Exploration of the AdEx parameter space through simulation and simulation-based-inference

Internship report

Jakob Huhle
Supervisor: Jakob Kaiser
Department of Physics and Astronomy
Ruprecht-Karls-University Heidelberg

June 2024

Abstract

This study investigates the parameters of the AdEx neuron model and their correlation with various neuronal firing patterns. We conducted simulations to explore the effects of parameter variations on different types of firing patterns. Results indicate that, despite changes in model parameters, firing patterns can still remain unchanged. This finding underscores the complex relationship between parameters and neuronal activity. In the second part of the study, simulation-based inference, specifically the SNPE algorithm, was employed to accurately predict the parameters underlying neuronal simulations. We successfully inferred both single and dual parameters, demonstrating the algorithm's efficacy in parameter estimation for the AdEx model. Consequently, these results might advocate for employing this algorithm for parameterizing AdEx neurons on neuromorphic hardware, such as the BrainScales2 system.

Contents

1 Introduction

Modeling neuronal firing patterns holds profound significance in neuroscience. These patterns are fundamental to understanding how neurons encode and process information in the brain. Accurate models not only illuminate the underlying mechanisms of neural activity but also pave the way for advancements in technology such as brain-machine interfaces and neuromorphic hardware.

A preferably simple model is needed to accurately simulate the firing patterns of neurons, closely approximating biological reality while remaining computationally efficient. Additionally, it should be versatile enough to describe the diverse behavior and types of neurons. Various models exist for simulating neuronal firing patterns, such as the Leaky Integrate-and-Fire model, Hodgkin-Huxley models, and more complex biophysical conductance-based models (Gerstner and Kistler 2002). Among these, the Adaptive Exponential (AdEx) Integrate-and-Fire model (Brette and Gerstner 2005) has shown particular promise due to its simplicity and its ability to closely match direct measurements in cortical neurons (Jolivet et al. 2008).

In the first part of this investigation, the versatility of the model is explored. Simulations are conducted for different parameter settings of the AdEx equations, and classifications are performed, enabling the mapping of certain parameter choices to specific resulting firing patterns. This helps in understanding how various parameters influence neuronal behavior. The investigation is based on the work of Naud et al. 2008.

In the second part, we adopt an inverse approach. Starting with the generated data, we aim to infer the underlying parameters of the AdEx equations that led to the creation of this data. To achieve this, we leverage simulation-based inference, specifically the Sequential Neural Posterior Estimation (SNPE) algorithm (Greenberg, Nonnenmacher, and Macke 2019). We explore how well these parameters can be inferred solely from the time series data of the voltage trace and the adaptation current.

This dual approach not only enhances our understanding of the parameter space within the AdEx model and its effects on neuronal firing patterns but also provides a robust framework for parameter inference, contributing significantly to both theoretical neuroscience and practical applications. For instance, it might advocate for the use of the SNPE algorithm in parameter optimization on neuromorphic hardware that utilizes the AdEx equations in order to correctly emulate voltage traces of biological neurons.

2 Background

2.1 AdEx

The Adaptive Exponential Integrate-and-Fire (AdEx) model (Brette and Gerstner 2005) is an advancement of the Leaky Integrate-and-Fire (LIF) model, introducing an exponential term to effectively handle fast input signals (Fourcaud-Trocmé et al. 2003). Furthermore, it includes an additional recovery variable that is crucial for modeling adaptation and resonance properties (Izhikevich 2003). The model outlines the behavior of the membrane potential V and the adaptation current w when a current I is injected. It comprises two coupled differential equations and a reset condition. The equations are defined as:

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_L(V - E_L) + g_L\Delta_T \exp\left(\frac{V - V_T}{\Delta_T}\right) + I - w, \tag{1}$$

$$\tau_w\frac{\mathrm{d}w}{\mathrm{d}t} = a(V - E_L) - w \tag{2}$$

with C being the total capacitance, g_L the total leak conductance, E_L the effective rest potential, Δ_T the threshold slope factor and V_T the effective threshold potential. These parameters are also referred to as the scaling parameters because they determine the scaling of the time axis, the stretching, and the offset of the state variables. We can rewrite C and g_L into the time scale $\tau_m = \frac{C}{g_L}$. The remaining parameters are also called the bifurcation parameters. Those are the conductance a, the time constant τ_w, the spike-triggered adaptation b, and the reset potential V_r. Adjusting these parameters leads to qualitative changes in the system's behavior, enabling different firing patterns to emerge.

When the membrane potential surpasses V_T, the exponential term induces a positive feedback mechanism, causing a rapid rise in the action potential. This upward surge is abruptly stopped at a reset threshold, set at 0 mV. Instead of a natural downswing, the action potential is immediately reset, as defined by the following condition:

If $V > 0\,\mathrm{mV}$, then

$$\begin{aligned} V &\to V_r \\ w &\to w_r = w + b \end{aligned} \tag{3}$$

2.2 AdEx Firing Patterns

The following summarizes the approach by Naud et al. 2008. To differentiate between various firing patterns, we first examine the phase space where the state variables V and w are plotted against each other (see Figure 1). Here, a nullcline represents the region in the phase space where a given variable remains constant. The intersection points of nullclines are referred to as fixed points, which can vary in their stability. In fact, a bifurcation[1] of the system occurs when the stability of a fixed point changes. This can happen in two ways: either through an Andronov-Hopf bifurcation or a saddle-node bifurcation (Gerstner and Kistler 2002). The current at which a fixed point loses stability is referred to as the rheobase. At that current, repetitive spiking occurs.

