

Fast reconstruction of degenerate populations of conductance-based neuron models from spike times

Julien Brandoit¹, Damien Ernst^{1,2}, Guillaume Drion^{1❸}, Arthur Fyon^{1❸*}

1 Montefiore Institute, University of Liège, 10 allée de la découverte, Liège, 4000, Belgium

2 LTCI, Telecom Paris, Institut Polytechnique de Paris, 19 place Marguerite Perey, Palaiseau, 91120, France

❸These authors contributed equally to this work.

* afyon@uliege.be

Abstract

Inferring the biophysical parameters of conductance-based models (CBMs) from experimentally accessible recordings remains a central challenge in computational neuroscience. Spike times are the most widely available data, yet they reveal little about which combinations of ion channel conductances generate the observed activity. This inverse problem is further complicated by neuronal degeneracy, where multiple distinct conductance sets yield similar spiking patterns. We introduce a method that addresses this challenge by combining deep learning with Dynamic Input Conductances (DICs), a theoretical framework that reduces complex CBMs to three interpretable feedback components governing excitability and firing patterns. Our approach first maps spike times directly to DIC values at threshold using a lightweight neural network that learns a low-dimensional representation of neuronal activity. The predicted DIC values are then used to generate degenerate CBM populations via an iterative compensation algorithm, ensuring compatibility with the intermediate target DICs, and thereby reproducing the corresponding firing patterns, even in high-dimensional models. Applied to two neuronal models, this algorithmic pipeline reconstructs spiking, bursting, and irregular regimes with high accuracy and robustness to variability, including spike trains

generated by Poisson processes. It produces diverse degenerate populations within milliseconds on standard hardware, enabling scalable and efficient inference from spike recordings alone. Beyond methodological advances, we provide an open-source software package with a graphical interface that allows experimentalists to generate and explore CBM populations directly from spike trains without requiring programming expertise. Together, this work positions DICs as a practical and interpretable link between experimentally observed activity and mechanistic models. By enabling fast and scalable reconstruction of degenerate populations directly from spike times, our approach provides a powerful way to investigate how neurons exploit conductance variability to achieve reliable computation and provides the foundation for experimental applications that span from neuromodulation studies to real-time model-guided interventions.

Author summary

Neurons communicate through spikes, and spike timing is a crucial part of neuronal processing. Spike times can be recorded experimentally both intracellularly and extracellularly, and are the main output of state-of-the-art neural probes. On the other hand, neuronal activity is controlled at the molecular level by the currents generated by many different transmembrane proteins called ion channels. Connecting spike timing to ion channel composition remains an arduous task to date. To address this challenge, we developed a method that combines deep learning with a theoretical tool called Dynamic Input Conductances (DICs), which reduce the complexity of ion channel interactions into three interpretable components describing how neurons spike. Our approach uses deep learning to infer DICs directly from spike times and then generates populations of “twin” neuron models that replicate the observed activity while capturing natural variability in membrane channel composition. The method is fast, accurate, and works using only spike recordings. We also provide open-source software with a graphical interface, making it accessible to researchers without programming expertise.

Introduction

A central objective in both experimental and computational neuroscience is to understand how neurons generate and regulate electrical activity from their ion channels, linking microscopic mechanisms to macroscopic dynamics. Extracellular recordings, particularly spike times, remain the most widely accessible data, especially when probing large networks [1–3]. Spike times provide rich information about neural dynamics across brain regions and conditions. However, while they reveal what neurons do and how they respond to external perturbations, they rarely reveal the underlying mechanisms, specifically which combinations of biophysical parameters produce the observed activity. This question is crucial because such biophysical parameters are primary targets of neuromodulators and neuroactive medications [4–6].

A major obstacle in linking neuronal activity with biophysical properties is *neuronal degeneracy*, the well-established fact that multiple distinct parameter sets, particularly effective ion channel densities, can produce similar spiking behaviors [7]. This functional flexibility enhances robustness but complicates interpretation, as neurons exhibit inherently nonlinear, high-dimensional dynamics. When different parameter combinations produce indistinguishable activity, identifying which configurations underlie recorded behavior becomes an underdetermined problem [8–10].

Conductance-based models (CBMs) provide a principled framework to tackle this challenge computationally. Their parameters, such as maximal ion conductances, link directly to measurable biological properties like ion channel expression. However, realistic CBMs are complex, often requiring dozens of parameters [9, 11, 12]. Combined with degeneracy, this complexity makes inferring conductances from experimental data highly challenging and often computationally intractable.

This difficulty defines a core inverse problem in computational neuroscience: *given an observed activity pattern such as a spike train, how can one identify a set – or better, the population – of biophysical models that could have generated it?* Solving this problem is essential for bridging experimental observations and mechanistic understanding, yet existing approaches face key limitations. Many require full voltage traces or intracellular recordings, making them sensitive to noise and less feasible in practice. Others are computationally expensive, lack robustness to biological variability,

require enormous amounts of data, or fail to account for degeneracy altogether [13–22].

A promising direction comes from dynamical systems theory, and particularly the concept of Dynamic Input Conductances (DICs), three voltage-dependent curves that capture the aggregated influence of all ion channels, separated by timescale: fast (spike upstroke), slow (spike downstroke and interspike interval), and ultra-slow (interburst interval) [23, 24]. Each DIC reflects a voltage-dependent feedback gain, with its sign indicating whether the contribution is positive or negative. Fast localized positive feedback combined with global slow negative feedback supports excitability and spiking, whereas slow localized positive feedback and global ultra-slow negative feedback promote bursting. DICs thus provide an interpretable, low-dimensional representation of neuronal excitability and firing patterns.

Importantly, any CBM, regardless of complexity, can be approximated by three DIC curves, which are sufficient to characterize even complex bursting dynamics. Moreover, the values of these curves at the threshold voltage are especially predictive of the resulting firing pattern. This enables a further reduction: a CBM can be summarized by just three scalar DIC values evaluated at threshold. Building on this idea, recent work has shown that degenerate CBM populations can be generated from such scalar DICs [24], but two limitations remain. First, no automated quantitative mapping exists from observed spike patterns to DIC values, leaving the process dependent on expert intuition. Second, existing DIC-to-CBM generation methods do not reliably enforce threshold DIC constraints in complex models, leading to residual errors characterized by outlying neuronal activities within the population.

In this work, we address both limitations. We introduce a lightweight deep learning architecture that maps spike times directly to DIC values at threshold, providing a data-driven solution to the first gap. In parallel, we propose an iterative compensation algorithm that improves the DIC-to-CBM generation block, yielding more accurate enforcement of target DICs even in high-dimensional models. Together, these two components form a complete pipeline: given only spike times, the method outputs a population of degenerate CBMs within milliseconds on standard hardware. We validate the pipeline on two distinct models and show that it faithfully reconstructs neuronal activity across spiking, bursting, and irregular regimes, while maintaining robustness to variability and noise.

By combining deep learning with DIC theory, this approach provides a practical solution to the inverse inference problem. It bridges experimentally accessible observables and mechanistic CBMs, demonstrates that DICs serve as interpretable low-dimensional intermediates, and enables scalable reconstruction of degenerate populations from spike times alone.

To support widespread adoption and facilitate daily use in experimental settings, we provide our entire pipeline as an open-source software package with a graphical interface [25]. Recognizing that many researchers in the biomedical sciences come from diverse backgrounds where programming and numerical modeling are not core skills, the tool is designed to be intuitive and easy to use, allowing experimentalists to generate and explore model populations directly from spike recordings without writing code.

Results

General problem statement

In this work, we assumed a known conductance-based model (CBM) whose membrane dynamics is described by:

$$C \frac{dV}{dt} + g_{\text{leak}}(V - E_{\text{leak}}) = - \sum_{i \in \mathcal{I}} \bar{g}_i m_i^{p_i} h_i^{q_i} (V - E_i) + I_{\text{ext}} \quad ,$$

where V denotes the membrane potential, C is the membrane capacitance, g_{leak} and E_{leak} are the leak conductance and reversal potential, and each ionic current $i \in \mathcal{I}$ is characterized by its maximal conductance \bar{g}_i , gating variables m_i and h_i with powers p_i and q_i , and reversal potential E_i . The external current input is denoted by I_{ext} . In this formulation, the structure of the model (i.e., the functional form of the ionic currents and their gating dynamics) was assumed to be fixed and known (see Materials and Methods). The unknown parameters are the conductances $\bar{g} = [\bar{g}_1, \dots, \bar{g}_{|\mathcal{I}|}, g_{\text{leak}}] \in \mathcal{G}$, which must be inferred from observations extracted from the voltage trace. A specific choice of \bar{g} fully specifies one instance of the considered CBM, while all other parameters (e.g., reversal potentials, capacitance, ...) are assumed to be known and fixed. We chose spike times as representation of the activity, as these data are easily accessible from both intracellular and extracellular recordings. We denote such a

recorded spike times sequence by:

$$x = [t_1, t_2, \dots, t_{N_{\text{spikes}}}] \quad .$$

The quantity N_{spikes} denotes the total number of spikes detected in the recording. Its value depends both on the duration of the recording and on the firing activity of the neuron, which makes x a variable-length representation of the neuronal activity.

Due to degeneracy [4, 26], the mapping:

$$x \longmapsto \bar{g} \in \mathcal{G}^*(x) \quad ,$$

is not bijective: the solution to the inference problem is not a single point, but rather a subspace $\mathcal{G}^*(x) \subset \mathcal{G}$ containing infinitely many parameter sets compatible with x [23, 24, 27].

Our objective is to build a set of P models (with P freely chosen by the experimentalist) with different conductance values that all reproduce a firing pattern similar to x . We call this set \mathcal{P} , and since each model is defined by its value of \bar{g} we write $\mathcal{P} = \{[\bar{g}_1, \dots, \bar{g}_{|\mathcal{I}|}, g_{\text{leak}}]_i\}_{i=1}^P$. We can build such a set by generating P instances from the subspace $\mathcal{G}^*(x)$, such that we can infer a whole degenerate population. Traditional approaches do not take degeneracy into account, and therefore are only able to generate a single point from $\mathcal{G}^*(x)$ as a solution. Moreover, they often require the full voltage trace, which is harder to obtain than spike times. Others try to directly learn the mapping:

$$x \longmapsto \mathcal{G}^*(x) \quad ,$$

as a whole (for example, through the support of a learned joint distribution) leading to methods that are slow, data-intensive and hard to train, especially because \mathcal{G} is of high-dimensionality.

In this work, to overcome the limitations of existing methods, we adopt an intermediate strategy that neither infers a single solution nor attempts to directly learn the full high-dimensional solution space. Our approach divides the problem into two parts and leverages intermediate low-dimensional representations of CBMs called dynamic input conductances (DICs) (see Materials and Methods) [23, 24, 28]. For any

CBM, we can analytically construct a DIC representation determined by:

$$g_{\text{DICs}}(V) = S(V; \bar{g}) \cdot \bar{g} \quad , \quad (1)$$

where S is the sensitivity matrix determined by the CBM, and $g_{\text{DICs}}(V) : V \mapsto \mathbb{R}^3$ provides a low-dimensional equivalent of the high-dimensional conductance vector \bar{g} . Importantly, only the DIC values evaluated at a specific potential, the threshold potential V_{th} (see Materials and Methods), are sufficient to capture most of the spontaneous activity associated with a given CBM [23, 24]. The three components of the DIC representation correspond to distinct timescales of the membrane dynamics (see Materials and Methods), giving rise to three separate DIC curves: the fast component $g_f(V)$, the slow component $g_s(V)$, and the ultra-slow component $g_u(V)$. Each curve captures the influence of ionic conductances acting on its respective timescale. This reduces the dimensionality of the degenerate solution space $\mathcal{G}^*(x)$ to just three scalars, providing a compact summary of all conductance configurations that produce similar firing activity [24].

Leveraging this intermediate representation, our approach proceeds in two steps (Fig 1). First, we learn a mapping from observed spike times x to the corresponding DIC values:

$$x \mapsto g_{\text{DICs}}(V_{\text{th}}) \quad ,$$

enabling parameter inference in a tractable, low-dimensional space by inferring the left-hand side, $g_{\text{DICs}}(V_{\text{th}})$, of relation (1). Second, given a predicted DIC vector, we generate valid CBM conductance vectors \bar{g} that are compatible with these DIC values:

$$g_{\text{DICs}}(V_{\text{th}}) \mapsto \bar{g} \in \mathcal{G}^* \quad ,$$

thus recovering diverse, biologically plausible instances of the original high-dimensional model and generating solutions of relation (1).

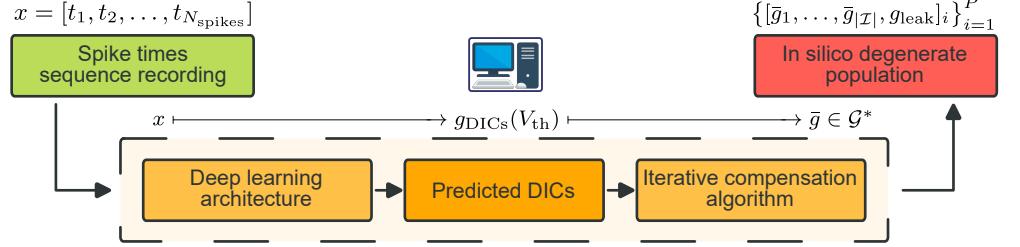


Fig 1. Our proposed approach. Spike time sequences are processed by a deep learning model that predicts dynamic input conductances (DICs), a compact representation of the high-dimensional conductance space. These predicted DIC values serve as targets for an iterative compensation algorithm, which explores the degenerate solution space to generate multiple conductance configurations \bar{g} that reproduce the input spike pattern. This two-step strategy (i) reduces the inference problem to predicting only three scalar DIC values, and (ii) leverages compensation to recover diverse biologically plausible parameter sets consistent with the observed activity.

To tackle the first part, we use a *deep learning architecture* to approximate the mapping from spike times to DICs. For the second part, we propose to improve an existing compensation method [24] that carries out the mapping from DICs to conductance values. We call this improved method the *iterative compensation algorithm*. Overall, our approach reads:

$$x \xrightarrow{\text{Deep learning architecture}} g_{\text{DICs}}(V_{\text{th}}) \xrightarrow{\text{Iterative compensation algorithm}} \bar{g} \in \mathcal{G}^* .$$

This requires to (i) have a reliable method to augment back from DICs to \bar{g} ; (ii) build a synthetic dataset to train the deep learning architecture; (iii) create an architecture that is able to process input of varying length; (iv) ensure that this methodology is robust to noise and can be generalized to many CBMs.

From DIC values to degenerate neuronal populations

The generation of degenerate populations of CBMs that satisfy target DIC values focuses on identifying output conductance vectors \bar{g} that are compatible with an input set of constraints. These constraints specify desired values along the fast, slow, and ultra-slow DIC components, $g_f(V)$, $g_s(V)$, and $g_u(V)$, at particular voltages along the DIC curves. In this study, we concentrated on constraints at the threshold voltage, $g_{\text{DICs}}(V_{\text{th}}) = [g_f(V_{\text{th}}); g_s(V_{\text{th}}); g_u(V_{\text{th}})]$, but in principle, DIC constraints can be applied at any voltage and for any target values along any DIC curve [23]. The

relationship between DICs and the underlying conductances is given by Eq (1), where $S(V; \bar{g})$ is analytically determined by the CBM equations and is therefore known. Our goal is to find different values of \bar{g} such that the relation (1) is satisfied for imposed values of $g_{\text{DICs}}(V_{\text{th}})$ as input, acknowledging that multiple solutions may exist due to the degeneracy of the system, making generation an underdetermined problem.

