we use a conditional GAN (cGAN) structure [27] where the generator is trained with parameter sets X conditioned on the output features Y of a mechanistic model.

In this paper, we wish to solve an SIP to identify which ion channels are responsible for the altered excitability properties of hippocampal neurons in mouse models of amyloidopathy and tauopathy. The data for the SIP of interest in this paper are voltage traces recorded from hippocampal CA1 neurons in 12-month-old rTg4510 mice expressing pathogenic tau (Tamagnini et al., unpublished data), 24-month-old PDAPP mice overexpressing amyloid beta [3], and age-matched wildtype littermate controls for each transgenic phenotype. From these traces, we extract several electrophysiological features (such as action potential peak, width, and threshold) that capture the excitability properties of the cells. Neuronal excitability can be simulated using the conductance-based modeling formalism originally developed by Hodgkin and Huxley [28]. The mechanistic model for the SIP of interest in this paper is a conductance-based model of CA1 neurons that has been shown to be capable of reproducing key electrophysiological features of recorded voltage traces [6, 29]. By solving this SIP, we will map the features of the recorded voltage traces in the AD mutant and wildtype mice to the parameter space of the CA1 model. Our goal is to use the resulting parameter distributions to

infer which ion channel conductances are disrupted in the amyloidopathy and tauopathy mice compared to their age-matched wildtype controls, and which conductances change with age in the wildtype mice.

The remainder of the paper is organized as follows. In Section 2, we describe the experimental data and the features we extract from the recorded action potentials and hyperpolarization traces. In Section 3, we introduce a biophysical model of CA1 pyramidal neurons and our initial optimizations of the model parameters using differential evolution. In Section 4, we give a brief description of GANs and cGANs and illustrate our parameter inference methodology using the Rosenbrock function as a toy model. In Section 5, we train the cGAN on output of the CA1 model and then present it with synthetic target data. We show that cGAN outperforms a benchmark MCMC method on a relatively simple parameter inference task. We then validate its ability to accurately infer complex parameter distributions through a series of tests with synthetic target data. In Section 6, we present the trained cGAN with real target data and use the inferred parameter distributions to identify which ionic conductances are affected by age, amyloidopathy, and tauopathy. We conclude with a discussion of alternative methods and future work in Section 7.

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. [3], 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 paper, 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; Fig. 1A) 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; Fig. 1B). 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 Figs. 2A-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
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 Fig. 2C and are defined as follows:

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
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 Fig. 3 for the AP features, and Fig. 4 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 controls (Fig. 3 top middle panel) and AP width appears to be reduced in rTg4510 mice compared to their WT controls (Fig. 3 middle left panel).

3 Biophysical Model

CA1 pyramidal neuron model

Conductance-based modeling to describe the electrical activity of neurons was introduced by Hodgkin and Huxley in 1952 to explain the ionic mechanisms underlying the initiation and propagation of action potentials (APs) in the squid giant axon [30]. Nowacki et al [29] 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:

$$C\frac{dV}{dt} = I_{app} - I_{Na_T} - I_{Na_P} - I_{Ca_T} - I_{Ca_H} - I_{K_{DR}} - I_{K_M} - I_L - I_{K_H} \quad (1)$$

$$\frac{dx}{dt} = \frac{x_\infty - x}{\tau_x} \quad (2)$$

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:

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

First, we simulated the pyramidal neuron model with the parameter values provided in Nowacki et al [29] (Supplementary Table 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 Figs. 3 and 4). For example, the AP threshold and AP peak in the model are significantly more depolarized than the AP thresholds and peaks seen in these CA1 neurons (Fig. 3 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 [31, 32].

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 the Supplementary Methods.

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 Figs. 3 and 4). However, this model's AP threshold was still significantly more depolarized than in the data (Fig. 3 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 Supplementary Methods 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 Figs. 3 and 4). 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 (Fig. 5).

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. The default parameters are provided in Supplementary Table 1.

4 Parameter Inference Methodology: conditional Generative Adversarial Networks

4.1 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 is to map the density of observed data ($\mathcal{P}_Y$) 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 ($\hat{\mathcal{P}}_Y$) satisfies $\hat{\mathcal{P}}_Y \sim \mathcal{P}_Y$, and (2) α_X covers all possible solutions in the range described by the prior in input space ($\mathcal{P}_X$). In this section, we will first introduce standard GANs, and then move on to conditional GANs (cGANs) that 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 (Fig. 6). 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 Fig. 6). 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 the generator and discriminator one at a time would not only be time consuming, but also lead to instability due to a vanishing gradient for the loss function of 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:

$$L(\hat{y}, y) = [y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})] \tag{3}$$

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) we can get:

$$L(D(x), 1) = \log(D(x)) \tag{4}$$

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 Eqn. (3) we end up with:

$$L(D(G(z)), 0) = \log(1 - D(G(z))). \tag{5}$$

Panels A and B in Figure (7) are the visualization of Eqns. (4) and (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 A, $\log(D(x))$ 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 B, 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 Eqns. (4) and (5), the objective function for the Discriminator is:

$$\max_D \{\log D(x) + \log(1 - D(G(z)))\}. \tag{6}$$

The same logic also holds in the case of having a strong Generator, the only difference here is that we are going to minimize Eqn. (5), which corresponds to the right panel of Fig. 7. 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 Eqn. (5). Thus, the objective function for the Generator is:

$$\min_G \{\log(1 - D(G(z)))\} = \min_G \{\log D(x) + \log(1 - D(G(z)))\}. \tag{7}$$

In order to obtain a single objective function for the GAN, we combine Eqns. (6) and (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:

$$\min_G \max_D \{\mathbb{E}_{x \sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim P_z(z)}[\log(1 - D(G(z)))]\}. \tag{8}$$

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, the Discriminator can easily distinguish the real and the fake samples. This causes $\log(1 - D(G(z)))$ to saturate (see Fig. 7B), 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.

4.2 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 (cGANs) [27], 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 Fig. 8. The objective function of the cGAN is:

$$\begin{cases} \min\limits_{D} & \{-\mathbb{E}_{x\sim P_{data}(x)}[\log D(x\|y)] - \mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z\|y)))]\} \\ \min\limits_{G} & \{\mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z\|y)))] - \mathbb{E}_{z\sim P_z(z)}[\log(D(G(z\|y)))]\} \end{cases} \tag{9}$$

4.3 Illustration of cGAN training process

We used the Rosenbrock function (Eqn. 10) as a toy mechanistic model to illustrate the training process for cGAN:

$$Y = (1 - X_1)^2 + 100(X_2 - X_1^2)^2. \tag{10}$$

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 (9) 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 (see Supplementary Methods for more details on our JSD computation). 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 (Fig. 9 bottom right panel). This phenomenon occurs due to the vanishing gradient problem.

5 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 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 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 (Fig. 10). 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.

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 $\pm 100\%$ 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 (Fig. 11A). 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. 11B). 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}).

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

Although the results shown in Fig. 11B 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 (Figs. 12 and 13A 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 (Fig. 13B 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 the Supplementary Methods. 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 [11, 33]. 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 (Figs. 12 and 13 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.

5.2 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 (Fig. S8 lower triangle), whereas the G1 and G2 membrane hyperpolarization feature target distributions are similar (Fig. S9 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 14 (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 (Supplementary Fig. S2 top panels). For the cGAN sample vs. target data KS tests, the null was rejected 0 out of 72 times for G1 (Fig. S2 bottom left panel) and 2 out of 72 times for G2 (Fig. S2 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 (Fig. S8 upper triangle), membrane hyperpolarization features (Fig. S9 upper triangle), and parameters (Fig. 14 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 (Fig. S5 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 (Fig. S5 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 Fig. 14 (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 (Fig. S5 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 (Fig. S5 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

$$\binom{5}{k} \times 2^{k-1}, \quad k = 1, \cdots, 5 \tag{11}$$

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 (Fig. S3 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 (Fig. S3 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 (Fig. S4 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 (Fig. S4 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 (Fig. S6 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 (Fig. S6 lower portions). The results of the KS tests for all of the 5 choose k cases are summarized in Fig. S7.

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 experimental data in Section 6.

6 Parameter Inference on Experimental Target 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 (Figs. 1, 3, 4) 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 15A 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, Fig. 15B 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

4 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 Fig. S10 to Fig. 3, and Fig. S11 to Fig. 4).

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 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 Fig. 16), 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 Fig. 16). 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 Fig. 17), 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 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 Fig. 17). These results lead us to hypothesize that these 3 conductances underlie the changes in excitability properties observed with aging.

7 Discussion

The last decade has seen a rise in the application of population-based modeling in the neuronal and cardiac electrophysiology domains [13–15, 34–37]. 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 [16, 18, 19, 38, 39].

Here, we have introduced and employed a deep hybrid modeling (DeepHM) framework [17, 20] featuring conditional generative adversarial networks (cGANs) that can be categorized as a population of models technique. We compared the performance of cGAN and a standard Bayesian inference Markov chain Monte Carlo (MCMC) method [40] 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. Since the cGAN was able to accurately detect a variety of complex differences in the distributions of parameters across 2 groups of synthetic target data, we employed the cGAN to infer the parameter distributions across 4 groups of experimental target data (WT and mutant mice at 2 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 WT versus mutant and younger versus older mice. These results illustrate the value of mapping experimental data back to the parameter space of a mechanistic model. In future work, the predictions we made about the role of certain conductances in Alzheimer's disease and aging phenotypes can be tested experimentally.