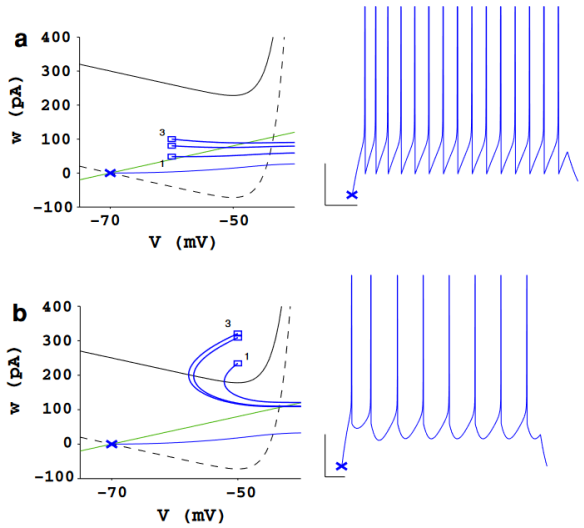

Figure 1: Phase space analysis of sharp (a) and broad (b) spike resets. The green line represents the w-nullcline, the black dashed line the V-nullcline, and the black solid line the V-nullcline in the presence of a stimulating current. Source: Naud et al. 2008.

It has been shown (Touboul and Brette 2008) that an Andronov-Hopf bifurcation occurs if $\frac{a}{g_L} > \frac{\tau_m}{\tau_w}$ at the rheobase:

$$I_{AH} = (g_L + a)[V_T - E_L - \Delta_T + \Delta_T ln(1 + \frac{\tau_m}{\tau_w})] + \Delta_T g_L \left(\frac{a}{g_L} - \frac{\tau_m}{\tau_w}\right). \quad (4)$$

Otherwise, a saddle-node bifurcation happens at the rheobase:

$$I_{SN} = (g_L + a)[V_T - E_L - \Delta_T + \Delta_T ln\left(1 + \frac{a}{g_L}\right)]. \quad (5)$$

The first step in distinguishing between different firing patterns can involve differentiating based on two types of spike resets: sharp and broad. Broad resets are

[1] a qualitative change in the system's dynamics.

characterized by their low curvature following the spike (see Figure 1). Those two types of spike resets have different trajectories in the phase plane. If the reset point after a spike is above the V-nullcline ($\frac{dV}{dt} < 0$), the voltage will decrease before it increases again for the next spike. Hence, a broad reset takes place. On the other hand, if the reset point is below the V-nullcline ($\frac{dV}{dt} > 0$), the voltage will immediately increase, resulting in a sharp reset.

With equation (1) we can rewrite this into a mathematical condition with the reset adaption current w_r:

$$w_r > -g_L(V_R - E_L) + g_L \Delta_T \exp\left(\frac{V_r - V_T}{\Delta_T}\right) + I. \tag{6}$$

If this condition is true, the reset is broad. If not, the reset is sharp.

Aside from the type of resets, we can also distinguish between various firing patterns based on changes in the inter-spike intervals (ISI). Figure 2 displays various types of firing patterns.