In the following, we first present the method introduced by [24], which addressed this problem using a linear approximation. We then discuss the main limitations and shortcomings of this approach. Building on this foundation, we introduce an improved algorithm, which we call **the iterative compensation algorithm**. The remainder of the section compares both methods, highlighting how the iterative compensation algorithm enables the generation of degenerate populations more reliably and accurately.

The method of [24] addressed the generation of degenerate CBM populations from DIC values using *a single linear compensation step*. In that approach, the full conductance vector was split into two subsets, $\bar{g} = [\bar{g}_{\text{comp}}; \bar{g}_{\text{random}}]$. The random subset values, \bar{g}_{random} , were sampled from distributions slightly broader than those observed experimentally [7], while the remaining compensable subset, $\bar{g}_{\text{comp.}}$, was adjusted to satisfy the DIC constraints. Sampling \bar{g}_{random} allows the compensation problem to become fully determined, generating different valid solutions depending on the random value of \bar{g}_{random} and thus producing a degenerate population of CBMs.

To determine the components $\bar{g}_{\text{comp.}}$ that made \bar{g} compatible with the DIC constraints, the following linear equation was solved:

$$g_{\text{DICs}}(V)|_{V=V_{\text{th}}} = [g_f(V); g_s(V); g_u(V)]|_{V=V_{\text{th}}} = S(V)|_{V=V_{\text{th}}} \cdot \bar{g} \quad , \quad (2)$$

where $S(V)$ is the sensitivity matrix describing how each conductance contributes to the fast, slow, and ultra-slow DIC components (see Materials and Methods). It also defines the *compensatory structure* of Eq (2), which can be linear, as in [24], or nonlinear, depending on its functional form.

This linear method is exact when the compensatory structure is linear, i.e., when $S(V)$ does not depend on $\bar{g}_{\text{comp.}}$. However, it fails for models in which $S = S(V; \bar{g}_{\text{comp.}})$, i.e. when S depends on the compensated conductances themselves such that the compensatory structure is nonlinear. For example, the STG neuron model explicitly

includes intracellular calcium dynamics that depend on calcium conductances, thereby introducing nonlinearities:

$$g_{\text{DICs}}(V)|_{V=V_{\text{th}}} = S(V; \bar{g})|_{V=V_{\text{th}}} \cdot \bar{g} \quad . \quad (3)$$

In [24], this nonlinear system was handled by fixing values of $\hat{g}_{\text{comp.}}$ a priori, reducing the problem to a linear one with $S(V; \hat{g}_{\text{comp.}}) \approx \hat{S}(V)$. This approximation produced populations whose actual DIC values deviate from the targets, leading to residual errors, in the sense that an element \bar{g} generated based on this approximation will lead to the following inequality:

$$g_{\text{DICs}}(V)|_{V=V_{\text{th}}} \neq S(V; \bar{g})|_{V=V_{\text{th}}} \cdot \bar{g} \quad , \quad (4)$$

and this is because $\hat{S}(V_{\text{th}}; [\bar{g}_{\text{random}}; \hat{g}_{\text{comp.}}]) \neq S(V_{\text{th}}; \bar{g})$. To quantify this mismatch, we define the residual (Euclidean) norm at threshold as:

$$\begin{aligned} \|r\|_2 &= \|\hat{S}(V; [\bar{g}_{\text{random}}; \hat{g}_{\text{comp.}}])|_{V=V_{\text{th}}} \cdot \bar{g} - S(V; \bar{g})|_{V=V_{\text{th}}} \cdot \bar{g}\|_2, \\ &= \left\| \underbrace{g_{\text{DICs}}(V)|_{V=V_{\text{th}}}}_{\text{Target DICs}} - \underbrace{S(V; \bar{g})|_{V=V_{\text{th}}} \cdot \bar{g}}_{\text{Actually enforced DICs}} \right\|_2, \end{aligned} \quad (5)$$

that is the norm of the difference between the target DICs and the actually enforced DICs.

As a result of the inequality (4), the DIC constraints $g_{\text{DICs}}(V_{\text{th}})$ were not strictly enforced across the population, leading to neuron populations that are neither truly degenerate nor consistent with the target constraints, and resulting in inconsistent firing patterns or significant outliers.

To overcome these limitations, we introduced the iterative compensation algorithm, which generalizes the linear scheme of [24] by applying multiple compensation steps *iteratively*. Starting from an initial guess, the compensable conductances are updated step by step, recomputing the sensitivity matrix approximation at each iteration to better match the DIC constraints (see Materials and Methods for details). The process can be repeated until the residual norm between target and actual DIC values is sufficiently small.

Fig 2A compares the mean residuals norm obtained with the default linear method (hatched) and our iterative compensation algorithm (plain) over up to 10 iterations, tested on 1,633 target DICs ($P = 250$ instances each). For each target DICs, we report the mean residual norm computed over the P generated solutions. After only five iterations, the iterative method reduced residuals by a factor of 15, striking a balance between accuracy and computational cost; accordingly, we used five iterations as the default in the remainder of this work.

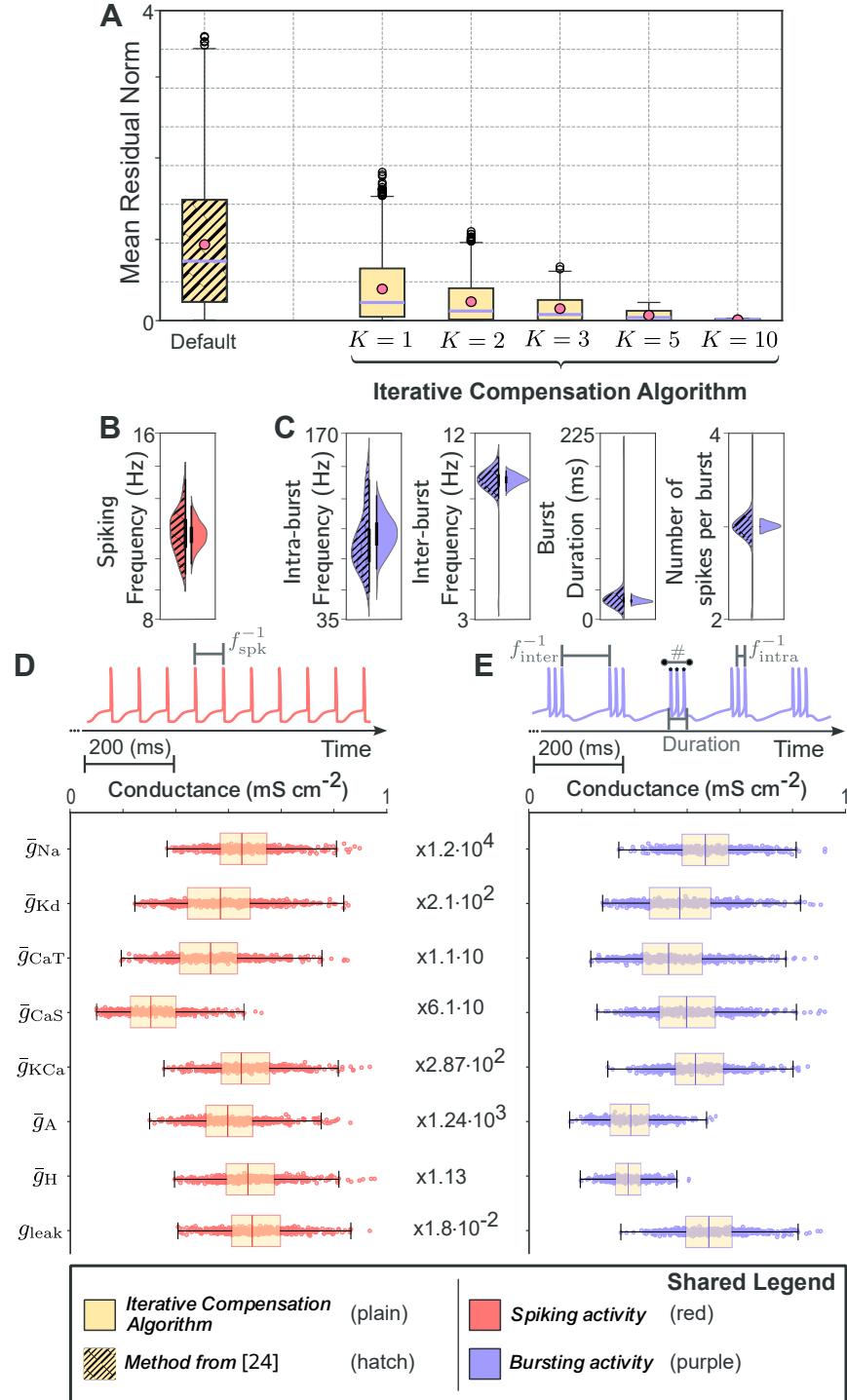


Fig 2. Iterative compensation improves constraint satisfaction and preserves degeneracy in CBMs with nonlinear dynamics. (A) Mean residuals of DIC constraints across 1,633 targets and 250 instances per population, comparing linear compensation (hatched, 0 iterations from [24]) to iterative compensation (plain), showing a rapid reduction in residuals. (B) Distribution of mean firing frequency f_{spk} in spiking neurons (red), demonstrating tighter and more consistent activity with iterative compensation (plain) compared to linear (hatched). (C) Distributions of bursting features (intra-burst frequency f_{intra} , inter-burst frequency f_{inter} , burst duration, spikes per burst #) in bursting neurons (purple), showing more compact activity profiles with iterative compensation. (D, E) Maximal conductance values across spiking (D) and bursting (E) neuron populations generated with iterative compensation, revealing high variability despite similar firing patterns, illustrating preserved degeneracy.

Since DICs serve as an intermediate representation rather than the final target, a more direct measure of accuracy is given by the firing pattern distributions of the generated populations. Fig 2B–C compare populations generated with the default (hatched) and iterative (plain) methods for spiking neurons (red, Fig 2B) and bursting neurons (purple, Fig 2C). The iterative algorithm reduces outliers and produces more compact, centered distributions of activity statistics. Spiking activity is summarized by mean firing frequency, while bursting activity is described by intra-burst frequency, inter-burst frequency, burst duration, and spikes per burst (illustrated in Fig 2D,E). These tighter distributions confirm that the iterative approach enhances the reliability of the DIC-based generative method.

As its original linearized version, the iterative compensation algorithm produces degenerate populations. Sampling of \bar{g}_{random} introduces variability across instances, resulting in distinct compensation problems even for the same target DIC. This ensures that maximal conductance vectors remain heterogeneous, often differing by several folds, while still producing consistent activity statistics. This is illustrated in both spiking (Fig 2D) and bursting (Fig 2E) populations.

This improved compensation step forms the basis for generating the large synthetic dataset used to train our deep learning architecture, described next; but is also an integral part of the complete algorithmic pipeline at inference (Fig 1).

Building a synthetic dataset that links DIC values to a diverse range of neuronal activities

For the deep learning architecture to predict DIC values from spike times, a suitable dataset was required. None exists in the literature, as DICs have only recently emerged as a tool to reduce the dimensionality of neuronal activity. We therefore constructed a large open source synthetic dataset spanning a broad range of DIC values and corresponding activity patterns (i.e., spike times) [29]. Instead of sampling conductance parameters directly, we uniformly sampled the slow and ultra-slow DICs $g^* = (g_s(V_{\text{th}}); g_u(V_{\text{th}}))$ (Fig 3A; only part of the dataset is shown for clarity), since these components primarily shape firing activity. The fast DIC $g_f(V_{\text{th}})$ was constrained qualitatively to ensure spontaneous activity, i.e. sufficiently negative (see Materials and

Methods). Sampling was performed within bounded regions extending beyond ranges derived from broad biological priors (see Supporting Information, S1 Appendix). This approach ensures that the architecture learns not only typical biological activities but also edge cases relevant during inference.

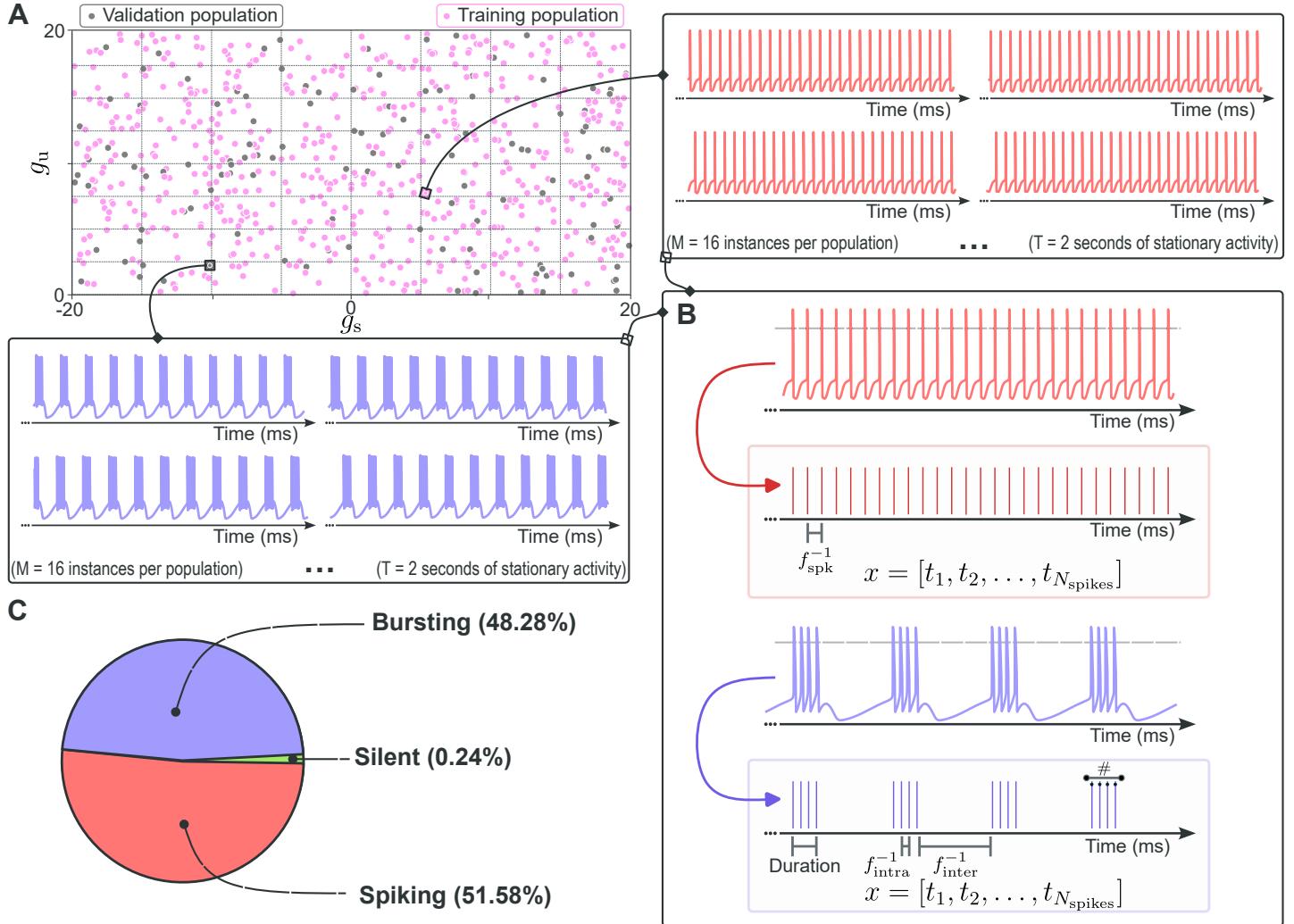


Fig 3. The synthetic dataset generation process from samling in the DICs space. **(A)** Subset of the sampled DIC space used to generate degenerate CBM populations. Each dot corresponds to one population (16 instances), belonging to the training (pink) or validation (gray) set. Sampling is uniform and bounded to ensure broad coverage. **(B)** Schematic of the dataset generation pipeline: each population is simulated, spike times are extracted, and descriptors are computed. Each instance is then classified as spiking, bursting, or silent. **(C)** Class distribution across the full dataset: 51.58% spiking, 48.28% bursting, and 0.24% silent.