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 [41].

In this paper, 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 and a 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 [9, 42], to our knowledge our work is among the first to apply a deep learning approach to an SIP.

Acknowledgments

This work was partially supported by NSF DMS grants 1555237 and 2152115 to COD. KCAW was graciously supported by the EPSRC New Horizons grant (EP/V048716/1) and the EPSRC Hub for Quantitative Modeling in Healthcare (EP/T017856/1). The PDAPP data were collected under the Medical Research Council grant G1100623 (FT), and the rTg4510 data under the Alzheimer's Society Junior Fellowship grant AS-JF-14-007 (FT). KCAW and FT received support from the the Wellcome Trust ISSF2 Centre grant WT105618MA.

Data Availability Statement

The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

Figures

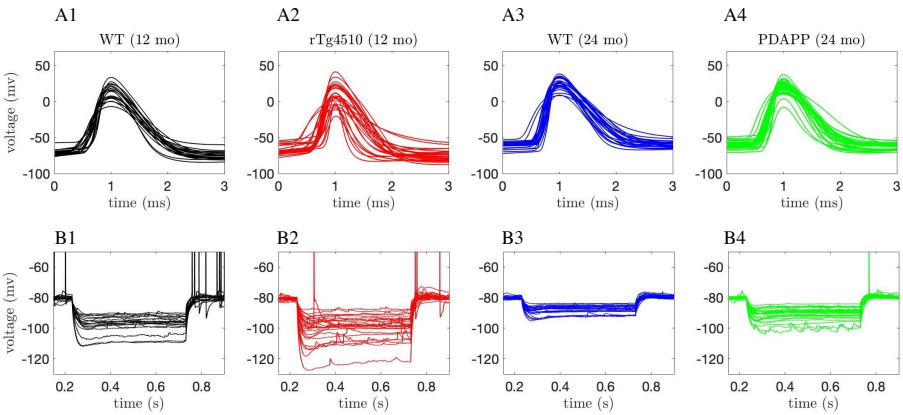

Fig. 1 Experimental recordings. Waveforms of the first action potential in response to depolarizing current pulses (**A1-A4**) and voltage traces in response to hyperpolarizing current pulses (**B1-B4**) injected into CA1 pyramidal neurons for 4 different categories of mice: Wild-type (WT) 12-month-old 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**).

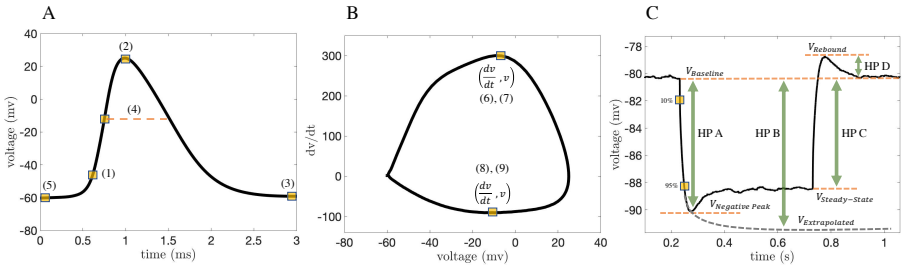

Fig. 2 Schematic of feature extraction. (A-B) Action potential features: (1) AP threshold, (2) AP peak, (3) AP trough, (4) AP width, (5) AP min voltage before the pulse, (6) AP max positive rate of rise, (7) AP voltage at max positive rate of rise, (8) AP max negative rate of rise, (9) AP voltage at 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.

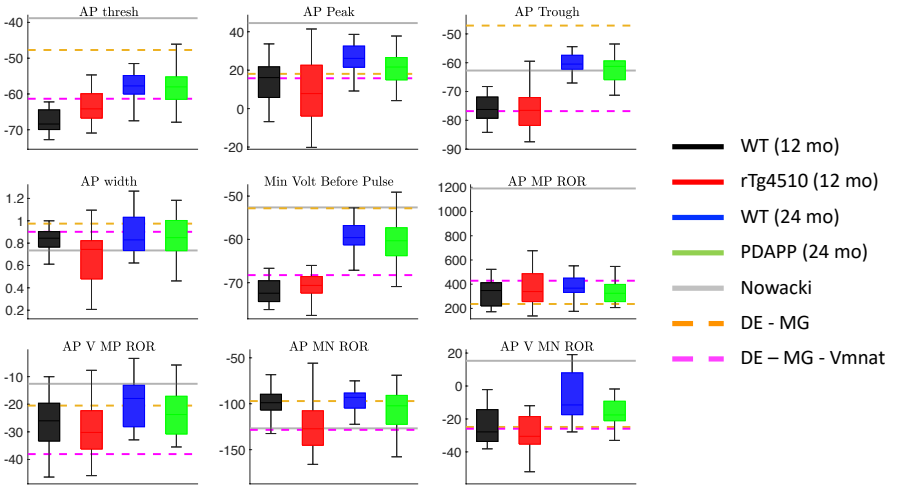

Fig. 3 Action potential features in experimental data and initial models. Box and whisker plots of the action potential (AP) features extracted from the experimental data and the biophysical CA1 model with three different parameter sets: (1) the parameters in the original Nowacki et al. paper ([29], 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).

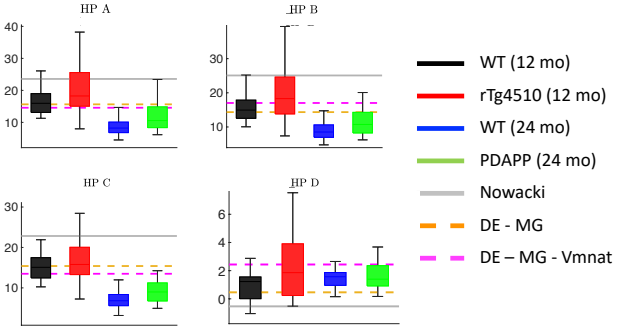

Fig. 4 Membrane hyperpolarization features in experimental data and initial models. Box and whisker plots of the membrane hyperpolarization features extracted from the experimental data and the biophysical CA1 model with three different parameter sets: (1) the parameters in the original Nowacki et al. paper ([29], 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).

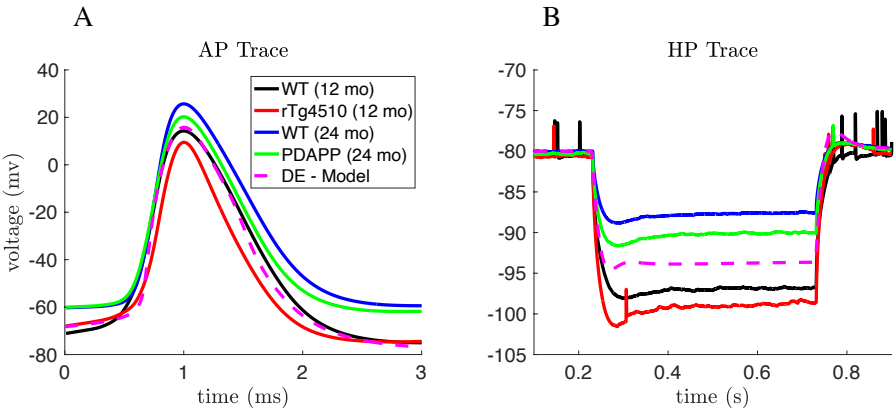

Fig. 5 Average AP and hyperpolarization voltage traces from experimental data and optimized model. The DE-Model shown here is the same model that was labeled DE-MG-Vmnat in Fig. 4, and was obtained by minimizing the least square error between the DE-Model output and the average AP and hyperpolarization traces across all 4 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).

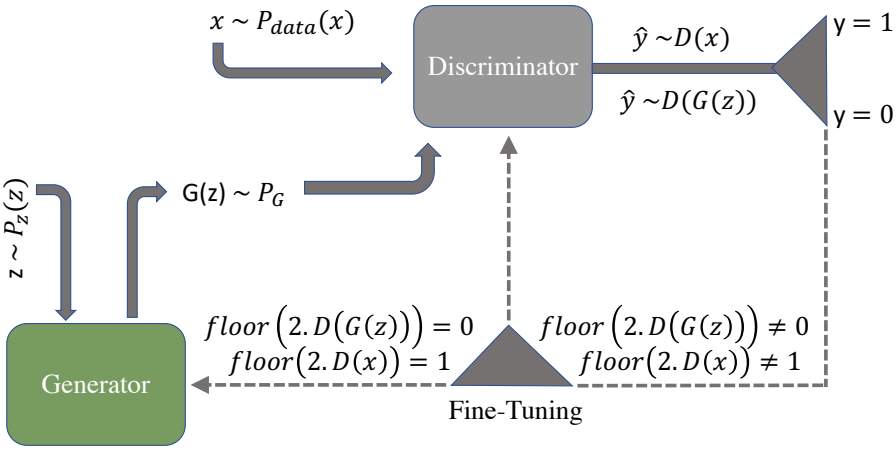

Fig. 6 Schematic of a generative adversarial network (GAN). The Generator (G) and Discriminator (D) are two neural networks that compete with each other during the training process, with the end result being that G can produce fake samples from a distribution that matches the distribution of the real 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.