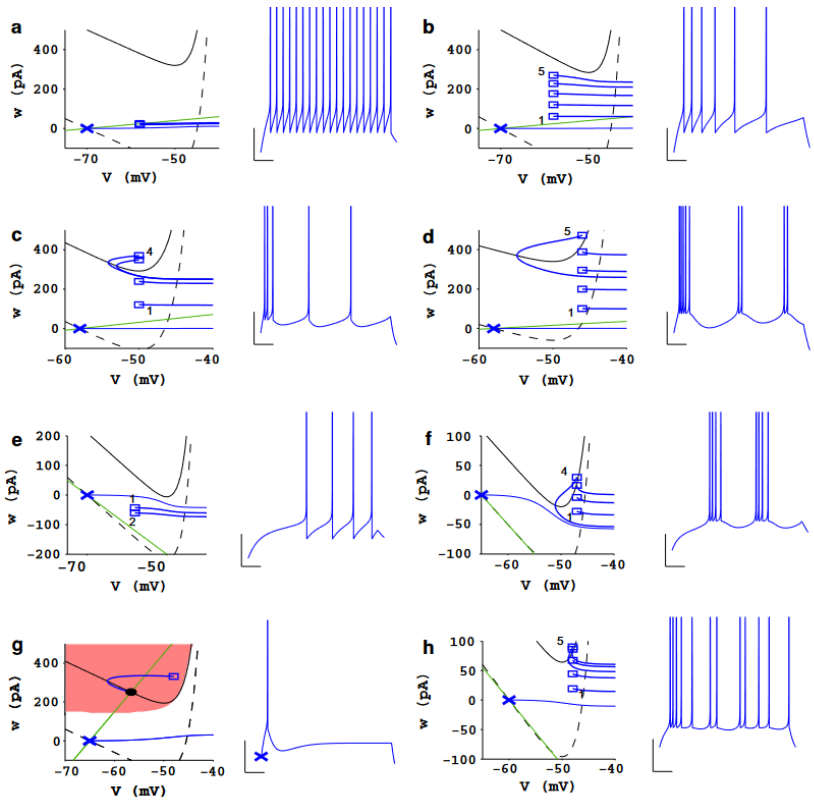

Figure 2: (a) **tonic spiking**: consistent ISIs, (b) **adapting spiking**: increasing ISIs, (c) **initial bursting**: initial emitted spikes with high frequency, (d) **regular bursting**: periodic alternation between sharp and broad resets, (e) **delayed accelerating spiking**, (f) **delayed regular bursting**, (g) **transient spiking**, (h) **irregular spiking**: ISIs change without periodicity; Source: Naud et al. 2008.

3 Simulation of the AdEx parameter space for generating diverse neuronal firing patterns

3.1 Methods

The following section aims to reconstruct the results obtained from the study by Naud et al. 2008 and therefore adopts the methodology outlined in that paper.

3.1.1 Simulation

As initial conditions for the model, we set $V(0) = E_L$ and $w(0) = 0$. Furthermore, the scaling parameters were fixed to constant values since the problem depends solely on the bifurcation parameters. Changing the scaling parameters would not alter the type of firing pattern, although it may affect the amount of current required to transition between different firing types. In alignment with the aforementioned paper, these values were set to $C = 100$ pF, $g_L = 10$ nS, $E_L = -70$ mV, $V_T = -50$ mV and $\Delta_T = 2$ mV. Hence, $\tau_w = \frac{C}{g_L} = 10$ ms.

Next, a constant step current was injected, with an amplitude twice that of the rheobase as described in formulas (4) and (5), inducing repetitive spiking in the neuron. Transient and delayed spiking patterns only occur near the rheobase current, and therefore were not investigated in that study. Throughout the entire simulation, the injected current remained constant. It is worth noting that altering the external current could also influence the type of firing pattern.

With all parameters configured, the AdEx equations were simulated for either 16 seconds or until 50 spikes occurred, whichever came first. A time resolution of 0.001 ms was utilized, determining the intervals at which the variables were updated.

The simulation was implemented in Python using both NEST[2] 3.0 and Brian[3] 2.7.0. Both are Python packages based on C++ for faster simulation.

3.1.2 Classification

The different firing patterns were then classified based on the voltage and adaption current traces of the simulated neurons.

Many neurons demonstrate spike-frequency adaptation, where the inter-spike intervals (ISI) increase over time with sustained stimulation. Classification into adapting and non-adapting firing patterns is based on the adaptation index, which is defined as

$$A = \frac{1}{N-k-1} \sum_{i=k}^{N} \frac{ISI_i - ISI_{i-1}}{ISI_i + ISI_{i-1}} \tag{7}$$

[2]https://github.com/nest/nest-simulator.
[3]https://github.com/brian-team/brian2.

with N being the total number of spikes (Naud et al. 2008). We set $k = 4$ to ignore any initial transient. In the case of tonic spiking, where all ISIs are equal throughout the pattern, A would be zero. A increases in the case of an adapting firing pattern and decreases in the case of an accelerating pattern. The adaptation index varies depending on the number of spikes considered. Therefore, we only considered the first 20 spikes for the calculation.

The following rules were applied to obtain the final results (Naud et al. 2008):

- *tonic spiking*: sharp resets or strictly broad resets and $-0.01 < A < 0.01$
- *adapting spiking*: sharp resets or strictly broad resets and $A > 0.01$
- *accelerating spiking*: sharp resets or strictly broad resets and $A < -0.01$
- *initial bursting*: ordered sequence going from sharp to broad resets
- *regular bursting*: alternation between broad and sharp spike resets such that the number of sharp resets between each broad reset is constant after the third broad reset
- *irregular spiking*: alternation between broad and sharp spike resets such that the number of sharp resets between each broad reset is not constant after the third broad reset

Whether a reset is a sharp or a broad one was determined with equation (6) with w_r being the value of the adaption current one timestep after the spike.

3.2 Results

Figure 3 presents our reconstruction of plot d from Figure 6 in Naud et al. 2008 (see Figure 16 in the appendix).

As we can see, spike patterns remain unchanged along certain alterations in the parameter space, highlighting the complex relationship between the firing patterns and the AdEx parameters. The shape of the initial bursting domain is related to the V-nullcline, as discussed in Naud et al. 2008.