For each of the $N = 75,000$ sampled DIC couples, we generated a degenerate population of $M = 16$ CBM instances using the iterative compensation algorithm, yielding a total of $|\mathcal{T}| = 1,200,000$ simulated neurons. Each instance was simulated after

discarding transients, without any input current, and its spike times were extracted for further analysis (Fig 3B). We computed descriptive metrics to characterize activity: for spiking neurons, mean spiking frequency; for bursting neurons, intra-burst and inter-burst frequencies, burst duration, and number of spikes per burst. Firing class (spiking, bursting, or silent) was assigned based on interspike interval variability (see Materials and Methods). Silent instances (0.24%) were discarded due to their lack of spontaneous spikes and the absence of usable information for the pipeline. The final dataset consisted of 51.58% spiking and 48.28% bursting neurons (Fig 3C), providing a balanced and diverse foundation for training across firing regimes.

To prevent data leakage, instances from the same population were kept together and never split across training and validation sets. We allocated $|\mathcal{T}_{\text{train}}| = 1,000,000$ instances for training and $|\mathcal{T}_{\text{val}}| = 200,000$ for validation. A separate test set was generated only after the architecture was fully trained and fixed. This strict separation ensures that performance metrics reflect generalization to unseen DIC targets and conductance configurations.

Beyond training the architecture, this dataset also highlights how DICs represent neuronal behavior at the population level. Each activity descriptor can be viewed as a function of the sampled slow and ultra-slow DICs (Fig 4). A clear separation between spiking and bursting neurons emerges, with a threshold largely independent of g_u and located near $g_s \approx 0$. Moreover, smooth gradients appear for both bursting descriptors (Fig 4A) and the spiking one (Fig 4B), revealing a structured and nonlinear relationship between DIC targets and resulting activity. In particular, different neuronal activities are characterized by different DIC values, *yet multiple distinct DIC configurations can map to the same activity regime, especially in spiking*. This many-to-one correspondence captures the intrinsic degeneracy of neuronal systems and reflects the biological reality that diverse conductance profiles may underlie similar firing patterns. Although these descriptors are not used as inputs to the architecture, their smooth variation across DIC space supports the learnability of the inverse mapping from spike times to DICs. These patterns are particularly clear due to the absence of input current and noise, isolating the role of intrinsic conductances. This reinforces the role of DICs as low dimensional and interpretable intermediates for dimensionality reduction.

Consistent with prior work [23, 24], negative slow DIC values at threshold were

associated with bursting, while positive values lead to spiking. The overlapping zone between bursting and spiking (at the transition $g_s \approx 0$) is due to the heterogeneous populations that exist in this zone. This phenomenon is discussed in more detail in the Supporting Information (see S1 Appendix). The ranges of the colorbars confirm that the dataset spans a wide range of activity descriptors, ensuring diversity of neuronal dynamics.

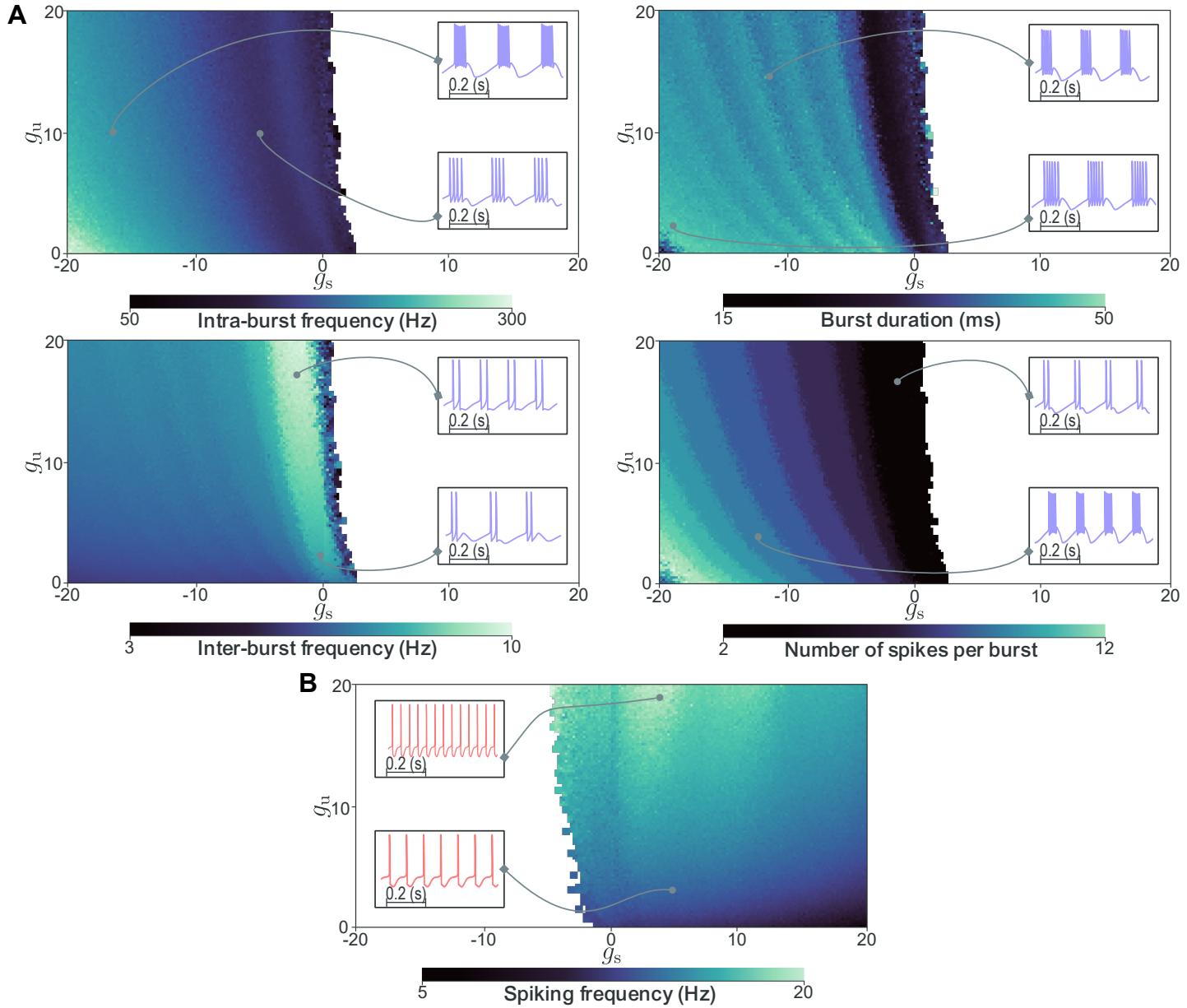


Fig 4. Activity descriptors vary smoothly across the DIC space. Heatmaps show how individual activity metrics vary across the DIC space, revealing clear gradients that reflect a nonlinear relationship between DIC constraints and neuronal firing patterns. **(A)** Bursting metrics including intra-burst frequency, inter-burst frequency, burst duration, and spikes per burst. **(B)** Spiking metrics summarized by mean firing rate. These patterns highlight the learnability of the inverse mapping from spike trains to DICs and support DICs as meaningful low-dimensional intermediates. Representative example traces illustrate diverse spiking and bursting activity within the dataset.

To our knowledge, this is the first dataset to systematically map descriptive activity metrics across the DIC space. This structured representation illustrates how conductances shape firing patterns, and how activity descriptors can be expressed in the DIC framework. *It may also help understand why neuromodulators often act on multiple*

channels with potentially opposing effects [4, 30]. Rather than modulating single conductances, they may shift the position of the neuron in DIC space, thereby altering its activity. Coordinated changes across channels could thus enable robust transitions between firing regimes by steering the effective conductances across timescales, highlighting the utility of the DIC framework for studying neuromodulation.

A deep learning model accurately predicts DIC targets from spike time sequences

Architecture and training overview

The core of the first block of our method (Fig 1) is a lightweight deep learning architecture that predicts DIC values directly from raw spike time sequences. Specifically, it estimates slow and ultra-slow DIC targets $\hat{g}^* \approx g^* = (g_s(V_{\text{th}}); g_u(V_{\text{th}}))$ without relying on summary statistics or hand-crafted features. The model follows an encoder–decoder design:

$$\underbrace{x \in \mathbb{R}^{* \times 1} \xrightarrow{\text{Encoder}} z_{\text{latent}} \in \mathbb{R}^{d_{\text{latent}}} \xrightarrow{\text{Decoder}} \hat{g}^* \in \mathbb{R}^2}_{\text{Projection into latent space} \quad \text{Mapping to DIC space}} ,$$

where the encoder converts variable-length spike sequences into a fixed-size latent vector, and the decoder maps this representation to the predicted DIC values. This structure allows the network to extract temporal features directly from spike times.

We deliberately chose to learn the encoding from data rather than rely on pre-defined summary statistics. Hand-crafted features would constrain the latent space to a fixed set of assumptions about relevance, whereas a learned encoder allows the model to discover task-relevant temporal patterns without prior biases. This representation was shaped directly by the loss during the training process, enabling the extraction of features optimized for DIC prediction. Although this design introduced additional parameters and increased training cost, these trade-offs were justified by improved flexibility and performance.

The final architecture comprises 115,627 trainable parameters, selected through a broad random hyperparameter search (see Supporting Information, S1 Appendix). While the model is highly flexible, it remains computationally efficient, supporting fast

training and inference on standard laboratory hardware, which makes it compatible with typical neurophysiology workflows.

Benchmarking on a standard laptop GPU shown that training completed within 24 hours. Inference is highly efficient, requiring approximately 1.25×10^{-5} seconds per forward pass on GPU and 6.24×10^{-5} seconds on CPU. This speed enables real-time or near real-time applications in experimental workflows without specialized hardware.

To guide representation learning and improve generalization, we introduced two auxiliary tasks operating on the latent space: (i) predicting descriptive neuronal activity metrics (spiking frequency, bursting properties), and (ii) classifying firing patterns as spiking or bursting. These auxiliary heads are softly integrated into the main regression objective via learnable weighting (see Materials and Methods), encouraging biologically meaningful representations without relying on hand-crafted features.

We further enhanced generalization using biologically inspired data augmentations that reflect typical experimental variability: random sequence cropping, spike deletion, and spike time jittering. Training was performed end to end with a composite objective comprising auxiliary task losses and heteroscedastic Huber loss, which incorporates uncertainty estimates predicted by a dedicated network head (see Materials and Methods). These uncertainty scores dynamically weighted the loss, providing interpretable confidence estimates for each predicted DIC value while reducing sensitivity to edge cases.

Prediction analysis and performance evaluations

We evaluated the backbone architecture, trained on the STG neuron dataset, using metrics spanning the primary DIC regression task as well as auxiliary regression and classification tasks. The model achieved high accuracy across these tasks, with 99.83% classification accuracy for firing classes and precise regression of neuronal activity features, which underscores its effectiveness in capturing complex neuronal dynamics.

The architecture accurately predicts key neuronal activity features, including mean spike frequency, intra- and inter-burst frequencies, burst duration, and number of spikes per burst. Mean absolute errors (MAE) for these auxiliary regressions are substantially lower than the intrinsic variability of the dataset (Table 1a), which confirms that the latent representation captures essential firing features.

Table 1. Performance summary of the backbone architecture on the STG neuron dataset. The table reports prediction accuracy for auxiliary regression tasks and the primary DICs regression task.

(a) **Auxiliary regression tasks: architecture error vs. dataset variability.** Mean absolute error (MAE) values for auxiliary regression tasks demonstrate substantially lower error than the inherent dataset variability (standard deviation). Descriptors include mean spike frequency (f_{spk}), intra- and inter-burst frequencies (f_{intra} , f_{inter}), burst duration, and mean number of spikes per burst (#).

Descriptor	f_{spk}	f_{intra}	f_{inter}	Duration	#
MAE	0.11 Hz	2.26 Hz	0.15 Hz	0.81 ms	0.16
Dataset std	3.02 Hz	47.91 Hz	1.11 Hz	7.85 ms	2.25

(b) **Dynamic Input Conductances prediction performance.** The first subtable shows the overall MAE for slow (g_s) and ultra-slow (g_u) DIC components. The second subtable separates MAE values by firing regime reflecting differences in prediction accuracy across distinct neuronal activity modes.

	DIC	g_s	g_u
MAE	2.84	1.75	
DIC	g_s (spiking)	g_s (bursting)	g_u (spiking)
MAE	4.65	0.80	2.29
			1.13

Performance on the primary DIC regression task (Table 1b) highlights important differences between bursting and spiking regimes. Bursting activity involves three distinct timescales [23]: fast spikes, intermediate bursting envelopes, and ultra-slow burst termination. It is well separated in the DIC space, with predictions spanning the full region where $g_s < 0$ (Fig 5A, purple). The model accurately captures this multiscale structure, reflected in relatively low MAE values for bursting DIC predictions.