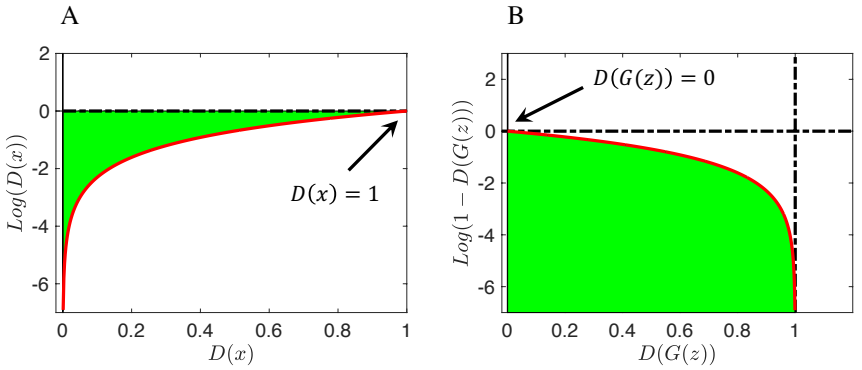

Fig. 7 Visualization of the components of the GAN objective function (Eqn. (8)). **(A)** Discriminator loss function (Eqn. (4)). $D(x)$ is the probability that x came from the real data rather than P_g. **(B)** Generator loss function given (Eqn. (5)). $D(G(z))$ is the probability that $G(z)$ came from P_g rather than P_{data}. The green shading identifies the possible regions for these two probabilities.

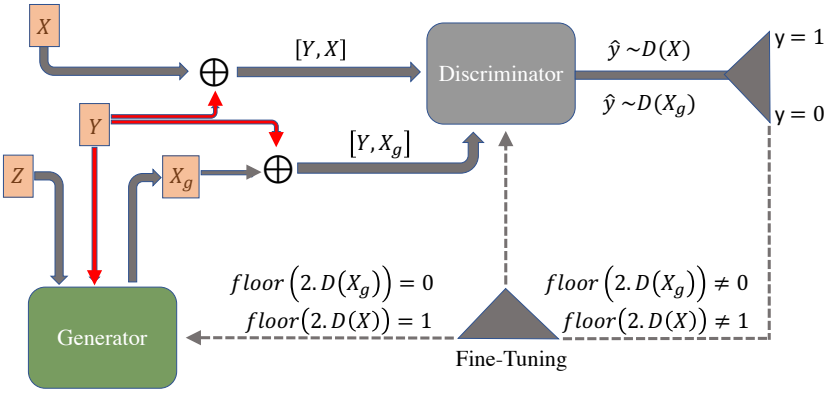

Fig. 8 Schematic of a conditional GAN (cGAN). Features of the training dataset (Y) are passed as a condition into the Generator (G) - already initialized with a Gaussian distribution (Z) - in order to produce some samples X_g given that condition $[Y, X_g]$. This output, along with real samples X augmented 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.

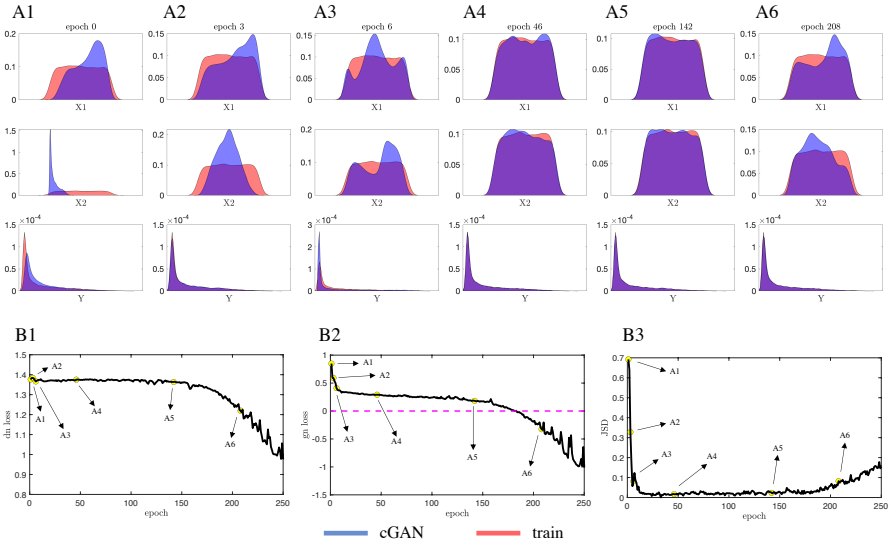

Fig. 9 Illustration of the cGAN training process on the Rosenbrock function (Eqn. 10). Over the course of the training process, both the discriminator and the generator improve. The discriminator gets better at distinguishing between real (coming from P_{data}) and fake (coming from P_g) samples, and 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.

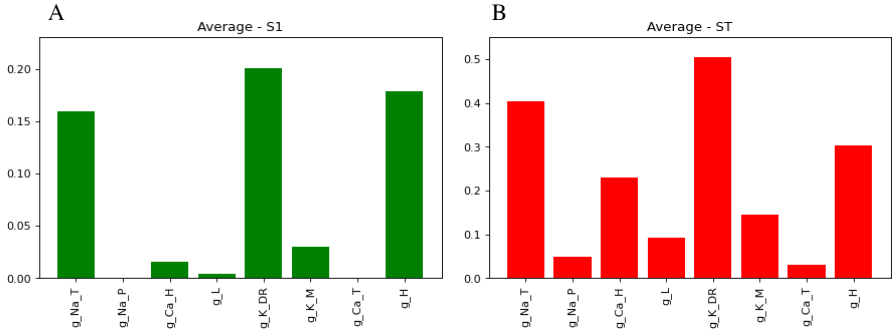

Fig. 10 Dimensionality reduction using variance-based (Sobol) sensitivity analysis. The height of the bars represent how sensitive the AP and hyperpolarization features are to each parameter. **(A)** S1: average first-order index across feature space. **(B)** ST: average total-effect index across feature space.

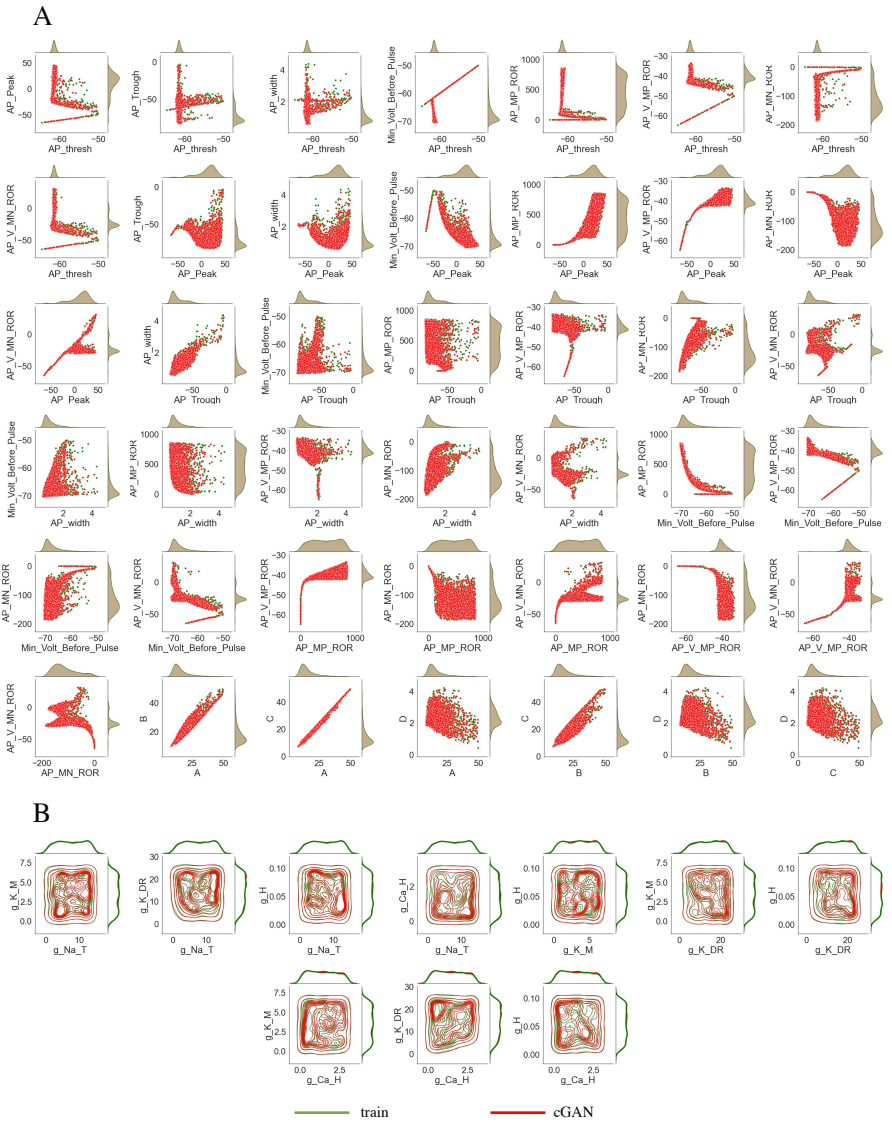

Fig. 11 Comparison of cGAN samples and training dataset. (A) Feature space: scatterplots (center of panel) and KDE plots (top and right of each panel) with cGAN samples in red and the training dataset in green. **(B)** Parameter space: contour plots (center of panel) and KDE plots (top and right of each 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).