In comparison with Figure 16, the overall structure of the plot was successfully reconstructed. Nevertheless, slight deviations are evident. Firstly, in the initial bursting (green) domain, a single pattern was classified as irregular (black). Secondly, the area of the adapting spiking (yellow) in the upper part of the plot is not consistently defined. Lastly, the structure of the irregular firing pattern differs from the original.

Upon closer investigation of the single irregular pattern (black point within the green area), we observe that a single reset was classified as sharp instead of broad. Examining w_r for this reset, we noticed it deviates only slightly from the right-hand side of equation (6), while the w_r values for the other resets in this pattern show significantly larger deviations. This is probably due to our simulation's resolution of

0.001 ms, which was not sufficiently high. As a comparison, the same simulation as in Figure 3 is displayed in Figure 17, but with a time resolution of 0.1 ms. As we can see, the results of the classification differ. The resolution determines the time intervals at which variables are updated. Had we chosen a higher resolution, all simulation variables, including w_r, would have obtained more precise values. w_r is defined as the value of the adaption current w directly after the spike. Hence, we obtained w_r by selecting the value of w one time step after the spike time. With much smaller time steps, w_r would be more accurate and thus most likely satisfy the condition in equation (6), correctly classifying the spike reset as broad instead of sharp.

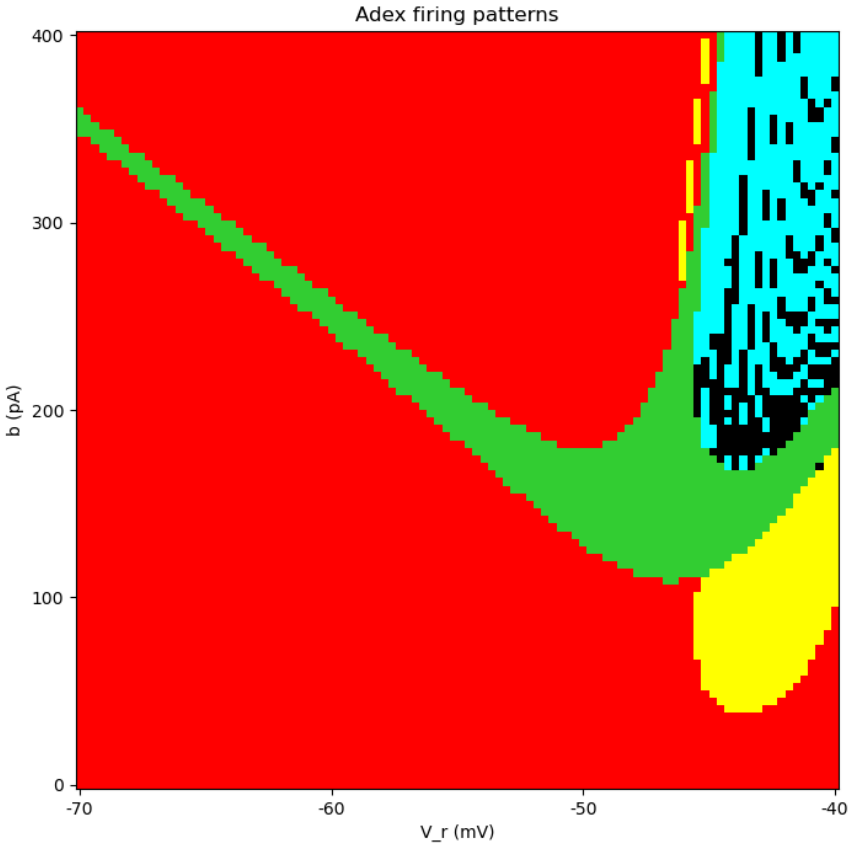

Figure 3: Parameter space exploration of b and V_r. We fixed $a = 0.001$ nS and $\tau_w = 5$ ms. The colors indicate different firing patterns: tonic spiking (red), adapting spiking (yellow), initial bursting (green), regular bursting (cyan), and irregular spiking (black). This is a replication of Figure 6d in Naud et al. 2008.

To investigate the deviations in the upper part of the plot where patterns were classified as tonic spiking (red) instead of adapting spiking (yellow), we plotted the adaptation index (see Figure 4). The distinction between these two patterns relies solely on the adaptation index A. If $A > 0.01$, the pattern is classified as adapting spiking and otherwise as tonic spiking (see Chapter 3.1.2). Surprisingly, the patterns classified as

tonic instead of adapting as in Figure 16 show not just small deviations from the critical value $A_c = 0.01$ but rather significant ones, with some even having an adaptation index close to 0. Increasing the simulation resolution also affects the adaptation index, as spike times become more accurate, influencing the inter-spike intervals (ISI) used in equation (7). However, it remains uncertain whether a higher resolution would sufficiently alter A to fulfill the condition $A > 0.01$ and thus reclassify all these patterns as adapting.