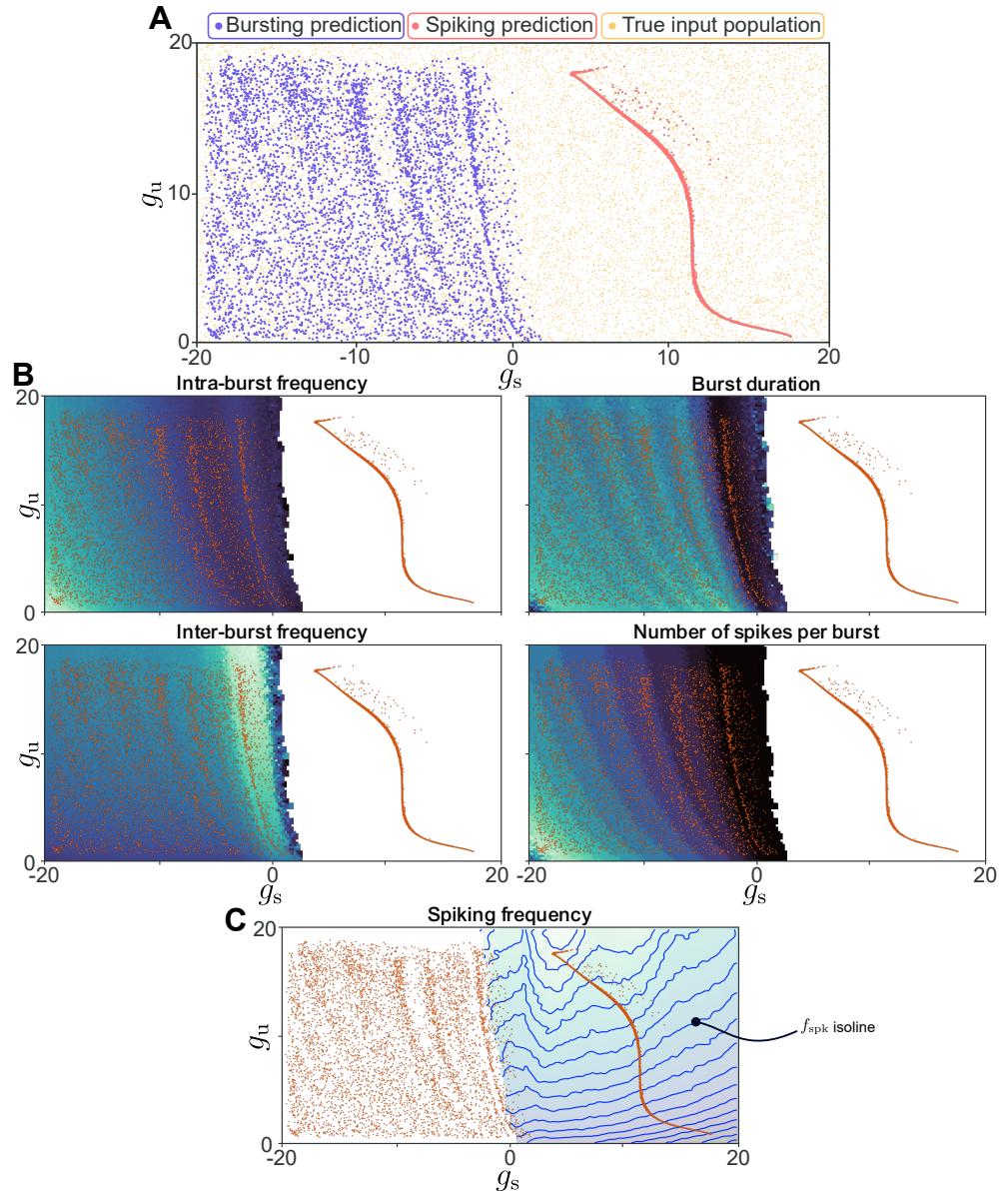


Fig 5. Architecture predictions and test set distributions in the DIC space for the STG neuron model. (A) Comparison of architecture predictions (purple for bursting, red for spiking) with the true test set distribution (yellow) across the full DIC space. Bursting inputs ($g_s < 0$) and spiking inputs ($g_s > 0$) are clearly separated, with bursting predictions distributed broadly and spiking predictions confined along a one-dimensional manifold. (B) Overlay of predicted bursting DICs, showing that predictions span the full range of bursting activity present in the dataset. (C) Overlay of predicted spiking DICs, highlighting the one-dimensional manifold structure across spiking frequencies. Blue isolines indicate spike frequency, illustrating how predictions traverse frequency levels transversally along the manifold. Together, these panels demonstrate that the architecture captures the distinct geometrical organization of bursting and spiking regimes in DIC space, supporting the role of DICs as meaningful low-dimensional intermediates for neuronal activity prediction.

In contrast, spiking activity requires only two timescales: fast spike generation and slow recovery shaping interspike intervals. Here, both g_s and g_u act as overlapping slow negative feedbacks around threshold, which leads to “DIC degeneracy”. Multiple (g_s, g_u) combinations produce indistinguishable spiking outputs (Fig 4B), collapsing predictions onto a one-dimensional manifold in the region $g_s > 0$ (Fig 5A, red). As a result, MAE values appear higher in the spiking regime, not due to poor predictive performance, but because DIC values are inherently ambiguous in this case. This many-to-one mapping illustrates the degeneracy of spiking dynamics in the 3-timescales DIC framework.

This distinction in prediction geometry is illustrated in Fig 5B–C. In the bursting regime (Fig 5B), predicted DICs span the full range of bursting activities observed in the dataset, showing that the model captures the rich variability of bursting metrics. In the spiking regime (Fig 5C), predictions are confined to a one-dimensional manifold that traverses the full spectrum of spiking frequencies. Blue isolines marking spike frequency reveal that predictions vary transversely along this manifold, which confirms that the model encodes meaningful firing rate variations despite parameter ambiguity. Together, these results validate DICs as robust, low-dimensional intermediates linking spike timing to neuronal activity across firing patterns.

Finally, DIC error metrics should be interpreted with caution. The sensitivity of the resulting neuronal activity to exact DIC values is unknown, especially in spiking regimes where big differences in DICs may correspond to indistinguishable activity. Thus, small MAEs are desirable but not strictly necessary. What is really important is the performance of the full generative pipeline: populations generated by iterative compensation using the predicted DICs as input (Fig 1). From this perspective, the architecture provides sufficiently accurate and biologically meaningful predictions to support downstream population generation.

The full generative pipeline reconstructs accurate and diverse degenerate populations from spike times

Our complete generative pipeline transforms spike time sequences into *in silico* degenerate populations of CBMs that reproduce the input activity while spanning diverse underlying parameter sets, using DICs at threshold as intermediates (Fig 1).

The process combines deep learning-based prediction of DIC values from spike times with the iterative compensation algorithm, which samples maximal conductance configurations compatible with these constraints. This approach captures the range of conductance combinations compatible with the input activity, generating populations rather than single solutions.

Qualitative inspection shows close agreement between generated and target activity patterns in both spiking and bursting regimes, with variability confined to levels expected under degeneracy (Fig 6A). Importantly, the method operates solely on spike time input, without requiring voltage traces. This ensures tractability and low computational cost compared to state-of-the art parameter inference methods. However, because the method is blind to subthreshold dynamics, certain features such as after-depolarizations shaping the interspike interval cannot be reproduced, particularly in the spiking regime.

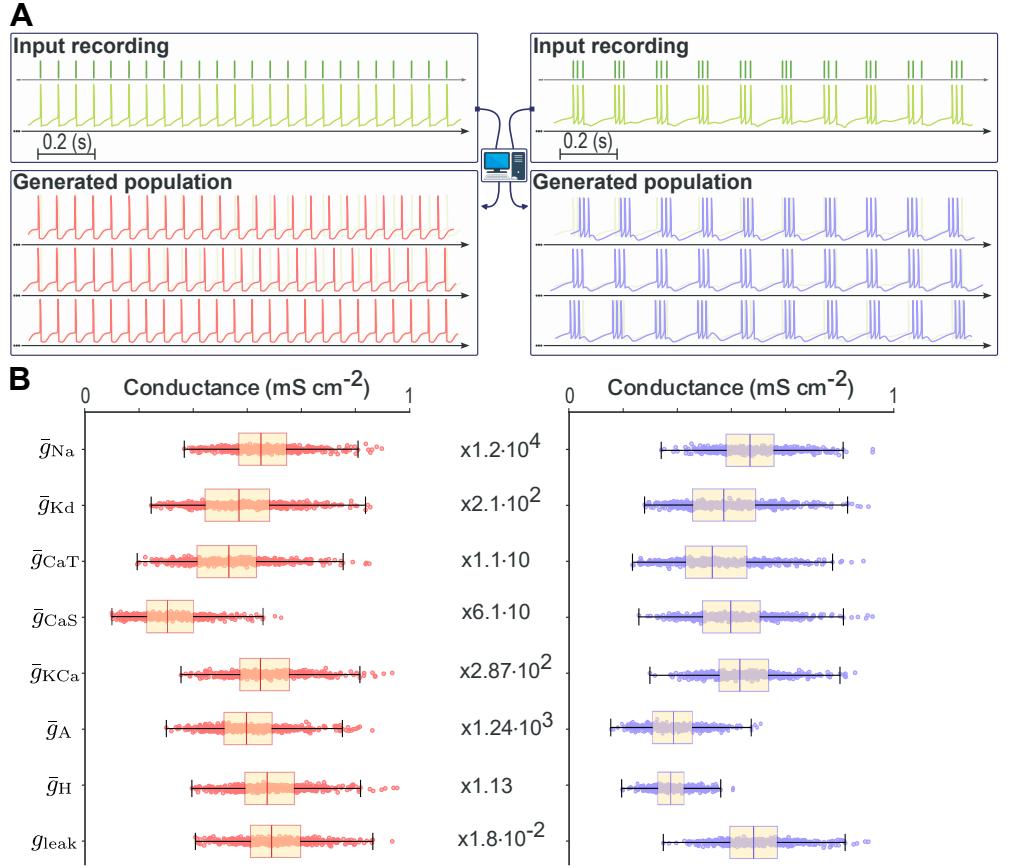


Fig 6. Backbone pipeline output for the stomatogastric ganglion (STG) neuron model. (A) Target spike trains (dark green, top) compared with generated spike trains for spiking (red, left) and bursting (purple, right) regimes, showing accurate reproduction of activity patterns. (B) Distributions of maximal conductances across 500 generated neurons in spiking (red, left) and bursting (purple, right) regimes, demonstrating broad parameter variability despite similar dynamics.

Across 500 generated neurons per regime, maximal conductance distributions are broad yet yield equivalent spiking or bursting dynamics (Fig 6B). This spread in parameter space confirms that the pipeline reconstructs multiple distinct solutions producing similar functional outputs, consistent with population-level degeneracy.

We next provide quantitative comparisons of input and generated firing patterns (Fig 7). We sampled 500 random points in the (g_s, g_u) plane and, for each point, generated a population of 16 neurons using the iterative compensation method. These neurons share identical DIC values at threshold and are therefore degenerate by construction, but still display variability in their firing patterns due to the approximation done through DICs constrained only at threshold, and the iterative

procedure. Such variability remains compatible with degeneracy and reflects biological heterogeneity.

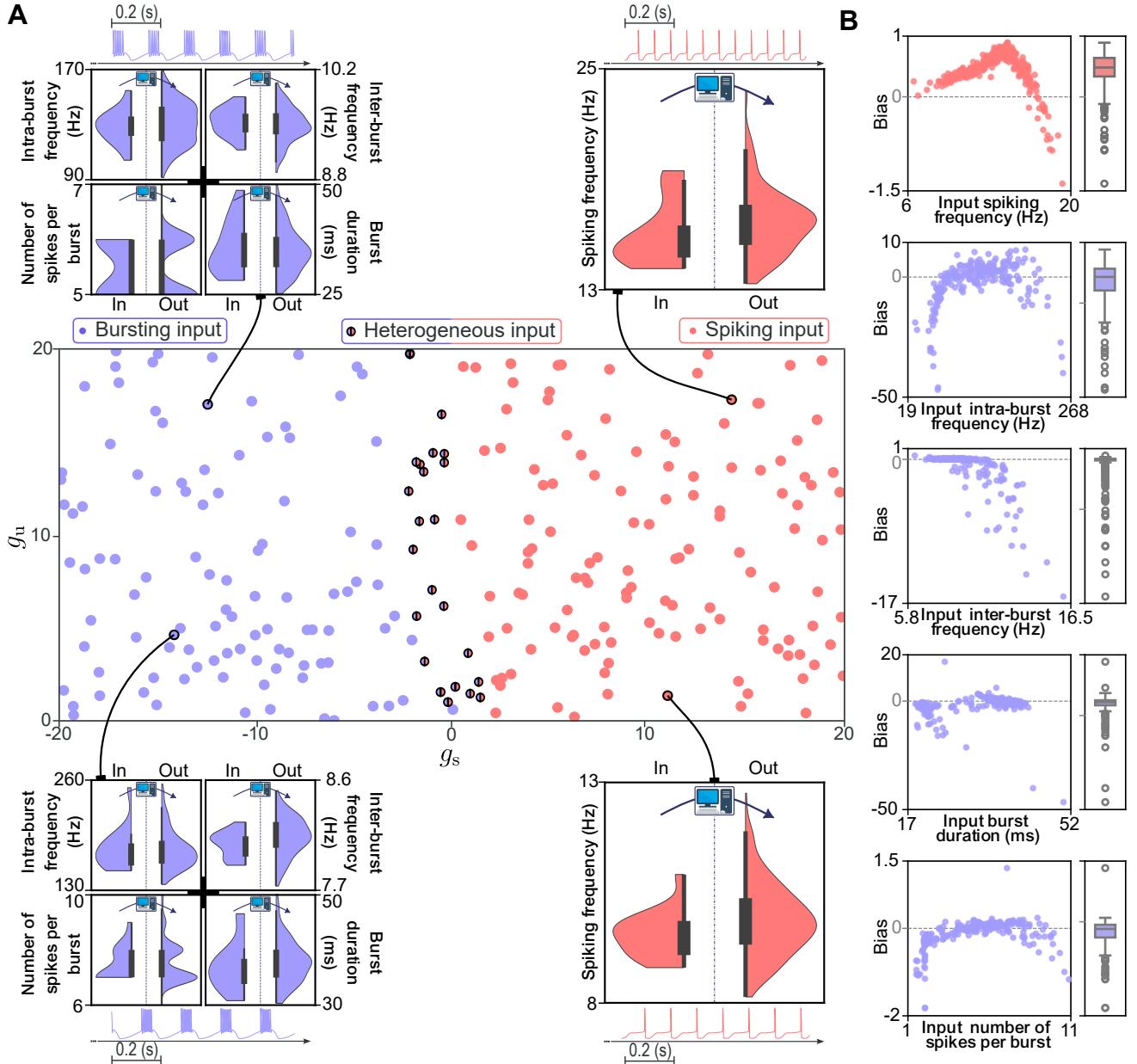


Fig 7. Quantitative comparison of input and generated populations. (A) Sampling of 500 points in (g_s, g_u) space (40% shown for clarity). Purple and red points indicate representative examples; two-colored points denote mixed populations near the spiking–bursting transition ($g_s \approx 0$). Violin plots compare distributions of selected activity metrics between the input population (left) and the corresponding generated population (right) for each example point. (B) Bias analysis across all 500 populations for five activity metrics: spiking frequency, intra-burst frequency, inter-burst frequency, burst duration, and number of spikes per burst. Scatter plots show biases for individual populations; box plots summarize the distribution of biases. Generated populations exhibit minimal bias and closely match the input metrics, with slight increases in spread and rare outliers in extreme conditions.

For each of the 500 input populations, we extracted the spike times of the 16 neurons and passed them through the full pipeline, yielding 16 predicted DICs per point. Each prediction was then used to generate a new population of 16 neurons via iterative compensation, which results in 16×16 neurons per point in the DIC space. This enabled direct comparison of input and output distributions for key activity descriptors: spiking frequency, intra-burst frequency, inter-burst frequency, burst duration, and number of spikes per burst.

Fig 7A shows the sampled DIC space, with representative example points highlighted. Points near $g_s \approx 0$ correspond to mixed populations containing both spiking and bursting neurons, reflecting the transition between regimes. For selected examples, violin plots compare input distributions (left) with reconstructed ones (right). Fig 7B summarizes reconstruction accuracy across all sampled points, reporting the bias (mean difference between generated and input values) for each metric as both scatter plots and box plots.

The generated distributions closely match the inputs, with only slightly increased spread. This is expected since predictions are made from individual spike trains before being aggregated into populations. Spiking metrics are reconstructed with high accuracy: biases are centered near 0.5 Hz, with occasional outliers (about 1 Hz). Bursting metrics are equally well preserved, with biases well below the scale of the features themselves and centered around zero; in particular, the mean number of spikes per burst is reconstructed with high precision. Overall, these results demonstrate that the pipeline faithfully recovers activity features across a wide variety of input populations while preserving degeneracy.

Robustness to stochastic inputs mimicking laboratory conditions

To assess the robustness of the pipeline under realistic temporal variability, we evaluated its performance on spike trains generated from homogeneous Poisson processes and nested Poisson burst generators (see Materials and Methods). Until now, the method had only been exposed to regular, spontaneous activity generated *in silico*. In physiology, however, only pacemaker neurons produce strictly regular activity, while most neurons fire irregularly. Poisson-based inputs thus provide a biologically grounded

baseline benchmark for testing generalization [31–33].

As shown in Fig 8A, despite training exclusively on regular spike patterns, the method successfully produces consistent neuronal populations when presented with stochastic inputs. In both spiking (left, red) and bursting (right, purple) conditions, the output populations closely reproduce the statistical structure of the input spike trains (top, dark green). This demonstrates that the model generalizes beyond its training regime: it averages mean firing rate in spiking and mean burstiness in bursting, producing plausible *in silico* populations even when confronted with irregular activity.