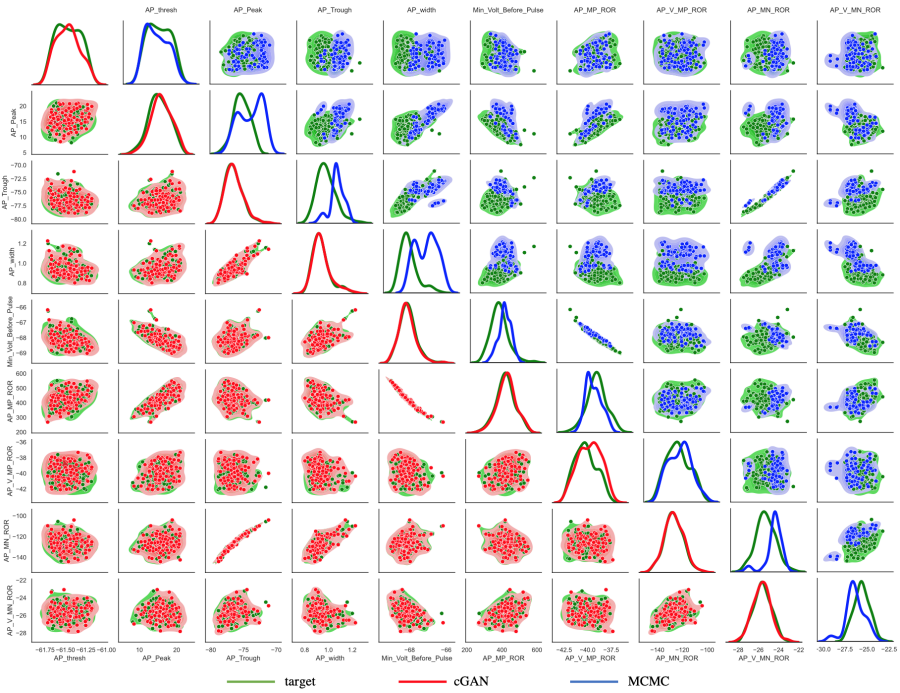

Fig. 12 Performance of cGAN and MCMC on synthetic target data - AP features. *Lower main diagonal and lower triangle* - KDE and scatter plots of the cGAN samples (red) versus target (green). *Upper main diagonal and upper triangle* - KDE and scatter plots of the MCMC samples (blue) versus target (green).

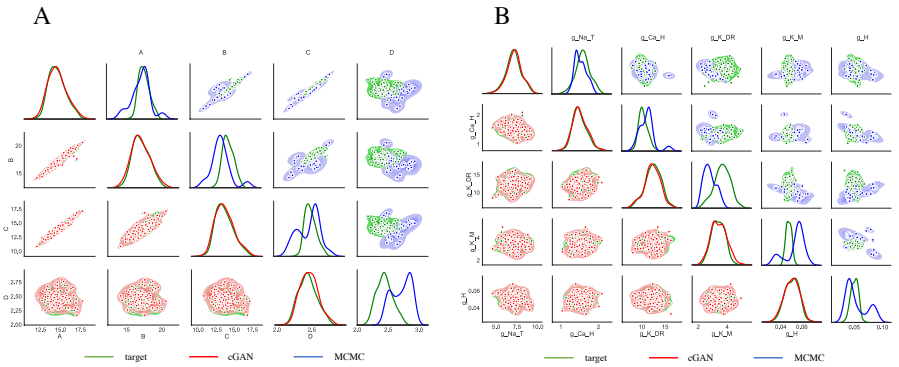

Fig. 13 Performance of cGAN and MCMC on synthetic target data - HP features and parameter space. *Lower main diagonal and lower triangle* - KDE and scatter plots of the cGAN samples (red) versus target (green). *Upper main diagonal and upper triangle* - KDE and scatter plots of the MCMC samples (blue) versus target (green). **(A)** HP features. **(B)** Parameter space.

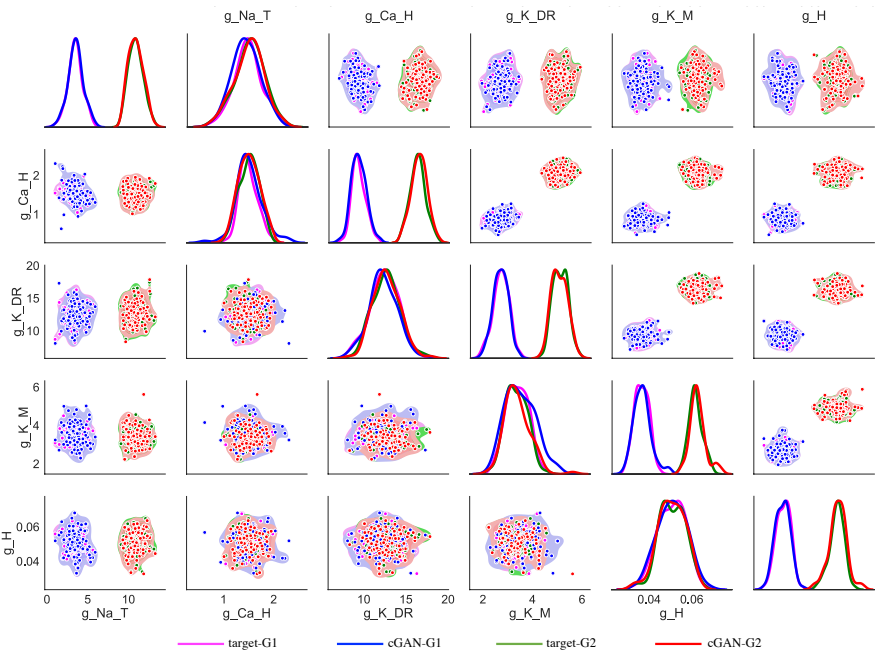

Fig. 14 Performance of cGAN in parameter space on synthetic targets from 2 groups with distinct parameter structures. KDE plots (main diagonals) and scatter plots (lower and upper triangles) for Group 1 (G1) target data (magenta), Group 2 (G2) target data (green), cGAN samples for G1 (blue) and cGAN samples for G2 (red). *Lower main diagonal and lower triangle* - only 1 parameter (g_{NaT}) is distributed differently in the G1 target data than in the G2 target data, and the other 4 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.

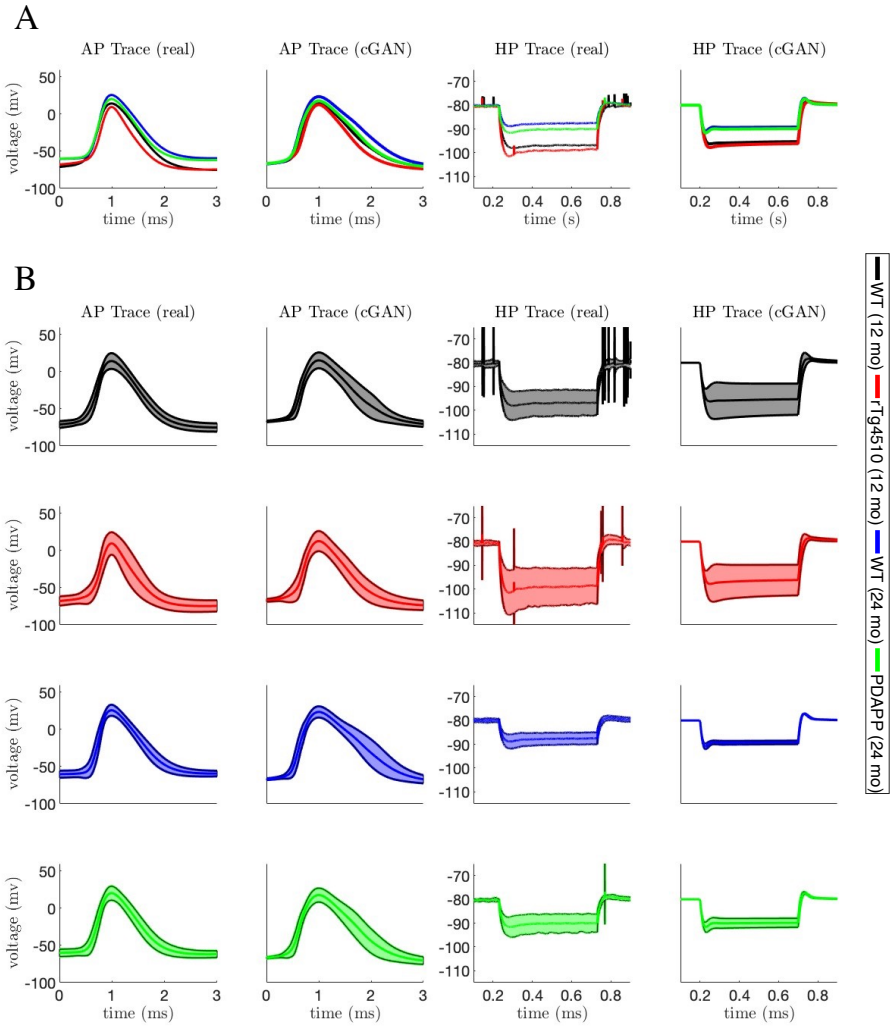

Fig. 15 ***AP and membrane hyperpolarization traces from cGAN samples with experimental targets.*** **(A)** Mean AP and membrane hyperpolarization traces from each experimental data category (1st and 3rd panels) and mean AP and membrane hyperpolarization traces from simulated the mechanistic model with 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.