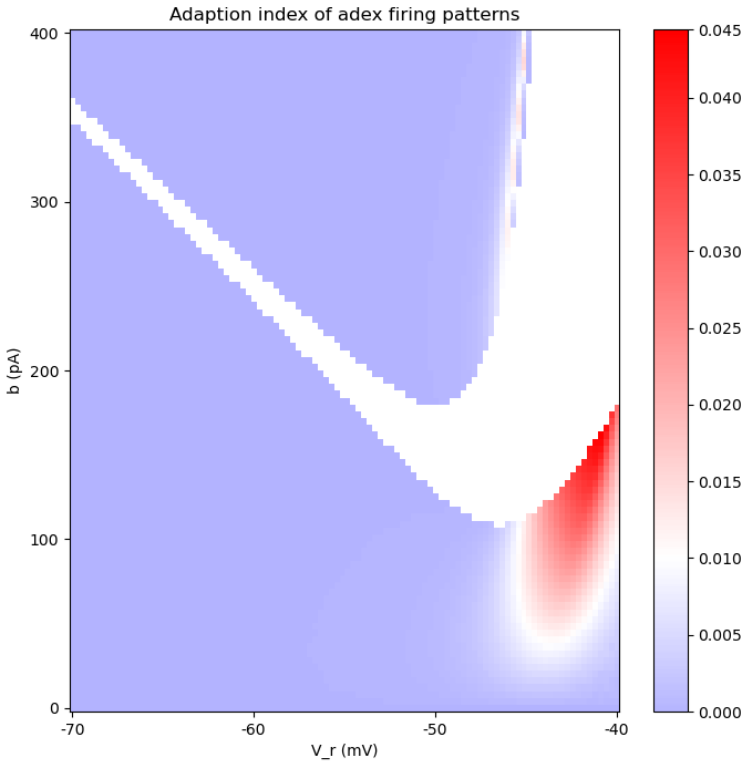

Figure 4: Adaptation index for tonic (blue) and adapting (red) spiking. The adaptation index is not calculated for the classification of initial bursting, regular bursting, and irregular spiking and is thus not included in this plot for these patterns. The arbitrary chosen critical value $A_c = 0.01$ serves as a threshold in classifying a firing pattern as adapting or tonic.

Lastly, we can see that the structure of the irregular firing patterns in Figure 3 is more sparse than in the one in Figure 16. As mentioned before, a higher resolution might be useful for correctly classifying resets. However, the choice of the simulation time step is more critical to the right of the plot in Figure 3 since firing occurs much faster in that part of the parameter space. This is illustrated in Figure 5 where the mean logarithmic inter-spike intervals (ISI's) are plotted.

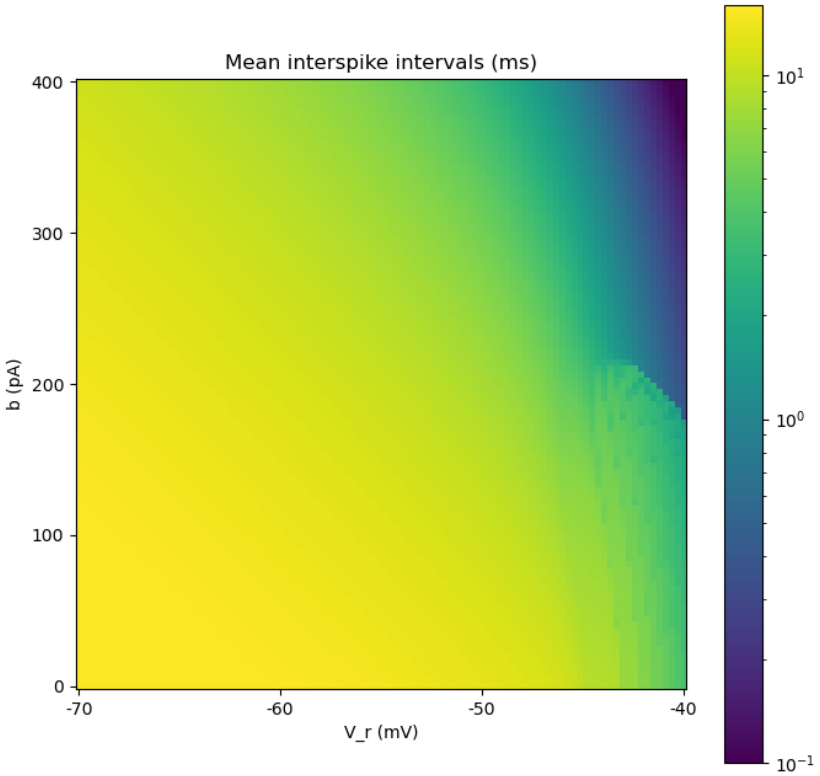

Figure 5: The mean inter-spike interval (ISI) indicates that faster spiking occurs towards the right side of the plot. This happens because the membrane potential V reaches the threshold V_T more quickly when it resets to a higher value V_r.

In summary, better results could be obtained with a higher simulation resolution. However, a higher resolution inevitably comes with increased computational runtime. A tradeoff must be found between runtime and classification accuracy. An example is shown in the appendix in Figure 17 for a smaller resolution of 0.1 ms. As we can see, the results in the classification differ. The respective voltage traces for the different resolutions are displayed in Figure 18. Nevertheless, there are still differences from plot d in Figure 16 that might not be fully corrected with a higher resolution. It is also possible that we used a slightly different classification scheme since not every detail was provided in Naud et al. 2008. For instance, we do not know the chosen simulation time step or the exact grid points for b and V_r.