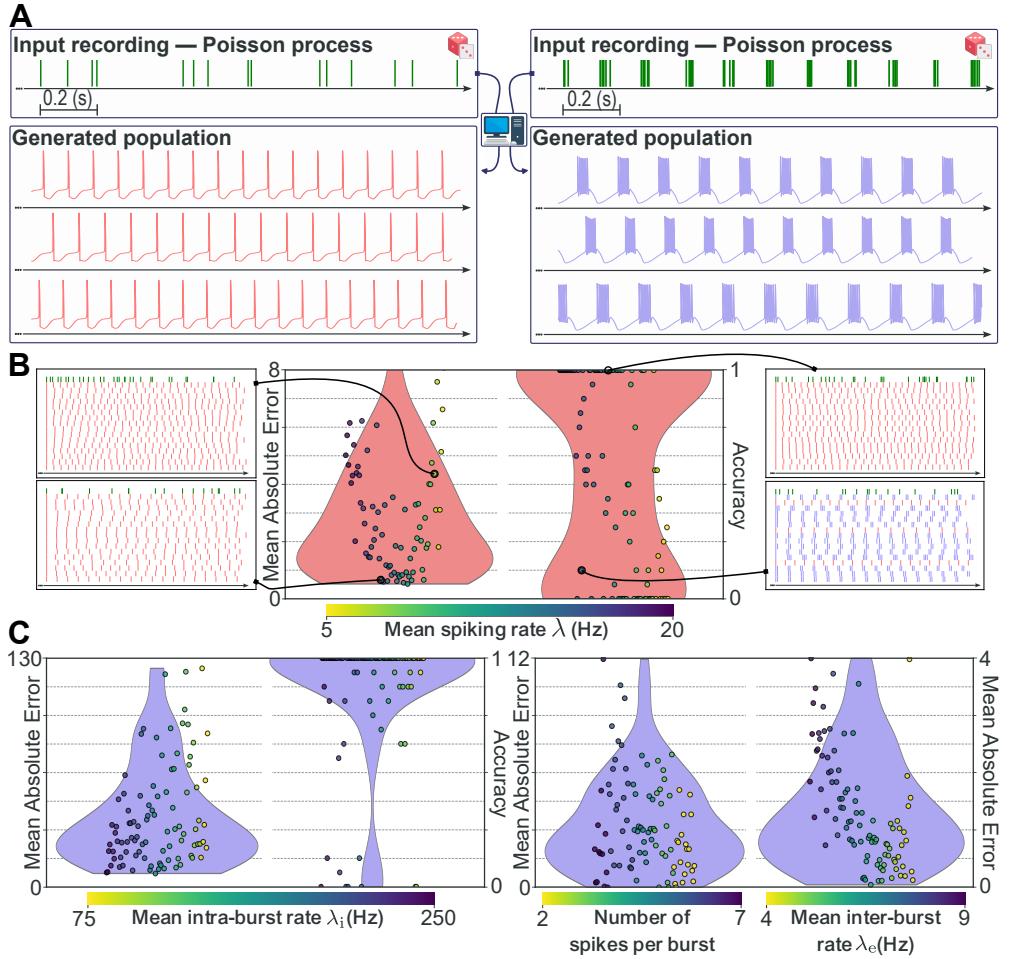


Fig 8. Comparison of input Poisson spike trains with generated spike train populations. (A) Example input spike trains (top, dark green) generated from spiking (left) and bursting (right) Poisson processes, with corresponding generated neuronal population traces (bottom) produced by the pipeline (red for spiking, purple for bursting). (B) Left violin plot shows the mean absolute error (MAE) between input Poisson firing rate λ and population-averaged spiking frequency across 16 generated instances per input; scatter points represent errors for each full generated population relative to its input. Right violin plot displays spiking accuracy, defined as the proportion of generated neurons classified as spikers. Insets show example spike rasters for input (top, dark green) and generated populations (bottom; red for spiking, purple for bursting). (C) Four violin plots for bursting Poisson inputs displaying, from left to right: MAE of intra-burst firing rate λ_i , bursting accuracy (proportion of neurons classified as bursters), MAE of number of spikes per burst, and MAE of inter-burst firing rate λ_e . Scatter points represent errors for each generated population.

For the spiking case, we sampled 100 mean firing rates λ and generated one homogeneous Poisson spike train per value. Each input was passed through the full method to generate a population of 16 CBM instances, with each population represented as a single dot in Fig 8B. The violin plot on the left illustrates the mean

absolute error (MAE) between the input rate λ and the population-averaged spiking frequency of the outputs. Errors remain low across a broad range of λ , with slight increases at low or high rates, reflecting the method limits in producing extreme firing rates. The violin plots on the right show spiking accuracy, measured as the percentage of generated instances classified as spikers. For low to moderate λ , the pipeline reliably produces spiking populations as expected. At high rates, some inputs are interpreted as bursting, because Poisson processes have a high coefficient of variation and lack a refractory period. Closely spaced spikes can therefore resemble bursts, and the method accordingly generates bursters. This indicates that the model provides coherent outputs aligned with the statistical structure of the inputs, even when classification becomes ambiguous. This can be seen as a limitation of the method, but also as a limitation of Poisson processes as stochastic neuronal models [34].

We performed a similar analysis with 100 Poisson bursting spike trains, sampling intra-burst rate λ_i , inter-burst rate λ_e , and number of spikes per burst. Each was passed through the pipeline to generate populations of 16 CBM instances. Fig 8C shows results for four metrics: MAE of λ_i , bursting accuracy, MAE of spike count per burst, and MAE of λ_e . Across all metrics, reconstruction errors remain low. Bursting accuracy is higher than in the spiking case, which is consistent with the tendency of the model to associate high temporal variability with bursting. Only in rare ambiguous cases did the pipeline produce spikers instead of bursters. This robustness arises from the use of DICs as intermediate representations: by reducing spike timing variability to a low-dimensional and interpretable space, the method can generalize to unseen irregular activity without producing incoherent outputs.

Because CBMs do not include intrinsic noise and were simulated without input currents, they generate highly regular dynamics. Despite this, the pipeline captures the average statistics of irregular Poisson inputs and generates populations consistent with spiking or bursting regimes. Even under extreme temporal variability, outputs remain coherent and stable. While Poisson processes are only simplified approximations of biological firing, these results demonstrate the resilience of the method to realistic temporal irregularity. In the next section, we extend this analysis to transfer learning on dopaminergic neuron models in order to further test generalization across neuronal classes.

The pipeline extends to new conductance-based models with minimal retraining

To demonstrate the scalability of our pipeline across distinct CBMs, we applied low rank adaptation (LoRA) [35] to transfer the backbone model trained on the STG dataset to a dopaminergic (DA) neuron model (see Materials and Methods). This setting highlights a drastic change of context: from a bursting neuron of the stomatogastric ganglion, embedded in the gastric mill rhythm, to a slow pacemaker DA neuron with a completely different conductance composition and dynamical repertoire. LoRA allows the STG trained backbone to remain intact for its original task, while efficiently adapting to the DA model with a minimal number of additional parameters.

By leveraging LoRA, we introduce only 39,844 new parameters, which is approximately 34% of the number required for full retraining, and we achieve parameter efficient transfer across highly dissimilar models. This strategy reduces both storage and computational demands and enables scalable adaptation of the pipeline to diverse CBMs. In Supporting Information (see S1 Appendix), we propose a comparison of the architecture trained from scratch on the DA model with the version adapted from the STG.

Despite the marked differences between STG and DA neurons, the adapted architecture achieves strong predictive performance in the DA DIC space (Fig 9A). It recovers key qualitative features also observed in the STG model: a clear separation between spiking and bursting regimes, and the emergence of a spiking manifold. Importantly, the DA DIC space is distinct from that of the STG, which reflects the intrinsic variability of this neuron type, its different conductance composition and underlying dynamical system. Moreover, the DA model exhibits a third activity class, *fast spiking* (a burst that never terminates), which appears at low g_u and negative g_s values as an additional manifold in DIC space. This demonstrates that, although the DIC landscapes differ between models, the same organizational principles emerge, which enables the architecture to generalize effectively.

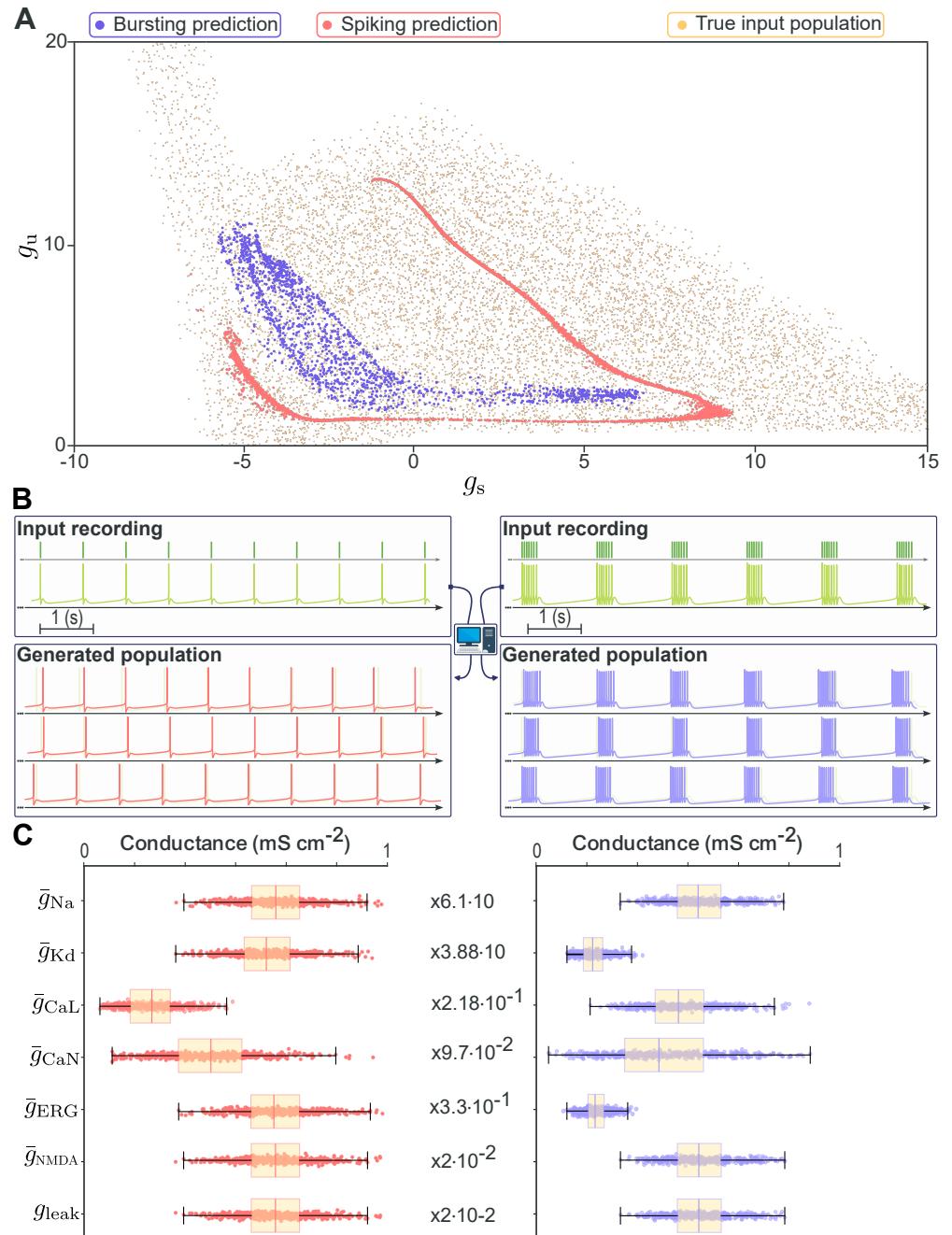


Fig 9. Performance of the adapted pipeline on the DA neuron model. (A) Architecture predictions in the Dynamic Input Conductances (DICs) space, showing clear separation between spiking (red) and bursting (purple) predictions similar to the STG model. (B) Target spike trains (dark green, top) compared with generated spike trains for spiking (red, left) and bursting (purple, right) regimes, showing accurate reproduction of activity patterns. (C) Distributions of maximal conductances across 500 generated neurons in spiking (red, left) and bursting (purple, right) regimes, demonstrating broad parameter variability despite similar dynamics.

Representative generated DA populations are shown in Fig 9B, which confirms that

the pipeline produces spike trains consistent with the target DA activity patterns. As in the STG case, the generated populations preserve degeneracy: despite similar output activity, the distributions of maximal conductances remain broad and heterogeneous (Fig 9C). This variability reflects the intrinsic degeneracy of the DA solution space and underscores the biological relevance of the generated populations.

In summary, LoRA provides an efficient strategy for extending the pipeline to new CBMs with minimal retraining. Even across neuron models with different conductances and activity repertoires, the pipeline preserves both accuracy and degeneracy, highlighting the robustness of the DIC framework as an intermediate representation for generalization across models.

Discussion

Our results demonstrate that the proposed pipeline generalizes across activity regimes, noisy spike train inputs, and even distinct conductance-based models, while consistently preserving degeneracy and accuracy. This highlights the robustness of Dynamic Input Conductances (DICs) as an intermediate representation for linking spike timing to underlying conductance variability. By combining a deep learning architecture that predicts DICs from spike times with an iterative compensation algorithm that enforces these values in conductance-based models, the method provides a computationally efficient and interpretable solution to the inverse problem of parameter inference.

The iterative compensation algorithm substantially improves satisfaction of DIC constraints while preserving intrinsic biological heterogeneity. While illustrated here with the STG neuron model, the method is broadly applicable to other CBMs with nonlinear compensatory structure (e.g., [12, 36]), enabling robust generation of degenerate populations consistent with target constraints.

Despite these advantages, the method has limitations. Its reliance on spike times implies that it is blind to subthreshold dynamics, such as after-depolarizations or detailed interspike interval shapes. However, the DIC framework can naturally capture such phenomena when extended to other voltage values [23]. Building datasets that incorporate DICs measured at multiple voltages, and retraining the architecture accordingly, would allow the method to infer richer neuronal dynamics. Extending the

iterative compensation to integrate these additional DIC constraints could enable the reconstruction of more complex and subtle neuronal behaviors. The possibility of reproducing responses to stimuli ($I_{\text{ext}} \neq 0$) that would further constrain the range of possible conductances is also an opportunity for future work. At the network level, a similar strategy could be employed to incorporate synaptic conductances into the DIC framework, providing a principled way to account for both intrinsic and synaptic contributions to activity [37]. In the long term, these extensions may pave the way for real-time applications, such as adaptive closed-loop neuromodulation [28, 38] or online tuning of neuromorphic devices [39].

Beyond parameter inference, our pipeline offers a new way of representing the conductance space of neurons during experiments. Measuring all ionic conductances in a single neuron is often impractical, as typically only a subset can be probed. The conventional approach of averaging conductance values across populations may lead to fragile or unrepresentative models, because degeneracy and nonlinearities imply that an average neuron may not be valid [27]. In contrast, our framework allows experimentalists to reconstruct a point cloud of conductances in real time using only spike time measurements. Such experiments could directly test how neuromodulators reshape conductance distributions. For example, by comparing the inferred population distributions before and after a bath application, one could visualize how modulation potentially tunes effective conductances and activity regimes.

To facilitate adoption, we also release the full framework as open-source software with a graphical interface [25]. The tool is designed to be intuitive and accessible, lowering the barrier for researchers without programming expertise to generate and explore degenerate CBM populations directly from spike recordings. This practical accessibility complements the methodological contributions, ensuring that the pipeline can be applied in diverse experimental settings.