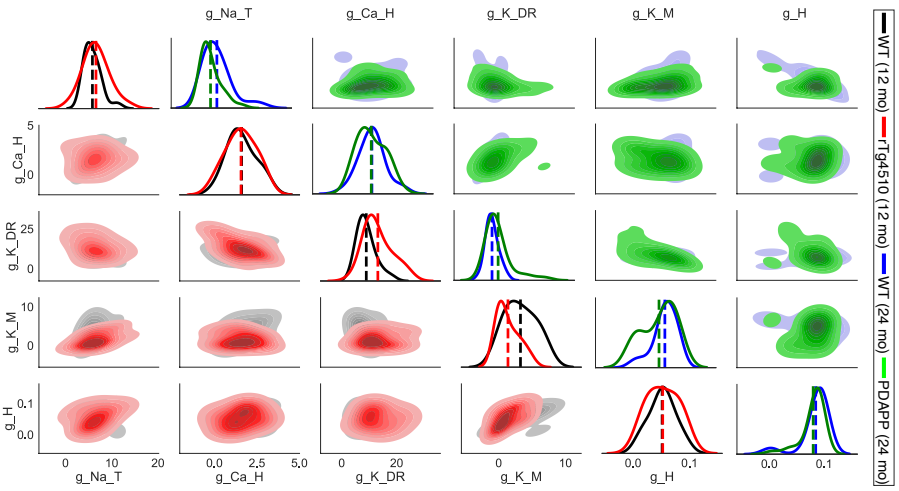

Fig. 16 Disease effect - parameter distributions from cGAN samples with experimental targets. *Lower main diagonal and lower triangle* - KDE and shaded contour plots of cGAN samples for 12-month-old WT (black) and 12-month-old tau mutant (rTg4510, red) mice. *Upper main diagonal and upper triangle* - KDE and shaded contour plots of cGAN samples for 24-month-old (blue) and 24-month-old amyloid beta mutant (PDAPP, green) mice.

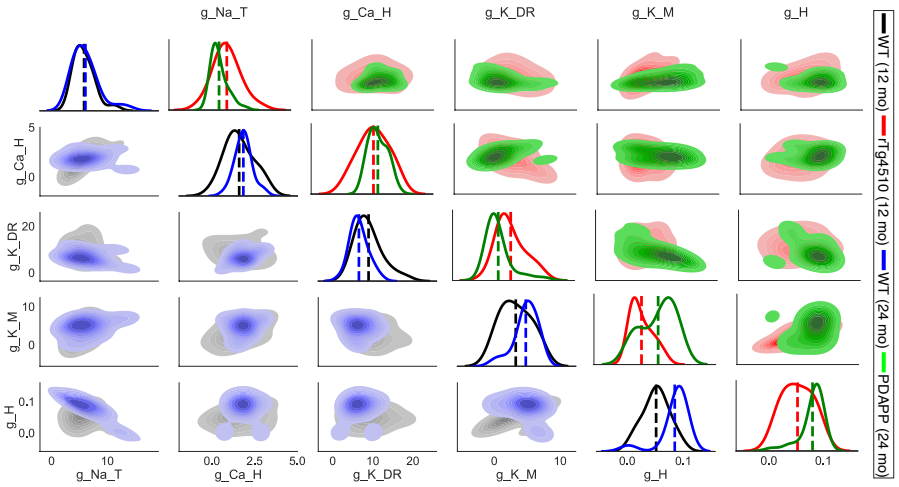

Fig. 17 Age effect - parameter distributions from cGAN samples with experimental targets. *Lower main diagonal and lower triangle* - KDE and shaded contour plots of cGAN samples for 12-month-old (black) and 24-month-old (blue) WT mice. *Upper main diagonal and upper triangle* - KDE and shaded contour plots of cGAN samples for 12-month-old (rTg4510, red) and 24-month-old (PDAPP, green) mutant mice.

References

[1] Hardy, J.: The amyloid hypothesis for Alzheimer's disease: a critical reappraisal. Journal of Neurochemistry **110**(4), 1129–1134 (2009)

[2] Spillantini, M.G., Goedert, M.: Tau pathology and neurodegeneration. The Lancet Neurology **12**(6), 609–622 (2013)

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

[4] Tamagnini, F., Scullion, S., Brown, J.T., Randall, A.D.: Intrinsic excitability changes induced by acute treatment of hippocampal CA1 pyramidal neurons with exogenous amyloid β peptide. Hippocampus **25**(7), 786–797 (2015)

[5] Vitale, P., Salgueiro-Pereira, A.R., Lupascu, C.A., Willem, M., Migliore, R., Migliore, M., Marie, H.: Analysis of age-dependent alterations in excitability properties of CA1 pyramidal neurons in an APPPS1 Model of Alzheimer's disease. Frontiers in Aging Neuroscience **13**, 668948 (2021)

[6] Booth, C.A., Witton, J., Nowacki, J., Tsaneva-Atanasova, K., Jones, M.W., Randall, A.D., Brown, J.T.: 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] Tamagnini, F., Walsh, D.A., Brown, J.T., Bondulich, M.K., Hanger, D.P., Randall, A.D.: Hippocampal neurophysiology is modified by a disease-associated C-terminal fragment of tau protein. Neurobiology of Aging **60**, 44–56 (2017)

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

[9] Gonçalves, P.J., Lueckmann, J.-M., Deistler, M., Nonnenmacher, M., Öcal, K., Bassetto, G., Chintaluri, C., Podlaski, W.F., Haddad, S.A., Vogels, T.P., *et al.*: Training deep neural density estimators to identify mechanistic models of neural dynamics. elife **9**, 56261 (2020)

[10] Panahi, M.R., Abrevaya, G., Gagnon-Audet, J.-C., Voleti, V., Rish, I., Dumas, G.: Generative Models of Brain Dynamics–A review. arXiv Preprint arXiv:2112.12147 (2021)

[11] Lawson, B.A., Drovandi, C.C., Cusimano, N., Burrage, P., Rodriguez, B., Burrage, K.: Unlocking data sets by calibrating populations of models to data density: A study in atrial electrophysiology. Science Advances **4**(1), 1701676 (2018)

[12] Britton, O.J., Bueno-Orovio, A., Van Ammel, K., Lu, H.R., Towart, R., Gallacher, D.J., Rodriguez, B.: Experimentally calibrated population of models predicts and explains intersubject variability in cardiac cellular electrophysiology. Proceedings of the National Academy of Sciences **110**(23), 2098–2105 (2013)

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

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

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

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

[17] Parikh, J., Rumbell, T., Butova, X., Myachina, T., Acero, J.C., Khamzin, S., Solovyova, O., Kozloski, J., Khokhlova, A., Gurev, V.: 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)

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

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

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

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

[22] Hitawala, S.: Comparative study on generative adversarial networks. arXiv preprint arXiv:1801.04271 (2018)

[23] Molano-Mazon, M., Onken, A., Piasini, E., Panzeri, S.: Synthesizing Realistic Neural Population Activity Patterns Using Generative Adversarial Networks. arXiv Preprint arXiv:1803.00338 (2018)

[24] Rosenfeld, B., Simeone, O., Rajendran, B.: Spiking Generative Adversarial Networks With a Neural Network Discriminator: Local training, Bayesian models, and Continual Meta-Learning. IEEE Transactions on Computers **71**(11), 2778–2791 (2022)

[25] Butte, S., Wang, H., Xian, M., Vakanski, A.: Sharp-GAN: Sharpness loss regularized GAN for histopathology image synthesis. In: 2022 IEEE 19th International Symposium on Biomedical Imaging (ISBI), pp. 1–5 (2022). IEEE

[26] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved Training of Wasserstein GANs. Advances in Neural Information Processing Systems **30** (2017)

[27] Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)

[28] Kass, R.E., Amari, S.-I., Arai, K., Brown, E.N., Diekman, C.O., Diesmann, M., Doiron, B., Eden, U.T., Fairhall, A.L., Fiddyment, G.M., *et al.*: Computational neuroscience: Mathematical and statistical perspectives. Annual Review of Statistics and Its Application **5**, 183 (2018)

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

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

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

[32] Price, K., Storn, R.M., Lampinen, J.A.: Differential Evolution: A Practical

Approach to Global Optimization. Springer, 53851, Lappeenranta (2006)

[33] Butler, T., Jakeman, J., Wildey, T.: Combining push-forward measures and Bayes' rule to construct consistent solutions to stochastic inverse problems. SIAM Journal on Scientific Computing **40**(2), 984–1011 (2018)

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

[35] Muszkiewicz, A., Bueno-Orovio, A., Liu, X., Casadei, B., Rodriguez, B.: Constructing human atrial electrophysiological models mimicking a patient-specific cell group. In: Computing in Cardiology 2014, pp. 761–764 (2014)

[36] Passini, E., Mincholé, A., Coppini, R., Cerbai, E., Rodriguez, B., Severi, S., Bueno-Orovio, A.: 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)

[37] Sánchez, C., Bueno-Orovio, A., Wettwer, E., Loose, S., Simon, J., Ravens, U., Pueyo, E., Rodriguez, B.: Inter-subject variability in human atrial action potential in sinus rhythm versus chronic atrial fibrillation. PLoS One **9**(8), 105897 (2014)

[38] Gadkar, K., Budha, N., Baruch, A., Davis, J., Fielder, P., Ramanujan, S.: 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)

[39] Jamei, M., Dickinson, G.L., Rostami-Hodjegan, A.: 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)

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

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

[42] Lueckmann, J., Bassetto, G., Karaletsos, T., Macke, J.: Likelihood-free inference with emulator networks. arxiv e-prints. arXiv preprint arXiv:1805.09294 (2019)

Supplementary Information

Saghafi et al. (2023). *Inferring parameters of pyramidal neuron excitability in mouse models of Alzheimer's disease using biophysical modeling and deep learning*

1 Mechanistic Model Parameters

Table 1 Parameters of CA1 pyramidal neuron model. For most parameters, we used the values given in Nowacki et al. [1] paper. Parameter values for the hyperpolarization-activated potassium current (I_H) are taken from Booth et al. [2]. 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.