4 Simulation-based inference of AdEx parameters

4.1 SNPE Algorithm

Mechanistic models are pivotal for understanding and predicting the behavior of complex systems under various circumstances. These models, grounded in the underlying physics or biology of the systems, allow researchers to simulate how systems evolve with respect to their parameters. However, one of the key challenges in leveraging these models is the inverse problem: determining the set of parameters θ that are able to reproduce the observed data $\hat{x}$ when plugged into the model. The simulation-based inference algorithm tries to approximate the posterior distribution $p(\theta|\hat{x})$, which represents the probability distribution of the parameters θ that generated the observed data. Generally, Bayesian inference

$$p(\theta|\hat{x}) = \frac{p(\hat{x}|\theta)p(\theta)}{\int dz p(\hat{x}|\theta')p(\theta')} \tag{8}$$

cannot be applied directly since the likelihood $p(\hat{x}|\theta)$ is unknown or too expensive to calculate. In most cases, it would be too expensive to compute the likelihood $p(\hat{x}|\theta) = \int dz p(\hat{x}, z|\theta)$ with all possible internal latent variables z of the model. Thus, we would like to approximate the posterior directly without knowing the likelihood (Cranmer, Brehmer, and Louppe 2020).

Using the Sequential Neural Posterior Estimation (SNPE) by Greenberg, Nonnenmacher, and Macke 2019, we can approximate the posterior directly. Therefore, we only need the mechanistic model from which we can generate observations x, a predefined prior $p(\theta)$ for the parameters in question, and the observed data $\hat{x}$. The algorithm then simulates the model for a set of randomly drawn parameters from the prior. Subsequently, a neural density estimator (NDE) is trained, capturing the relationship between the parameters and the data. If we are only interested in a specific observation, we can perform the training sequentially. Here, we use the estimated posterior from the previous rounds as the new prior. However, we need to retrain our model for a new observation $\hat{x'}$. Figure 6 visualizes the approach of SNPE.

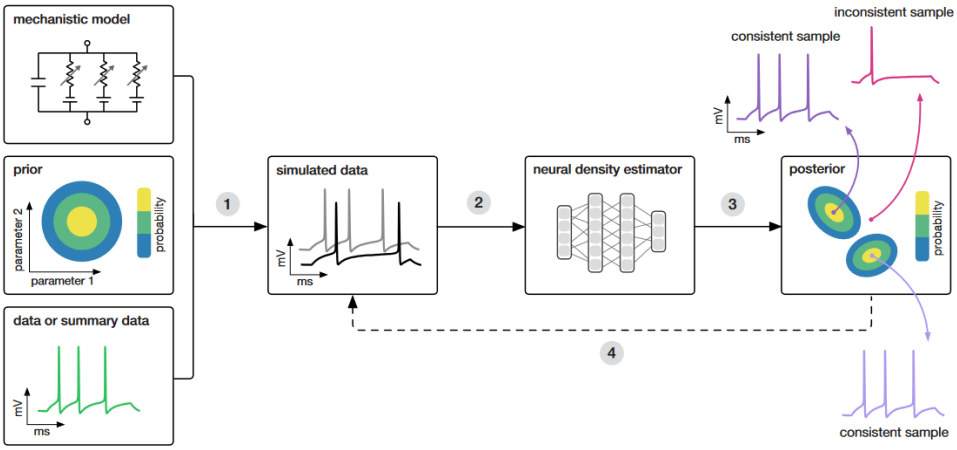

Figure 6: The SNPE algorithm. A mechanistic model, a prior for the model parameters, and the (embedded) data are the inputs for the algorithm. SNPE then (1) performs simulations of randomly drawn samples, and (2) trains a deep density estimation to learn the correlation of the data and the model's parameters network. Afterward, (3) this network can derive the underlying parameters of the presented data within the range of the prior. (4) In the case of multi-round inference, an initial estimate of the posterior can be used as the new prior in subsequent simulation and training rounds. Source: Goncalves et al. 2020.

4.2 Embedding of the raw data

When dealing with high-dimensional observed data, dimensionality reduction techniques are often applied to encode the data into a lower-dimensional feature space to save computational costs during training of the NDE (Cranmer, Brehmer, and Louppe 2020). This step is pivotal to strike the right balance between reducing information redundancy while retaining relevant features essential for effective parameter inference. It ensures that the encoded features capture the essential aspects of the data that are most useful for density estimation in simulation-based inference. In this study, we utilized the discrete wavelet transform (DWT) for the data encoding.