In summary, we show that combining deep learning with DIC theory enables the efficient and accurate reconstruction of degenerate conductance-based populations from spike times alone. By using DICs as an interpretable low-dimensional representation, the pipeline achieves both robustness and generalization, faithfully reproducing neuronal activity across regimes and models. This work not only advances parameter inference methodologies but also provides a foundation for experimental strategies to

probe how neurons and modulators exploit degeneracy to maintain reliable function. Ultimately, by linking spike timing to conductance variability through interpretable DIC representations, our approach opens the door to scalable, experiment-ready tools for uncovering how degeneracy supports robust neuronal computation.

Materials and methods

Conductance-based models

We use conductance-based models (CBMs) to simulate neuronal dynamics. CBMs are biologically plausible and grounded in biophysical principles, describing the membrane potential V as a function of ionic and leak currents across the membrane. These dynamics are governed by the following equation:

$$C \frac{dV}{dt} + g_{\text{leak}}(V - E_{\text{leak}}) = - \sum_{i \in \mathcal{I}} \bar{g}_i m_i^{p_i}(V, t) h_i^{q_i}(V, t)(V - E_i) + I_{\text{ext}} \quad . \quad (6)$$

Here, C is the membrane capacitance, set to $1 \mu\text{F cm}^{-2}$. The set \mathcal{I} contains all ionic conductances considered for the model. Each ionic current is characterized by its maximal conductance \bar{g}_i , reflecting the maximal effective channel density, and by gating variables m_i and h_i raised to integer powers p_i and q_i , representing activation and inactivation dynamics, respectively. E_i is the Nernst reversal potential of the considered ion. The leak current $I_{\text{leak}} = g_{\text{leak}}(V - E_{\text{leak}})$ models passive ion flow. External input current I_{ext} (e.g., injected current or synaptic input) is set to zero in all simulations ($I_{\text{ext}} = 0$). In this work, we denote $\bar{g} = [\bar{g}_1; \bar{g}_2; \dots; \bar{g}_{|\mathcal{I}|}, g_{\text{leak}}] \in \mathbb{R}_+^{N_{\text{model}}} = \mathcal{G}$ the vector of maximal conductances (extended with the leak conductance) that distinguishes different instances of a model through its N_{model} components. The maximal conductances vector contains the only parameters that are not treated as fixed constants in this work.

Gating variables $X \in \{m_i, h_i\}$ are dimensionless quantities constrained between 0 and 1. They follow first-order voltage-dependent dynamics of the form:

$$\tau_X(V) \frac{dX}{dt} = X_\infty(V) - X \quad , \quad (7)$$

where $\tau_X(V)$ is the voltage-dependent time constant describing how rapidly the gating variable responds to voltage changes, and $X_\infty(V)$ is the steady-state activation or inactivation value.

As a proof of concept for our pipeline, we use two established neuron models: a slightly modified isolated stomatogastric ganglion (STG) neuron model [9], in which we simplify the calcium reversal potential computation to reduce complexity; and a dopaminergic (DA) neuron model [40], in which we block SK channels to enable bursting in a wider range of conductances. These two models exhibit different firing patterns. The ones considered in this work are spiking and bursting for the STG model and slow pacemaking, bursting, and fast spiking for the DA model. Biologically, these two neurons have different roles and timescales. Detailed specifications and parameters for both models are provided in the Supporting Information (see S1 Appendix).

The stomatogastric ganglion neuron model

The STG neuron model is adapted from [9] and includes 7 ionic conductances: g_{Na} (fast sodium), g_{Kd} (delayed rectifier potassium), g_{KCa} (calcium-dependent potassium), g_A (transient A-type potassium), g_{CaS} (slow calcium), g_{CaT} (transient T-type calcium), and g_H (hyperpolarization-activated inward cation). This model explicitly incorporates intracellular calcium concentration dynamics through an additional ordinary differential equation involving $\frac{d\text{Ca}}{dt}$, which modulates calcium-dependent potassium currents through a voltage- and calcium-dependent steady-state gating function $m_{\infty, \text{KCa}}(V, \text{Ca})$.

The dopaminergic neuron model

The DA neuron model [40] includes 6 ionic conductances: g_{Na} (fast sodium), g_{Kd} (delayed rectifier potassium), g_{CaL} (L-type calcium), g_{CaN} (N-type calcium), g_{ERG} (ether-à-go-go-related gene potassium), and g_{NMDA} (NMDA receptor-mediated). It incorporates a magnesium-sensitive current through NMDA influenced by a fixed magnesium concentration.

Dynamic input conductances (DICs)

While CBMs are biologically grounded, they are generally not analytically tractable, as it is not possible to directly infer the membrane potential from \bar{g} without simulation.

Dynamic input conductances (DICs), introduced in [23], address this limitation by providing a scalable and mathematically grounded framework for analyzing CBMs. The DIC formalism decomposes the total membrane response into a sum of timescale-specific components that reflect the dynamical influence of different ionic currents.

Timescale decomposition of membrane dynamics

We partition the influence of membrane currents into three characteristic temporal components. The fast dynamic conductance $g_f(V)$ governs the rapid voltage changes underlying spike upstroke. The slow conductance $g_s(V)$ regulates membrane repolarization and interspike interval (ISI) behavior. Finally, the ultra-slow component $g_u(V)$ accounts for long-term subthreshold integration, adaptation, and shapes the bursting envelope.

In CBMs, these timescale-specific conductances can be computed analytically. For each ionic gating variable or internal state X_i , we evaluate its influence on the membrane current through:

$$\begin{aligned} g_f(V) &= \frac{1}{g_{\text{leak}}} \left[-\frac{\partial \dot{V}}{\partial V} - \sum_i w_{fs,X_i} \left(\frac{\partial \dot{V}}{\partial X_i} \cdot \frac{\partial X_{i,\infty}}{\partial V} \right) \right] \Big|_V , \\ g_s(V) &= \frac{1}{g_{\text{leak}}} \left[- \sum_i (w_{su,X_i} - w_{fs,X_i}) \left(\frac{\partial \dot{V}}{\partial X_i} \cdot \frac{\partial X_{i,\infty}}{\partial V} \right) \right] \Big|_V , \\ g_u(V) &= \frac{1}{g_{\text{leak}}} \left[- \sum_i (1 - w_{su,X_i}) \left(\frac{\partial \dot{V}}{\partial X_i} \cdot \frac{\partial X_{i,\infty}}{\partial V} \right) \right] \Big|_V . \end{aligned} \quad (8)$$

The voltage dependence notation is omitted in the right-hand sides for improved readability. The weighting functions $w_{fs,X_i}(V)$ and $w_{su,X_i}(V)$ distribute the contribution of each variable across timescales based on their voltage-dependent kinetics. These weights are defined as logarithmic distances relative to chosen reference time constants that separate fast, slow, and ultra-slow regimes. As a result, a given ionic current can contribute simultaneously to multiple timescales, depending on the behavior of its gating variables across voltages. The sign convention and the normalization by the leak conductance used here follow the approach in [24], and thus differ from the original formulation in [23]. The exact functional form of these weights and details about DICs are provided in the Supporting Information (see S1 Appendix).

A more convenient formulation of relations (8) is possible through a sensitivity matrix $S(V)$:

$$g_{\text{DICs}}(V) = \begin{bmatrix} g_f(V) \\ g_s(V) \\ g_u(V) \end{bmatrix} = S(V; \bar{g}) \cdot \bar{g} \quad . \quad (9)$$

The sensitivity matrix summarizes a normalized influence of each conductance on the different timescales that should be scaled by the maximal conductances to get the underlying dynamic conductances of a given instance (see S1 Appendix). When S is independent of \bar{g} , we say that the compensation structure is *linear*, whereas the opposite is called a *nonlinear compensation*.

Dimensionality reduction via DICs

In the DIC framework, neuronal excitability is largely characterized by the values of the conductance components at a critical voltage called the threshold potential V_{th} . This threshold corresponds to the voltage at which the neuron is maximally sensitive to changes in its maximal conductance parameters [23]. Following [24], we approximate V_{th} as the first decreasing zero of the total conductance curve $g_t = g_f + g_s + g_u$:

$$g_t(V_{\text{th}}) = 0 \quad \text{with} \quad g_t(V_{\text{th}} - \delta V) > 0 > g_t(V_{\text{th}} + \delta V), \quad \forall \text{ sufficiently small } \delta V > 0. \quad (10)$$

Evaluating the DICs at this voltage yields a compact vector representation of the model:

$$g_{\text{DICs}}(V_{\text{th}}) = \begin{bmatrix} g_f(V_{\text{th}}) \\ g_s(V_{\text{th}}) \\ g_u(V_{\text{th}}) \end{bmatrix} \in \mathbb{R}^3 \quad . \quad (11)$$

This transforms the high-dimensional parameter space $\bar{g} \in \mathcal{G}$ into a low-dimensional space:

$$\bar{g} \in \mathcal{G} \quad \longrightarrow \quad g_{\text{DICs}}(V_{\text{th}}) \in \mathbb{R}^3 \quad . \quad (12)$$

Importantly, this mapping is not injective: many distinct sets of maximal conductances can yield the same DIC vector and therefore similar activity. This property enables controlled exploration of degeneracy in CBM populations.

Generating maximal conductances from DICs

The method introduced in [24] provides a principled way to generate maximal conductance vectors \bar{g} from $g_{\text{DICs}}(V_{\text{th}})$. This approach samples directly from the solution space defined by the DIC constraints at threshold, allowing the generation of degenerate model realizations that exhibit the same spontaneous activity.

The procedure consists of two compensation steps. Each compensation can be described in two parts and aims to make \bar{g} compatible with DIC constraints. The first part samples a subset of the conductance vector, and the second computes the remaining components such that they are compatible with the constraint we want to enforce on DIC values. We proceed in two compensations performed in series in order to (i) first impose a sufficiently negative fast DIC value $g_f(V_{\text{th}})$, such that the resulting \bar{g} will be spontaneously active, and (ii) to modulate the spontaneous activity toward the targeted one, associated with DIC constraints on the slow and ultra-slow values $g_s(V_{\text{th}})$ and $g_u(V_{\text{th}})$, respectively. Therefore, the first compensation aims to constrain $n = 3$ DIC values (the fast, the slow and the ultra-slow), and the second $n = 2$ DIC values (only the slow and the ultra-slow, the fast one being already enforced to be sufficiently negative). The DIC constraints of the second step are either the one that we infer using deep learning \hat{g}^* , or g^* that we get from the uniform sampling when generating the dataset. The DIC constraints of the first step are fixed and chosen such that the population is spontaneously active [24]. More details about the generation steps can be found in Supporting Information (see S1 Appendix).

The first step starts by sampling a subset $\bar{g}_{\text{random}} \sim \mathcal{D}_{\text{generation}}$ of the maximal conductance vector $\bar{g} = [\bar{g}_{\text{random}}; \bar{g}_{\text{comp.}}]$, from $\mathcal{D}_{\text{generation}}$ taken to extend beyond the biological range [7, 24]. To impose $n = 3$ DIC values, the subset \bar{g}_{random} is such that it has $N_{\text{model}} - n$ components. The remaining n components of \bar{g} correspond to the subset $\bar{g}_{\text{comp.}}$, whose values are determined through the linear compensation step described by Eq (2). A convenient way of writing this compensation step is by decomposing the sensitivity matrix into $S = [S_{\text{random}}; S_{\text{comp.}}]$ following \bar{g} :

$$\underbrace{S_{\text{comp.}}(V_{\text{th}}) \cdot \bar{g}_{\text{comp.}}}_{A} = \underbrace{g_{\text{DICs}}(V_{\text{th}}) - S_{\text{random}}(V_{\text{th}}) \cdot \bar{g}_{\text{random}}}_{b} . \quad (13)$$

We introduce the following notations, $A := S_{\text{comp.}}(V_{\text{th}}) \in \mathbb{R}^{n \times (N_{\text{model}} - n)}$ and $b := g_{\text{DICs}}(V_{\text{th}}) - S_{\text{random}}(V_{\text{th}}) \cdot \bar{g}_{\text{random}} \in \mathbb{R}^n$ that will be helpful to describe our iterative compensation algorithm.

The second step follows a similar structure, starting from the \bar{g} resulting from the first compensation, a system similar to (13) is built based on a decomposition into two subsets that can differ from the one used during the initial compensation (see S1 Appendix). Since we start from the results of the first step, no sampling is done at the beginning of this compensation step.

The generation procedure described above relies on two approximations. First, the threshold potential V_{th} must be chosen a priori, rather than determined consistently from the total conductance curve of the compensated solution. Second, the generation step is exact only if the compensation structure is linear in the compensated conductances. In [24], when the sensitivity matrix depends on the compensated conductances, it is approximated using a fixed default value $\hat{g}_{\text{comp.}}$, i.e. $S(V_{\text{th}}; \hat{g}_{\text{comp.}}) \approx \hat{S}(V_{\text{th}})$. It is this approximation that is then used to build the system (13). However, it may introduce substantial residuals and compromise the reliability of the generation procedure.

Iterative Compensation Algorithm

Iterative solution to the nonlinear compensation problem

While the linear compensation (equation 13) works well for simpler models like the DA neuron model, where sensitivities depend only on voltage, it becomes inaccurate in complex CBMs. In the STG model, intracellular calcium dynamics introduce nonlinear dependencies, and the sensitivity matrix depends on the compensated conductances.

To address this, we introduce the iterative compensation algorithm, which generalizes the linear scheme by applying multiple compensation steps iteratively. At each iteration, the sensitivity matrix approximation is recalculated based on the current conductances $\hat{S}^{(k+1)}(V_{\text{th}}) \approx S(V_{\text{th}}; \bar{g}^{(k)})$, and we compute the new compensated values through:

$$A(\bar{g}_{\text{comp.}}^{(k)}) \cdot \bar{g}_{\text{comp.}}^{(k+1)} = b(\bar{g}_{\text{comp.}}^{(k)}), \quad k = 0, \dots, K + 1 \quad . \quad (14)$$

Starting from an initial guess $\bar{g}_{\text{comp.}}^{(0)}$, the process is repeated until the residual norm between the target and actual DIC values is sufficiently small. During the dataset generation and inference processes, we use $K = 5$ iterations as a default value, and the default value $\hat{g}_{\text{comp.}}$ from [24] as the initial guess.

Choice of a priori threshold voltage

The generation requires evaluating sensitivities at a fixed voltage V_{th} . However, the actual threshold voltage of a conductance vector \bar{g} is unknown before knowing \bar{g} . To circumvent this, we approximate V_{th} by a constant value shared across all instances of a given CBM. This approximation is supported empirically by sampling 4000 conductance vectors from broad conductance distributions $\mathcal{D}_{\text{analysis}}$ and computing their theoretical V_{th} using Eq (10). The resulting distribution is narrow and unimodal, allowing selection of the median (close to the mean) as a stable reference for generation. We use -51 mV and -55.5 mV for the STG and the DA, respectively. These fixed values remain stable when doubling the sample size, confirming their suitability for sensitivity evaluation. Details are provided in Supporting Information (see S1 Appendix).

Synthetic dataset generation

We simulate CBMs using Python 3.11 with the SciPy library [41]. The system of differential equations is solved using the `solve_ivp` function with the BDF method [42], well-suited for stiff systems such as models with multiple timescales. Although RK45 was tested, BDF consistently provided superior stability and accuracy. The maximum allowed time step is set to 0.05 ms, matching the spike time sampling period, with an adaptive step size during integration.