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	-

2 Supplementary Methods

2.1 Differential Evolution (DE) - global optimization

Differential evolution is a type of stochastic global optimization and a population-based search technique first introduced by Storn

and Price, 1997 [3, 4]. 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 number 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 donor vector with the best member of the previous generation with a probability rate bring both exploration and exploitation to the search and lead the system get out of the local minima. *Selection* - during the selection step, we have a new vector, which is a candidate for going to the new generation in replace of the target vector, if it satisfies certain conditions, and 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 getting worse than before.

The Differential Evolution Algorithm that we used in this project known as **classic DE** in [3]. 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.

2.2 Sobol indices - global sensitivity analysis

Sensitivity Analysis (SA) [5–9] addresses the question of how much the output of a model changes given a change in the input. There are two main classes of 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 consider the behavior of the system

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

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 [10, 11]. 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, this correspond to certain parameters in the model which this probability distribution reflect 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 be 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:

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

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 satisfying the orthogonality property, which includes:

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

where:

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

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

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 $Var\left[\mathbb{E}[f(X)\|X_k]\right]/Var(f(X))$, 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:

$$
S_u = \frac{D_u(X_u)}{Var(f(X))}, \quad \sum_{u \subset \{1,\cdots,p\}} S_u = \frac{\sum_u D_u(X_u)}{Var(f(X))} = \frac{Var(f(X))}{Var(f(X))} = 1 \tag{2}
$$

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:

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

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.

2.3 Markov Chain Monte Carlo - MCMC

The ***Monte Carlo*** method [12, 13] 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 that one can 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 in the long run all the samples come from the desired distribution. One of the main issues with using a Monte Carlo method is that is a memoryless process, which can lead to a very long and time consuming process to draw samples from a complex distribution. In other words, if one sample comes from a high density 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 probabilities of the future depend only on the present [14, 15]. 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 values with higher probability. The main idea behind Metropolis-Hastings is that by running this algorithm for a good amount 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 important fact 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.

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 is the current value and $q(x\|x_i)$ be a proposal distribution. This proposal distribution can be symmetric or asymmetric in this method. Define 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**

2.4 Jensen Shannon Divergence

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

where

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

3 Supplementary Results

3.1 Synthetic target methodology

As we are dealing with 5 parameters in our mechanistic model, we design different target data scenarios with 5 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 5 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 5 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 5 different cases in this scenario, since there are 5 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 Eqn. (3). 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.

$$\binom{n}{k} \times 2^{k-1}, \quad k = 1, \cdots, 5 \tag{3}$$

3.2 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 Figs. S1-S6 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).

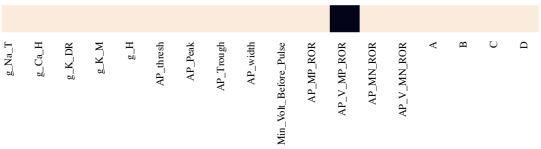

Fig. S1 5 choose 0 - KS tests for cGAN samples versus target data samples. Peach color indicates failure to reject the null hypothesis that the cGAN samples and synthetic target data samples are from the same distribution.

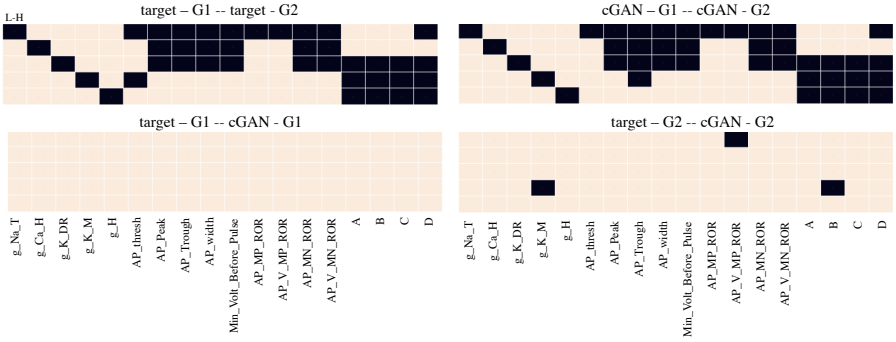

Fig. S2 5 choose 1 - KS tests. ***Top left*** - KS tests on two groups of targets (i.e. G1 (G2) with low - **L** (high - **H**) values of the parameter). ***Top right*** - KS test on cGAN samples corresponding to the two groups of target data (i.e. cGAN-G1 (cGAN-G2) with low - **L** (high - **H**) values of the parameter). ***Bottom left*** - KS test of cGAN-G1 versus target-G1. ***Bottom right*** - KS test of cGAN-G2 versus target-G2.

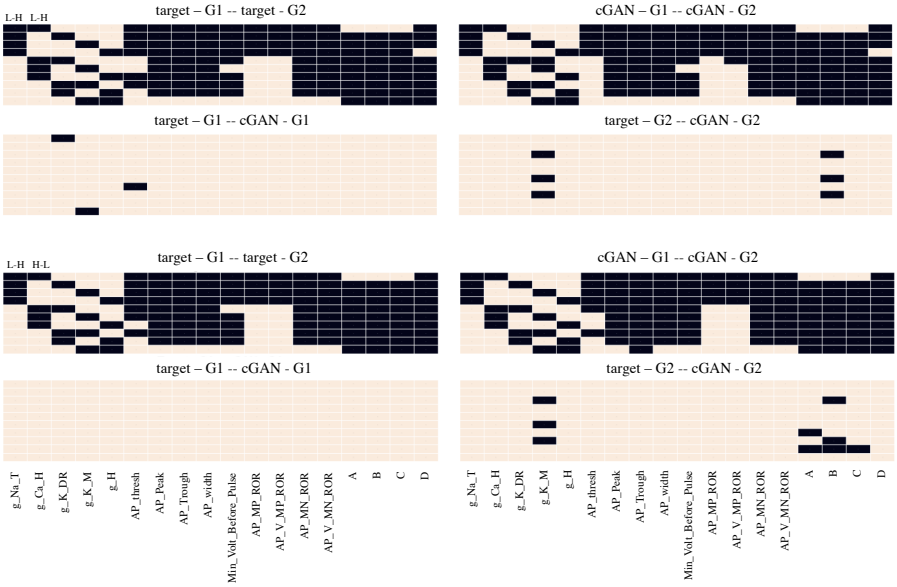

Fig. S3 5 choose 2 - KS tests. Panels are arranged in a similar fasion as Fig. S2. Top: L-H, L-H. Bottom: L-H, H-L.