4.2.1 Discrete Wavelet transformation

The Wavelet Transformation (Weeks and Bayoumi 2002) is a mathematical tool used to analyze signals with time-varying frequencies. Fields of applications include, among others, data compression (B'charri et al. 2016; Manikandan and Dandapat 2014), noise reduction (Alfaouri and Daqrouq 2008), and feature extraction (Zhang et al. 2006; Y. Wang, Yan, and Q. Wang 2016) of a signal. The Wavelet Transform is similar to the Fourier Transform. The Fourier Transform represents a signal through a linear combination of sine waves, transforming the signal from the time domain to the frequency domain. However, it has a limitation: it does not indicate at which specific times these frequencies occur. In time series data, frequencies often vary over time. In contrast, the Wavelet Transform uses basis functions called "wavelets", which are localized in both frequency and time, to transform the data.

These wavelets have finite energy, meaning they are concentrated in both time and frequency and are integrable. Additionally, wavelets have zero mean. Various wavelet families exist (see Figure 7 for examples). Within each family, subfamilies differ in the number of coefficients, impacting the wavelet's approximation accuracy and smoothness (i.e. the Daubechi wavelet "db2" has a smoother shape than "db1").

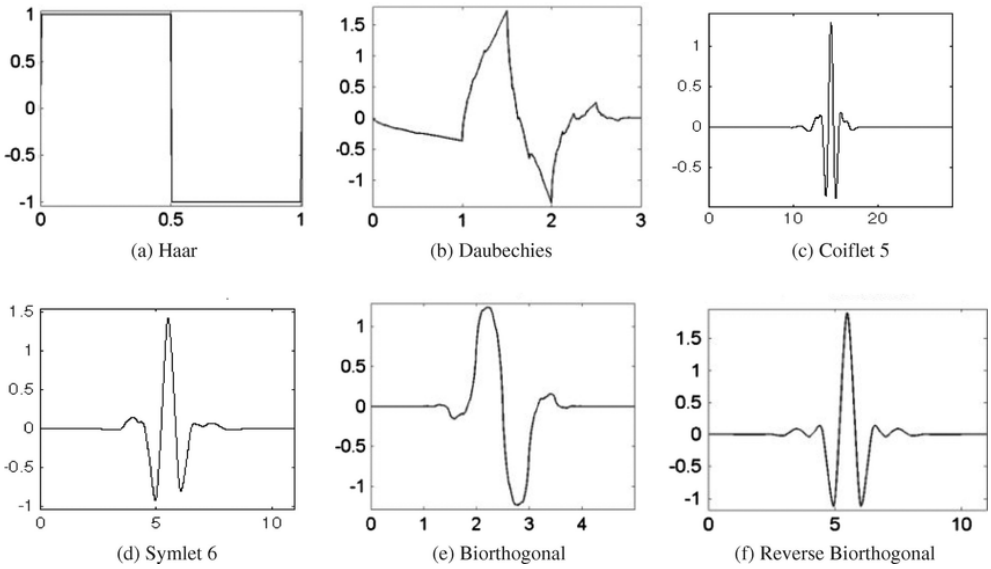

Figure 7: Examples of Wavelet families. Source: Figure 3 of Zope-Chaudhari, Venkatachalam, and Buddhiraju 2015.

A "mother wavelet" $\psi(x)$ can be transformed by scaling and translating it. The general form of a daughter wavelet $\psi_{a,b}(x)$ is:

$$\psi_{a,b}(x) = \frac{1}{\sqrt{a}}\psi\left(\frac{x-b}{a}\right) \tag{9}$$

where a is the scaling factor and $\frac{b}{a}$ is the translation parameter. A higher scale factor a corresponds to a lower frequency, whereas a lower scale factor corresponds to a higher frequency. With these wavelets, signals can be analyzed across multiple scales, making Wavelet Transformations valuable for studying signals with time-varying characteristics, encompassing high-frequency transients to slowly varying trends and everything in between.

Wavelet transformation consists of performing a convolution operation on a given time series data with a specified wavelet (Chaovalit et al. 2011). The transform of a signal $f(t)$, resulting in the wavelet coefficients $T_{a,b}$, can be written as:

$$T_{a,b}(f) = \frac{1}{\sqrt{a}}\int dt f(t)\psi\left(\frac{x-b}{a}\right) \tag{10}$$

with a and b as discrete values. The Discrete Wavelet Transform (DWT) uses different wavelets, each with different scales, and multiplies them with the signal at different locations in time. The transformation is implemented as a filter bank (see Figure 8).

Low-pass and high-pass filters are employed to divide the data into low- and high-frequency components. At each level, the low-frequency part is further decomposed, yielding different coefficients: approximation coefficients reflect the output of the low-pass filter in DWT, while detail coefficients correspond to the high-pass filter output. This process separates the signal into multiple components, each representing a subband of the original signal. With each successive stage, the number of samples is halved. The Wavelet Transform output yields the same number of coefficients as there are data points in the signal. Using the inverse Wavelet transform, the original signal can be perfectly reconstructed solely from these coefficients.