Simulations are run for a total duration of 5000 ms for the STG model and 12 000 ms for the DA model. To avoid transient effects, the first 3000 ms of each simulation is discarded. Our implementation leverages multi-core CPU architectures to parallelize simulations, accelerating dataset generation.

From each simulated membrane potential trace $V(t)$, we emulate experimental recordings that only capture spike times:

$$V(t) \xrightarrow{\text{transformed into}} x = [t_1, t_2, \dots, t_{N_{\text{spikes}}}], \quad \text{where } t_1 < t_2 < \dots < t_{N_{\text{spikes}}} .$$

The synthetic dataset is generated by uniformly sampling values in the g_s - g_u space to ensure broad coverage. For each sampled DICs vector, we generate 16 instances with distinct \bar{g} using our iterative compensation algorithm. Bounds for DICs sampling are empirically derived from extensive model sampling of $\mathcal{D}_{\text{analysis}}$, and are $[-20; 20] \times [0; 20]$ for the STG and $[-10; 15] \times [0; 20]$ for the DA. Details can be found in Supporting Information (see S1 Appendix).

Dataset sizes are as follows: for the STG model, 1,000,000 instances are generated for training, 200,000 for validation, and 200,000 for testing. Validation sets are used for hyperparameter tuning, while the test set is reserved for final performance evaluation. Instances sharing the same DICs vector are kept within the same dataset partition to prevent data leakage. For the DA model, a reduced dataset (about 40% of the STG dataset size) is used, this fraction being determined through an ablation study on the STG data size. Transfer learning techniques employing LoRA adapters facilitate model adaptation with this smaller dataset.

Overall, the datasets consist of pairs (spike times, target DICs) = (x, g^*) , with underlying maximal conductances not used directly by the deep learning pipeline, which focuses on predicting DICs from spike times.

Instances are classified as *silent* if there are less than 3 spikes detected in the simulation; they are classified as *bursting* if the coefficient of variation of the inter-spike intervals (ISIs) is greater than 0.15, and as *spiking* otherwise (Fig 3).

Deep learning architecture and training

Problem formulation as supervised learning

We frame the prediction of DIC values from spike trains as a supervised learning task. Specifically, we aim to learn a parametric function f_θ such that:

$$f_\theta : \mathbb{R}_+^{1 \times *} \rightarrow \mathbb{R}^2, \quad x \mapsto \hat{g}^*, \quad \hat{g}^* \approx g^* , \quad (15)$$

where the input $x = [t_1, t_2, \dots, t_{N_{\text{spikes}}}]$ is a variable-length sequence of spike times, and the output $\hat{g}^* = (\hat{g}_s^*, \hat{g}_u^*)$ is an estimate of the true target values g^* . The notation $1 \times *$ emphasizes the arbitrary length of the input sequences of a single measure per element.

This regression task poses two main challenges: (1) the input sequences are of variable-length, while deep neural networks typically require fixed-size inputs; and (2) learning meaningful latent representations from raw spike data, without relying on manually engineered summary statistics, is inherently difficult.

To address these challenges, we design an encoder-decoder architecture that transforms raw spike trains into fixed-dimensional latent representations, from which the DIC values are regressed.

Network architecture

Our architecture is represented in Fig 10, and comprises an attention-based encoder [43] and a multi-headed decoder. The encoder (Fig 10B) maps the variable-length input sequence to a fixed-size latent vector $z_{\text{latent}} \in \mathbb{R}^{d_{\text{latent}}}$, addressing the first challenge. The decoder (Fig 10C) then processes this latent vector via multiple parallel heads to facilitate multi-task learning, thereby improving the structure and expressivity of the latent space.

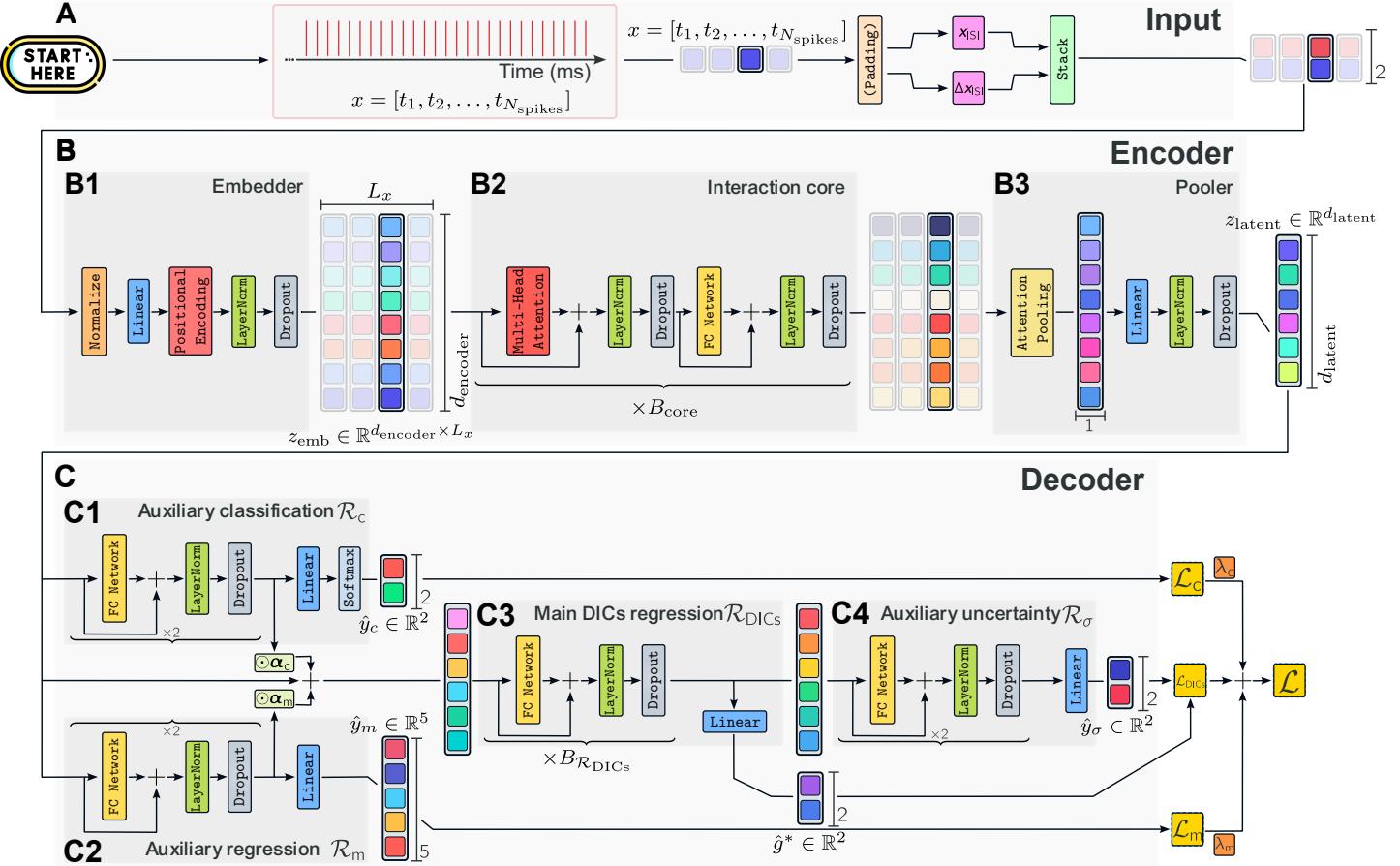


Fig 10. The deep learning architecture. (A) The input to the model consists of spike time sequences, from which ISIs and delta ISIs are extracted. These features are then stacked and fed into the encoder. (B) The encoder processes the input through three main components: the embedder, the interaction core, and the pooler. The embedder transforms the input sequence into a normalized, higher-dimensional representation. The interaction core processes this representation using multi-head attention mechanisms and fully-connected networks. The pooler aggregates the variable-length representation into a fixed-size latent representation. (C) The decoder transforms the fixed-size latent representation into the DICs space. It includes auxiliary tasks for classifying neuronal activity and regressing electrical activity metrics, as well as predicting uncertainty measures.

During training, the decoder includes four distinct output heads operating on the shared latent representation:

- **Main regression head $\mathcal{R}_{\text{DICs}}$** (Fig 10C3) : $\mathbb{R}^{d_{\text{latent}}} \rightarrow \mathbb{R}^2$ predicts the DIC values.
- **Classification head \mathcal{R}_c** (Fig 10C1) : $\mathbb{R}^{d_{\text{latent}}} \rightarrow \Delta^1$ classifies the input sequence as spiking or bursting based on inter-spike interval (ISI) variability.
- **Activity metrics head \mathcal{R}_m** (Fig 10C2) : $\mathbb{R}^{d_{\text{latent}}} \rightarrow \mathbb{R}^5$ regresses five firing metrics. For spiking sequences, this includes mean firing rate. For bursting

sequences, it includes mean intra-burst frequency, inter-burst frequency, burst duration, and number of spikes per burst.

- **Uncertainty head \mathcal{R}_σ** (Fig 10C4) : $\mathbb{R}^{d_{\text{latent}}} \rightarrow \mathbb{R}^2$ estimates $\log \sigma_{g^*}^2$ and $\log \sigma_{g_u^*}^2$, which are used for heteroscedastic loss weighting.

At inference time, only the output from $\mathcal{R}_{\text{DICs}}$ is used downstream in the pipeline as the target values for the generation process. The auxiliary heads are used during training to regularize the encoder.

Encoder design

The encoder consists of three components: the embedder (Fig 10B1), the interaction core (Fig 10B2), and the pooler (Fig 10B3).

Given a spike train x , we first compute the sequence of ISIs, denoted as $x_{\text{ISI}} = \Delta x = [t_2 - t_1, t_3 - t_2, \dots]$. To enhance burst detection, this sequence is augmented with second-order differences $\Delta x_{\text{ISI}} = [\Delta x_2 - \Delta x_1, \Delta x_3 - \Delta x_2, \dots]$. The resulting two-dimensional feature matrix is:

$$x_{\text{features}} = \begin{bmatrix} \log(1 + x_{\text{ISI},1}) & \log(1 + x_{\text{ISI},2}) & \dots \\ \Delta x_{\text{ISI},1} & \Delta x_{\text{ISI},2} & \dots \end{bmatrix} \in \mathbb{R}_+^{2 \times *} . \quad (16)$$

The logarithmic transformation $\log(1 + x_{\text{ISI}})$ stabilizes the ISI distribution.

The embedder converts this into a structured feature representation suitable for downstream processing. The feature matrix is standardized using statistics computed from the training set:

$$x_{\text{norm}} = \frac{x_{\text{features}} - \mu_{\text{train}}}{\sigma_{\text{train}}} \in \mathbb{R}^{2 \times *} . \quad (17)$$

The normalized sequence is projected into a higher-dimensional space $\mathbb{R}^{d_{\text{encoder}} \times *}$ and enriched with sinusoidal positional encoding P_{sin} [43]:

$$z_{\text{emb}} = Wx_{\text{norm}} + P_{\text{sin}} \in \mathbb{R}^{d_{\text{encoder}} \times *} . \quad (18)$$

Layer normalization [44] and dropout [45] are applied to the embeddings before they

are processed by the interaction core.

The interaction core consists of B_{core} stacked transformer blocks. Each block performs multi-head self-attention followed by a position-wise fully connected (FC) network. A fully connected network is defined by two hidden linear layers, with a GELU activation function [46] and a dropout layer in between. Residual connections, layer normalization, and dropout are applied to each sublayer to stabilize training and improve generalization. This structure captures both local and global dependencies in the spike train.

Finally, the output of the interaction core is aggregated using self-attention pooling, which reduces the representation from $\mathbb{R}^{d_{\text{encoder}} \times *}$ to $\mathbb{R}^{d_{\text{encoder}}}$. A linear projection maps this pooled representation into the latent space:

$$z_{\text{latent}} = W_{\text{pool}}z + b_{\text{pool}} \in \mathbb{R}^{d_{\text{latent}}} . \quad (19)$$

Decoder design

Each decoder head consists of residual fully connected (FC) blocks, each followed by layer normalization and dropout. All auxiliary heads (Fig 10C1-2,4) use two such blocks, concluding with a final linear layer; a softmax activation is applied after the classification head.

The final latent outputs of the auxiliary heads \mathcal{R}_c and \mathcal{R}_m are integrated with the encoder output via learnable element-wise mixing:

$$x_{\mathcal{R}_{\text{DICs}}} = z_{\text{latent}} + \alpha_m \odot z_m + \alpha_c \odot z_c , \quad (20)$$

where $\alpha_m, \alpha_c \in \mathbb{R}^{d_{\text{latent}}}$ are learnable parameters and z_m, z_c are the auxiliary latents.

The main DIC regression head $\mathcal{R}_{\text{DICs}}$ is made of $B_{\mathcal{R}_{\text{DICs}}}$ blocks. The final latent output serves as input to the uncertainty head \mathcal{R}_σ .

Training procedure and loss functions

The total loss is a weighted sum of the primary and auxiliary objectives:

$$\mathcal{L} = \mathcal{L}_{\text{DICs}} + \lambda_m \mathcal{L}_m + \lambda_c \mathcal{L}_c , \quad (21)$$

where λ_m and λ_c are hyperparameters that control the importance of auxiliary tasks.

The primary loss $\mathcal{L}_{\text{DICs}}$ is a heteroscedastic Huber loss:

$$\begin{aligned} \mathcal{L}_{\text{DICs}} = & \frac{1}{N} \sum_{i=1}^N \left[\frac{1}{\sigma_{g_s^*, i}^2} \cdot \text{Huber}(g_{s,i}^* - \hat{g}_{s,i}) + \log(\sigma_{g_s^*, i}^2) \right. \\ & \left. + \frac{1}{\sigma_{g_u^*, i}^2} \cdot \text{Huber}(g_{u,i}^* - \hat{g}_{u,i}) + \log(\sigma_{g_u^*, i}^2) \right] , \end{aligned} \quad (22)$$

where the Huber loss [47] is defined as:

$$\text{Huber}(a) = \begin{cases} \frac{1}{2}a^2 & \text{if } |a| \leq \delta, \\ \delta(|a| - \frac{1}{2}\delta) & \text{otherwise,} \end{cases} \quad (23)$$

with $\delta = 1$. This loss is robust to outliers and incorporates predictive uncertainty that smooth learning especially for spiking predictions where degeneracy leads to large errors.

The auxiliary loss for activity metrics is a masked mean squared error:

$$\mathcal{L}_m = \frac{1}{N} \sum_{i=1}^N \| (y_{m,i} - \hat{y}_{m,i}) \odot m_i \|^2 , \quad (24)$$

where the binary mask m_i selects relevant metrics based on the activity type:

$$m_i = \begin{cases} [1, 0, 0, 0, 0] & \text{if } y_{c,i} = 1 \text{ (spiking),} \\ [0, 1, 1, 1, 1] & \text{if } y_{c,i} = 2 \text{ (bursting).} \end{cases} \quad (25)$$

The classification loss is standard cross-entropy:

$$\mathcal{L}_c = -\frac{1}{N} \sum_{i=1}^N [\mathbb{I}(y_{c,i} = 1) \log(\hat{y}_{c,i,1}) + \mathbb{I}(y_{c,i} = 2) \log(\hat{y}_{c,i,2})] . \quad (26)$$

A balanced version is used during training on the DA model due to dataset imbalance.

To improve generalization, we apply three forms of data augmentation during training: (1) random cropping of spike trains to a window $D \sim \mathcal{U}[N_{\text{spikes}}/2, N_{\text{spikes}}]$, (2) Gaussian noise $\epsilon_i \sim \mathcal{N}(0, (2 \text{ ms})^2)$ added to spike times, and (3) 5% spike dropout. These augmentations simulate experimental variability and are applied independently for each training sample at each update.