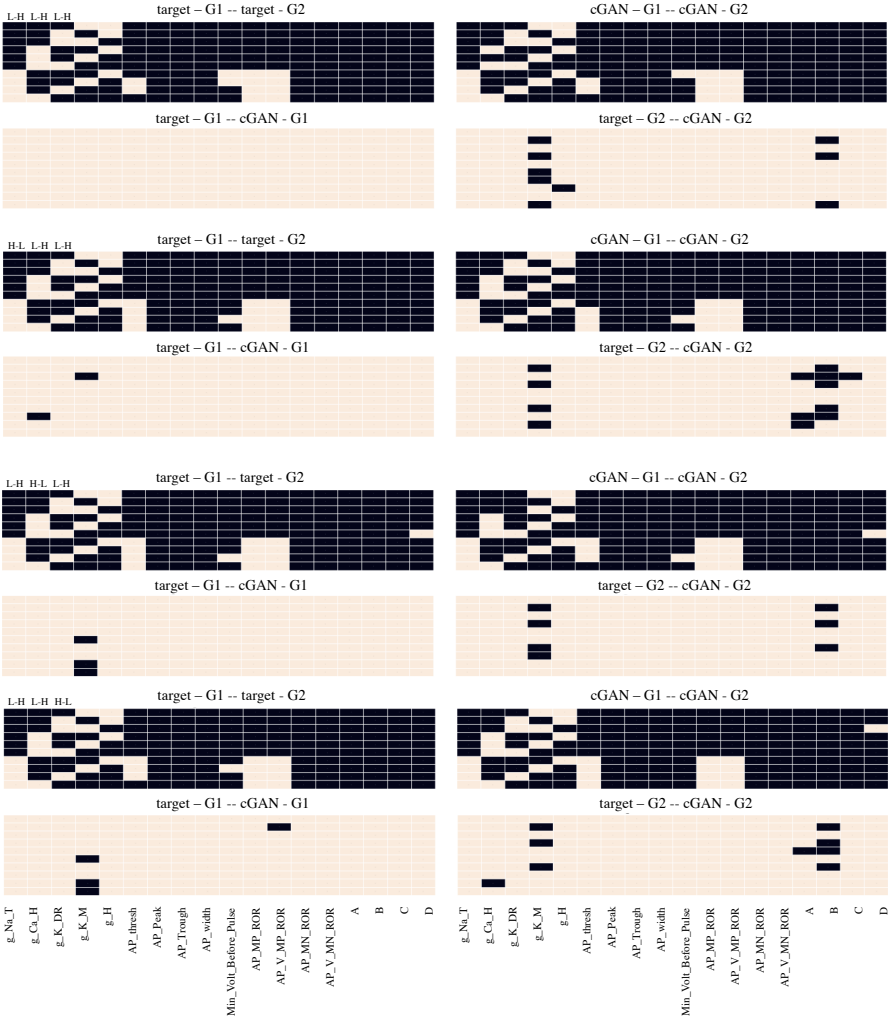

Fig. S4 5 choose 3 - KS tests. Panels are arranged in a similar fashion as Fig. S2. From top to bottom: *(1)* ***L-H L-H L-H***, *(2)* ***H-L L-H L-H***, *(3)* ***L-H H-L L-H***, *(4)* ***L-H L-H H-L***.

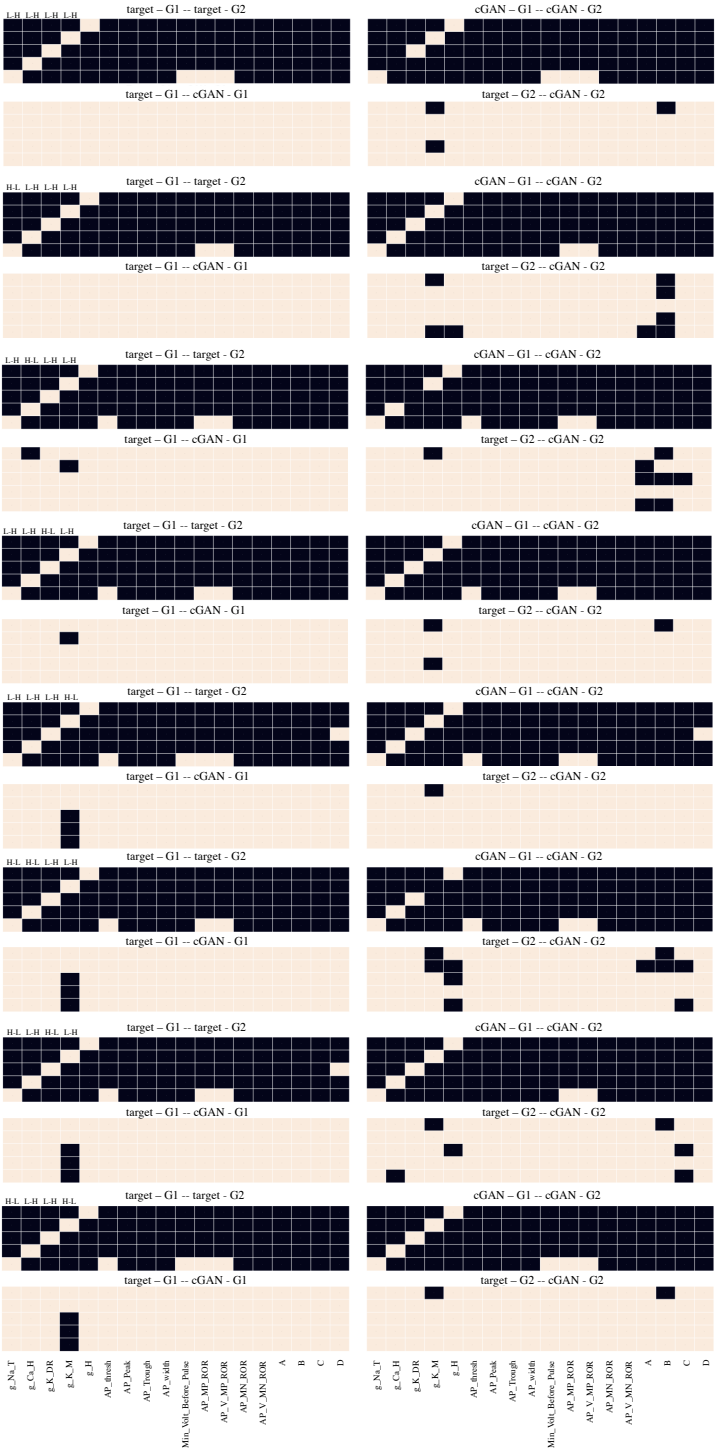

Fig. S5 5 choose 4 - KS tests. Panels are arranged in a similar fashion as Fig. S2. From top to bottom: ***(1) L-H L-H L-H L-H***, ***(2) H-L L-H L-H L-H***, ***(3) L-H H-L L-H L-H***, ***(4) L-H L-H H-L L-H***, ***(5) L-H L-H L-H H-L***, ***(6) H-L H-L L-H L-H***, ***(7) H-L L-H H-L L-H***, ***(8) H-L L-H L-H H-L***.

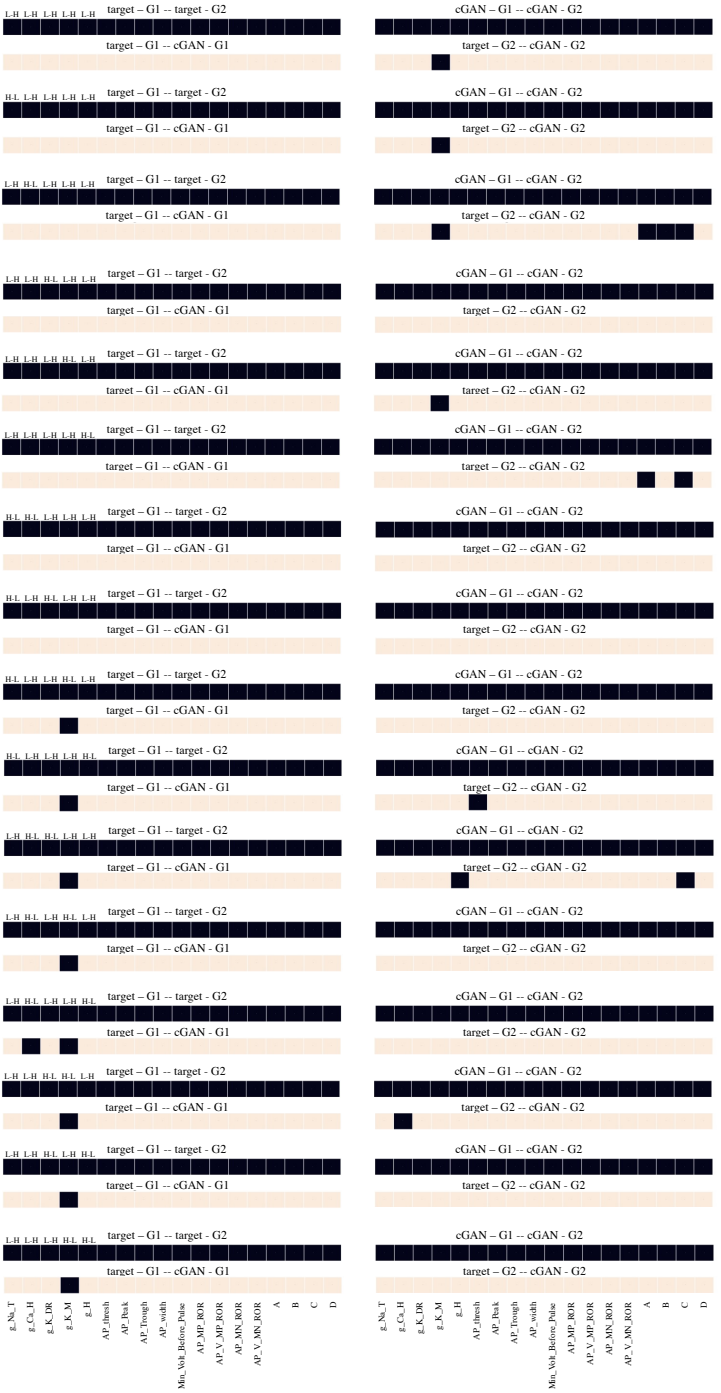

Fig. S6 5 choose 5 - KS tests. - Panels are arranged in a similar fashion as Fig. S2. From top to bottom: ***(1) L-H L-H L-H L-H L-H***, ***(2) H-L L-H L-H L-H L-H***, ***(3) L-H H-L L-H L-H L-H***, ***(4) L-H L-H H-L L-H L-H***, ***(5) L-H L-H L-H H-L L-H***, ***(6) L-H L-H L-H H-L L-H H-L***, ***7) 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***.