Hyperparameters are optimized via extended random search over 100

configurations [48], with performance evaluated using the primary DIC loss $\mathcal{L}_{\text{DIC}_S}$ on the validation set. Optimization uses AdamW [49] and a cosine annealing learning rate schedule with warm restarts. Details and results are provided in the Supporting Information (see S1 Appendix).

Evaluation metrics

Model evaluation is carried out at two levels: training/validation and final test performance. During training, we use the heteroscedastic loss $\mathcal{L}_{\text{DIC}_S}$ as the primary metric to assess whether the model is learning effectively. Hyperparameter optimization is performed based on the value of $\mathcal{L}_{\text{DIC}_S}$ on the validation set \mathcal{T}_{val} . For test-time evaluation on $\mathcal{T}_{\text{test}}$, we report the mean absolute error (MAE) between predicted and ground truth DIC values. We also evaluate the performance of the auxiliary tasks. The auxiliary regression head is evaluated using MAE. The auxiliary classification head is evaluated using balanced accuracy to account for potential class imbalance. These auxiliary metrics serve to verify that the network learns a meaningful and structured latent representation of the input spike trains.

Transfer to the DA model

We perform a transfer from the STG neuron model to the DA neuron model. The same data generation pipeline and neural network architecture are reused, with the exception of the introduction of parameter-efficient fine-tuning using Low-Rank Adaptation (LoRA) [35].

We introduce LoRA adapters in the linear layers of the network, and attention layers remain unmodified. The majority of parameters from the original STG-trained pipeline are frozen during training, and only the newly introduced LoRA parameters are updated. The input normalization layer is recalculated based on the DA training set. The supplementary Fig S4 (S1 Appendix) illustrates the modified architecture, indicating which components are frozen and which are adapted.

The transfer introduces one additional hyperparameter, r , which controls the rank of the LoRA adapter matrices. To select an appropriate configuration, we perform a grid search over $r \in \{2, 4, 8, 16, 32, 48, 64\}$. Each configuration is trained on the DA dataset

using the same optimization strategy and training protocol as for the STG model. The best results are observed for $r = 32$. Details are available in Supporting Information (see S1 Appendix).

The dataset for the DA model is generated using the same procedure as for the STG model, including rejection of silent populations and extraction of the corresponding DIC values.

During dataset generation, an initial scan of the sampled populations reveals a substantial class imbalance: a large proportion of silent populations, and a relatively small number of bursting cases compared to spiking ones. Silent populations are discarded. The number of sampled configurations is set to yield approximately 25,000 active (non-silent) populations in total.

Generation of Poisson and bursting spike trains

To assess the robustness of our pipeline to biologically plausible timing variability, we generate synthetic input spike trains using two stochastic models: a homogeneous Poisson process and a burst-extended variant. The ranges used to sample the processes parameters are provided in Supporting Information (see S1 Appendix).

Homogeneous Poisson process

Spike times are generated from a Poisson process with constant rate λ . Inter-spike intervals (ISIs) are sampled from an exponential distribution:

$$f_{\text{ISI}}(\tau) = \lambda e^{-\lambda\tau}, \quad \tau \geq 0 \quad ,$$

resulting in irregular and memoryless spike trains.

Poisson burst generator

To simulate bursting behavior, we use a nested Poisson process with two timescales:

1. **Burst onsets** follow a Poisson process with rate λ_e , with inter-burst intervals distributed as:

$$f_{\text{inter}}(\Delta B) = \lambda_e e^{-\lambda_e \Delta B} \quad .$$

2. **Intra-burst spikes** are generated from an independent exponential distribution with rate λ_i :

$$f_{\text{intra}}(\delta) = \lambda_i e^{-\lambda_i \delta} .$$

Each burst contains a fixed number of spikes, making it the only deterministic component of the process.

Supporting information

S1 Appendix Details and additional experiments We provide all the elements required to reproduce the results presented in this paper, as well as additional experiments and their results. This includes, for example, model equations, the ranges of values used, and hyperparameter tuning procedures.

References

1. Tiesinga P, Fellous JM, Sejnowski TJ. Regulation of Spike Timing in Visual Cortical Circuits. *Nature Reviews Neuroscience*. 2008;9(2):97-107. arXiv:18200026. doi:10.1038/nrn2315.
2. Mainen ZF, Sejnowski TJ. Reliability of Spike Timing in Neocortical Neurons. *Science (New York, NY)*. 1995;268(5216):1503-6. arXiv:7770778. doi:10.1126/science.7770778.
3. Williams AH, Poole B, Maheswaranathan N, Dhawale AK, Fisher T, Wilson CD, et al. Discovering Precise Temporal Patterns in Large-Scale Neural Recordings through Robust and Interpretable Time Warping. *Neuron*. 2020;105(2):246-59.e8. arXiv:31786013. doi:10.1016/j.neuron.2019.10.020.
4. Marder E. Neuromodulation of Neuronal Circuits: Back to the Future. *Neuron*. 2012;76(1):1-11. doi:10.1016/j.neuron.2012.09.010.
5. Bargmann CI, Marder E. From the Connectome to Brain Function. *Nature Methods*. 2013 Jun;10(6):483-90. doi:10.1038/nmeth.2451.

6. McCormick DA, Nestvogel DB, He BJ. Neuromodulation of Brain State and Behavior. *Annual Review of Neuroscience*. 2020 Jul;43(Volume 43, 2020):391-415. doi:10.1146/annurev-neuro-100219-105424.
7. Goaillard JM, Marder E. Ion Channel Degeneracy, Variability, and Covariation in Neuron and Circuit Resilience. *Annual Review of Neuroscience*. 2021;44:335-57. doi:10.1146/annurev-neuro-092920-121538.
8. Goldman MS, Golowasch J, Marder E, Abbott LF. Global Structure, Robustness, and Modulation of Neuronal Models. *Journal of Neuroscience*. 2001;21(14):5229-38. arXiv:11438598. doi:10.1523/JNEUROSCI.21-14-05229.2001.
9. Liu Z, Golowasch J, Marder E, Abbott LF. A Model Neuron with Activity-Dependent Conductances Regulated by Multiple Calcium Sensors. *Journal of Neuroscience*. 1998;18(7):2309-20. arXiv:9502792. doi:10.1523/JNEUROSCI.18-07-02309.1998.
10. Golowasch J, Abbott LF, Marder E. Activity-Dependent Regulation of Potassium Currents in an Identified Neuron of the Stomatogastric Ganglion of the Crab Cancer Borealis. *Journal of Neuroscience*. 1999;19(20):RC33-3. arXiv:10516335. doi:10.1523/JNEUROSCI.19-20-j0004.1999.
11. Taylor AL, Goaillard JM, Marder E. How Multiple Conductances Determine Electrophysiological Properties in a Multicompartment Model. *Journal of Neuroscience*. 2009;29(17):5573-86. arXiv:19403824. doi:10.1523/JNEUROSCI.4438-08.2009.
12. Yu N, Canavier CC. A Mathematical Model of a Midbrain Dopamine Neuron Identifies Two Slow Variables Likely Responsible for Bursts Evoked by SK Channel Antagonists and Terminated by Depolarization Block. *The Journal of Mathematical Neuroscience (JMN)*. 2015;5(1):5. doi:10.1186/s13408-015-0017-6.
13. Druckmann S, Banitt Y, Gidon AA, Schürmann F, Markram H, Segev I. A Novel Multiple Objective Optimization Framework for Constraining Conductance-Based Neuron Models by Experimental Data. *Frontiers in Neuroscience*. 2007 Oct;1. doi:10.3389/neuro.01.1.1.001.2007.

14. Prinz AA, Billimoria CP, Marder E. Alternative to Hand-Tuning Conductance-Based Models: Construction and Analysis of Databases of Model Neurons. *Journal of Neurophysiology*. 2003;90(6):3998-4015.
doi:10.1152/jn.00641.2003.
15. Shepardson D. Algorithms for Inverting Hodgkin-Huxley Type Neuron Models [Ph.D. Thesis]. Georgia Institute of Technology; 2009. Available from: <http://hdl.handle.net/1853/31686>.
16. Huys QJM, Ahrens MB, Paninski L. Efficient Estimation of Detailed Single-Neuron Models. *Journal of Neurophysiology*. 2006 Aug;96(2):872-90.
doi:10.1152/jn.00079.2006.
17. Burghi TB, Schoukens M, Sepulchre R. Feedback Identification of Conductance-Based Models. *Automatica*. 2021 Jan;123:109297.
doi:10.1016/j.automatica.2020.109297.
18. Burghi TB, O'Leary T, Sepulchre R. Distributed Online Estimation of Biophysical Neural Networks. In: 2022 IEEE 61st Conference on Decision and Control (CDC); 2022. p. 628-34. doi:10.1109/CDC51059.2022.9993018.
19. Meng L, Kramer MA, Eden UT. A Sequential Monte Carlo Approach to Estimate Biophysical Neural Models from Spikes. *Journal of Neural Engineering*. 2011;8(6):065006. arXiv:22058277. doi:10.1088/1741-2560/8/6/065006.
20. Meng L, Kramer MA, Middleton SJ, Whittington MA, Eden UT. A Unified Approach to Linking Experimental, Statistical and Computational Analysis of Spike Train Data. *PLOS ONE*. 2014;9(1):e85269.
doi:10.1371/journal.pone.0085269.
21. Ditlevsen S, Samson A. Estimation in the Partially Observed Stochastic Morris–Lecar Neuronal Model with Particle Filter and Stochastic Approximation Methods. *The Annals of Applied Statistics*. 2014 Jun;8(2):674-702.
doi:10.1214/14-AOAS729.

22. Gonçalves PJ, Lueckmann JM, Deistler M, Nonnenmacher M, Öcal K, Bassetto G, et al. Training Deep Neural Density Estimators to Identify Mechanistic Models of Neural Dynamics. *eLife*. 2020;9:e56261. doi:10.7554/eLife.56261.
23. Drion G, Franci A, Dethier J, Sepulchre R. Dynamic Input Conductances Shape Neuronal Spiking. *eNeuro*. 2015;2(1). doi:10.1523/ENEURO.0031-14.2015.
24. Fyon A, Franci A, Sacré P, Drion G. Dimensionality Reduction of Neuronal Degeneracy Reveals Two Interfering Physiological Mechanisms. *PNAS Nexus*. 2024 Oct;3(10):pgae415. doi:10.1093/pnasnexus/pgae415.
25. Brandoit J. Spike2Pop: Bridging Experimental Neuroscience and Computational Modeling [Software]. GitHub;
2025. Version 1.1.1. Software available at <https://github.com/julienbrandoit/Spike2Pop---Bridging-Experimental-Neuroscience-and-Computational-Modeling>
26. Marder E, Taylor AL. Multiple Models to Capture the Variability in Biological Neurons and Networks. *Nature Neuroscience*. 2011;14(2):133-8. doi:10.1038/nn.2735.
27. Golowasch J, Goldman MS, Abbott LF, Marder E. Failure of Averaging in the Construction of a Conductance-Based Neuron Model. *Journal of Neurophysiology*. 2002;87(2):1129-31. doi:10.1152/jn.00412.2001.
28. Fyon A, Sacré P, Franci A, Drion G. Reliable Neuromodulation from Adaptive Control of Ion Channel Expression. *IFAC-PapersOnLine*. 2023;56(2):458-63. doi:10.1016/j.ifacol.2023.10.1610.
29. Brandoit J, Ernst D, Drion G, Fyon A. Degenerate Conductance-Based Models Populations Datasets: STG and DA models. Zenodo; 2025. doi:10.5281/zenodo.16912161.
30. Nadim F, Bucher D. Neuromodulation of Neurons and Synapses. *Current Opinion in Neurobiology*. 2014;29:48-56. doi:10.1016/j.conb.2014.05.003.
31. Stevens CF, Zador AM. Input Synchrony and the Irregular Firing of Cortical Neurons. *Nature Neuroscience*. 1998;1(3):210-7. doi:10.1038/659.

32. Softky WR, Koch C. The Highly Irregular Firing of Cortical Cells Is Inconsistent with Temporal Integration of Random EPSPs. *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*. 1993;13(1):334-50. doi:10.1523/JNEUROSCI.13-01-00334.1993.
33. Churchland MM, Yu BM, Cunningham JP, Sugrue LP, Cohen MR, Corrado GS, et al. Stimulus Onset Quenches Neural Variability: A Widespread Cortical Phenomenon. *Nature Neuroscience*. 2010;13(3):369-78. doi:10.1038/nrn.2501.
34. Amarasingham A, Chen TL, Geman S, Harrison MT, Sheinberg DL. Spike Count Reliability and the Poisson Hypothesis. *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*. 2006;26(3):801-9. arXiv:16421300. doi:10.1523/JNEUROSCI.2948-05.2006.
35. Hu EJ, Shen Y, Wallis P, Allen-Zhu Z, Li Y, Wang S, et al.. LoRA: Low-Rank Adaptation of Large Language Models. arXiv; 2021. arXiv:2106.09685.
36. Drion G, Massotte L, Sepulchre R, Seutin V. How Modeling Can Reconcile Apparently Discrepant Experimental Results: The Case of Pacemaking in Dopaminergic Neurons. *PLOS Computational Biology*. 2011;7(5):e1002050. doi:10.1371/journal.pcbi.1002050.
37. Prinz AA, Bucher D, Marder E. Similar Network Activity from Disparate Circuit Parameters. *Nature Neuroscience*. 2004;7(12):1345-52. doi:10.1038/nn1352.
38. Hebb AO, Zhang JJ, Mahoor MH, Tsikos C, Matlack C, Chizeck HJ, et al. Creating the Feedback Loop: Closed-Loop Neurostimulation. *Neurosurgery Clinics of North America*. 2014;25(1):187-204. arXiv:24262909. doi:10.1016/j.nec.2013.08.006.
39. Mendolia L, Franci A. Designing, Tuning and Building Ultra-low-power Neuromodulable Mixed-feedback Silicon Neurons. In: 44th Benelux Meeting on Systems and Control; 2025. .
40. Qian K, Yu N, Tucker KR, Levitan ES, Canavier CC. Mathematical Analysis of Depolarization Block Mediated by Slow Inactivation of Fast Sodium Channels in

Midbrain Dopamine Neurons. *Journal of Neurophysiology*. 2014;112(11):2779-90.
doi:10.1152/jn.00578.2014.

41. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*. 2020;17:261-72. doi:10.1038/s41592-019-0686-2.
42. Shampine LF, Reichelt MW. The MATLAB ODE Suite. *SIAM Journal on Scientific Computing*. 1997;18(1):1-22. doi:10.1137/S1064827594276424.
43. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17. Red Hook, NY, USA: Curran Associates Inc.; 2017. p. 6000–6010.
44. Ba JL, Kiros JR, Hinton GE. Layer Normalization; 2016. arXiv:1607.06450.
45. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving Neural Networks by Preventing Co-adaptation of Feature Detectors; 2012. arXiv:1207.0580.
46. Hendrycks D, Gimpel K. Gaussian Error Linear Units (GELUs). arXiv; 2023. arXiv:1606.08415.
47. Huber PJ. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*. 1964;35(1):73-101. doi:10.1214/aoms/1177703732.
48. Bergstra J, Bengio Y. Random Search for Hyper-Parameter Optimization. *J Mach Learn Res*. 2012 Feb;13(null):281-305.
49. Loshchilov I, Hutter F. Decoupled Weight Decay Regularization. arXiv; 2019. doi:10.48550/arXiv.1711.05101.