5-choose-1

True	2/90
0/90	3/90

Total = 5/270

5-choose-2

True	7/360
3/360	15/360

Total = 25/1080

5-choose-3

True	9/720
9/720	38/720

Total = 56/2160

5-choose-4

True	7/720
15/720	41/720

Total = 63/2160

5-choose-5

True	0/288
9/288	13/288

Total = 22/864

Fig. S7 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.

3.3 Output of cGAN samples in feature space for synthetic target data

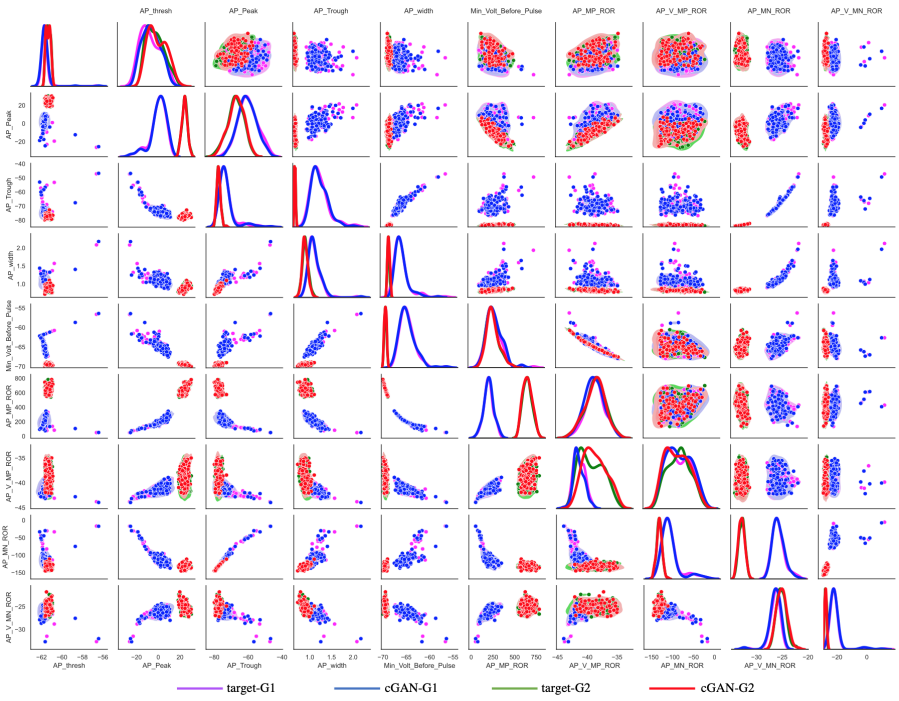

Fig. S8 Performance of cGAN on synthetic targets from 2 groups with distinct parameter structures - AP features. KDE plots (main diagonals) and scatter plots (lower and upper triangles) for Group 1 (G1) target data (magenta), Group 2 (G2) target data (green), cGAN samples for G1 (blue) and cGAN samples for G2 (red). *Lower main diagonal and lower triangle* - only 1 parameter (g_{NaT}) is distributed differently in the G1 target data than in the G2 target data, and the other 4 parameters have the same distribution in the G1 and G2 target data. We refer to this scenario as "5 choose 1" in the Supplementary Methods. *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 the Supplementary Methods.

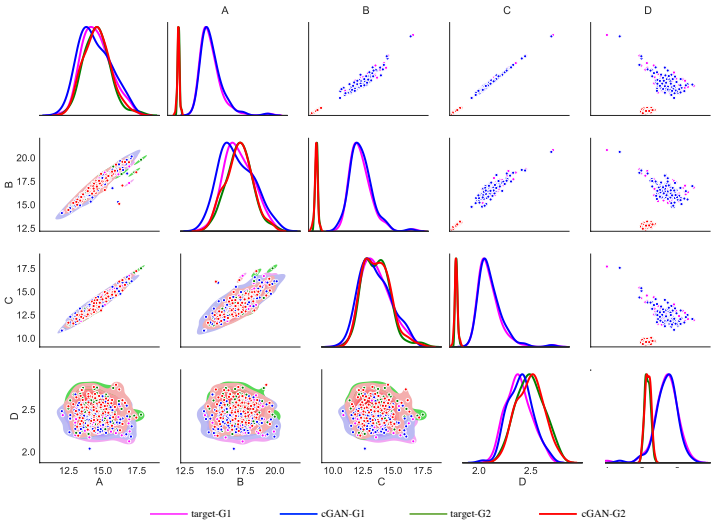

Fig. S9 Performance of cGAN on synthetic targets from 2 groups with distinct parameter structures - HP features. KDE plots (main diagonals) and scatter plots (lower and upper triangles) for Group 1 (G1) target data (magenta), Group 2 (G2) target data (green), cGAN samples for G1 (blue) and cGAN samples for G2 (red). *Lower main diagonal and lower triangle* - only 1 parameter (g_{NaT}) is distributed differently in the G1 target data than in the G2 target data, and the other 4 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.

3.4 Output of cGAN samples in feature space for experimental target data

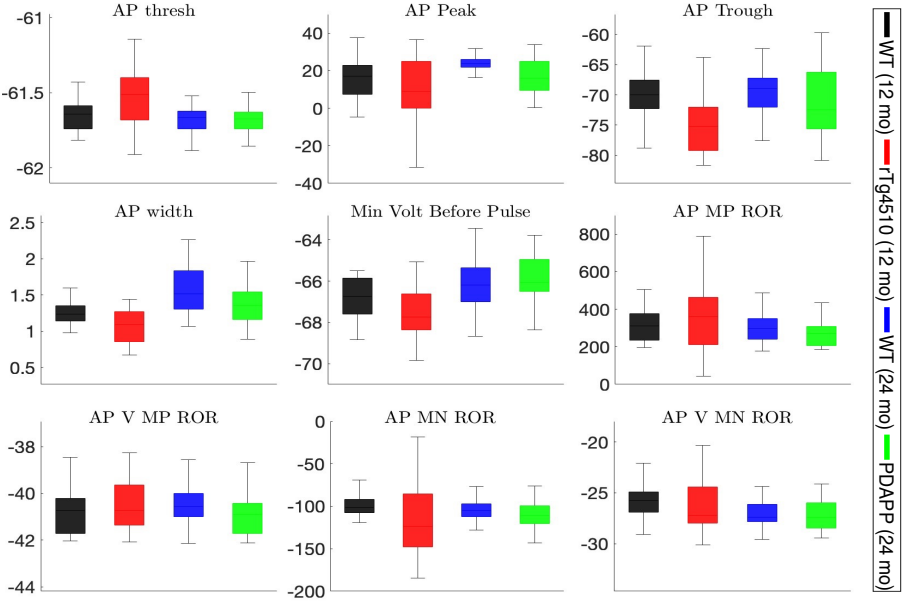

Fig. S10 Box and whisker plots of the action potential (AP) features extracted from the mechanistic model voltage traces obtained by pushing forward the cGAN parameter samples with experimental target data.

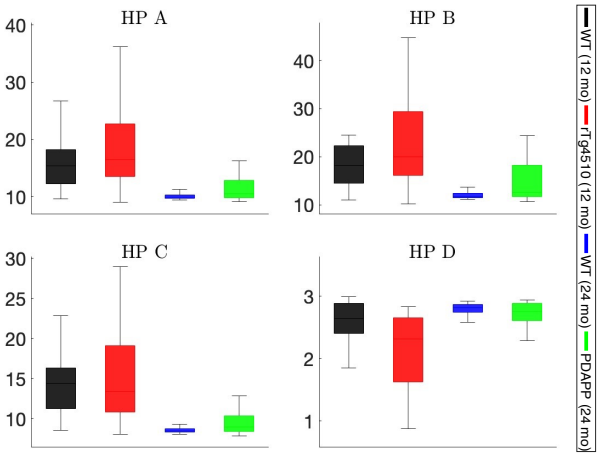

Fig. S11 Box and whisker plots of the membrane hyperpolarization (HP) features extracted from the mechanistic model voltage traces obtained by pushing forward the cGAN parameter samples with experimental target data.

References

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

[2] Booth, C.A., Witton, J., Nowacki, J., Tsaneva-Atanasova, K., Jones, M.W., Randall, A.D., Brown, J.T.: 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)

[3] Price, K., Storn, R.M., Lampinen, J.A.: Differential Evolution: a Practical Approach to Global Optimization. Springer, 53851, Lappeenranta (2006)

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

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

[6] Tosin, M., Côrtes, A., Cunha, A.: A Tutorial on Sobol' Global Sensitivity Analysis Applied to Biological Models. Networks in Systems Biology, 93–118 (2020)

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

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

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

[10] Jaxa-Rozen, M., Kwakkel, J.: 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)

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

[12] Hoff, P.D.: A First Course in Bayesian Statistical Methods vol. 580. Springer, NY10013, USA (2009)

[13] Gelman, A., Carlin, J.B., Stern, H.S., Rubin, D.B.: Bayesian Data Analysis. Chapman and Hall/CRC, 6000 Broken Sound Parkway NW, Suite 300 (1995)

[14] Brooks, S., Gelman, A., Jones, G., Meng, X.-L.: Handbook of Markov Chain Monte Carlo. CRC Press, 6000 Broken Sound Parkway NW, Suite 300 (2011)

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