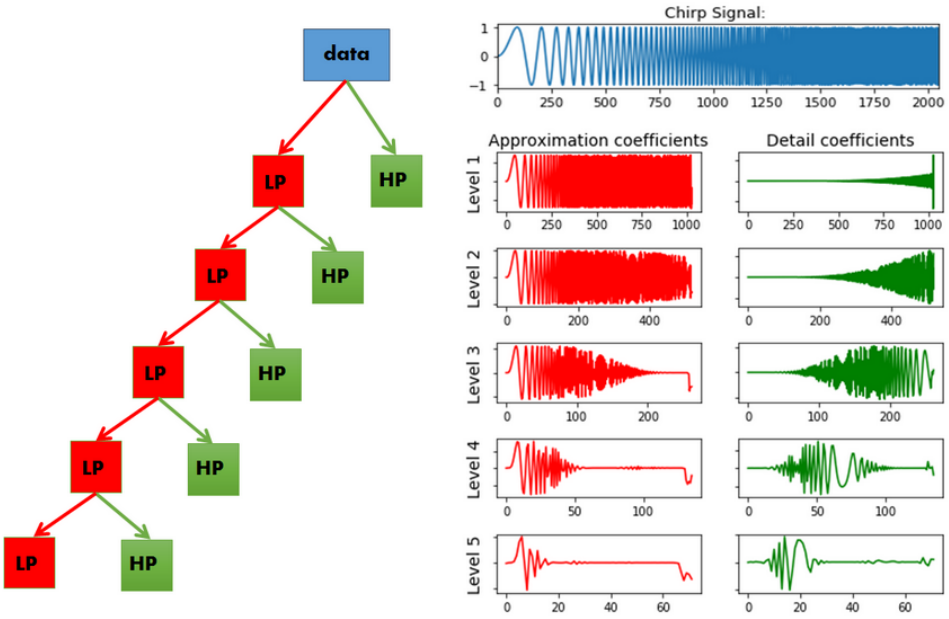

Figure 8: DWT with a filter bank. Source: https://ataspinar.com/2018/12/21/a-guide-for-using-the-wavelet-transform-in-machine-learning/

So far, the Wavelet transformation is not yet a dimensionality reduction technique because there are as many coefficients as there are data points in the original data. However, by separating the transformation into approximation and detail coefficients, we can selectively retain the most significant coefficients that effectively capture the overall structure of the data and discard the less important ones. This selective retention enables a reduction in the number of retained coefficients, thereby achieving a form of dimensionality reduction while preserving the essential information of the signal.

4.3 Methods

Our simulator, from which we generate observations x, consisted of the numerical integration over the AdEx equations (formulas (1) and (2)). The resulting data consists of a voltage trace and an adaptation current trace. We simulated the neuron for 5.024[4] seconds with a time resolution of 0.1 ms, resulting in data of dimensionality dim(x)=2×50240 data points. For the data embedding, we employed both one-dimensional (independently transforming each trace) and two-dimensional DWT using the Daubechies wavelet "db2". The two-dimensional DWT is simply one-dimensional DWT in the horizontal and vertical directions of the data (Weeks and Bayoumi 2002). The aim was that this would capture essential correlations between the voltage and the adaptation current trace. Different approaches have been explored to achieve dimensionality reduction using the Wavelet transform (Srivastava and Purwar 2017, Qu et al. 2003, Popivanov and Mille 2002). The more coefficients are discarded, the greater the data reduction.

To maintain a consistent resulting data size, we chose to use only the approximation coefficients at decomposition level 7 as our encoded data. Higher levels did not adequately capture the fundamental structure of the data, while the data size would be doubled with each lower level. This compression resulted in a reduction to 2×395 data points[5].

As we can see in Figure 9, the detailed coefficients contribute in regions with high frequency while the approximation coefficients contribute to low frequency components. Since spiking occurs rapidly in this example, the detailed coefficients closely resemble the structure of the original trace. Conversely, reconstruction using only the approximation coefficients is notably poor. Nonetheless, as demonstrated in Chapter 4.4, the NDE satisfactorily infers the parameters. Thus, the approximation coefficients still sufficiently represent the data for our purposes.

Given that our data does not conform to a power-of-two length and considering the substantial reduction to level 7, it is expected that boundary effects may occur during the transformation. However, these effects did not significantly affect our inference results. We utilized the PyWavelets[6] package for implementing the DWT. The resulting data was then flattened and fed into a fully connected neural network (FCNN) with layer dimensions 600, 400, 300, 100, and 30 to further refine it to its most essential features. The FCNN was simultaneously trained with the NDE to achieve an optimal embedding for inference.

The NDE was a Neural Spline Flow. In each round of the inference, 1000 simulations were performed. Initially, we inferred only a single parameter, specifically b from the AdEx model. Subsequently, we performed simultaneous inference on two parameters: b and the reset potential V_r. For these inferences, we defined uniform priors, ranging from 100 to 400 pA for b and from -70 to -40 mV for V_r. We halted the inference at a certain point when drawing posterior samples became too slow[7]. To conduct

[4] After a delay of 0.024 seconds, a constant step current was injected.
[5] Obtaining a compression ratio of $size_{new}$ / $size_{old} \approx 0.008$.
[6] https://github.com/PyWavelets/pywt.
[7] This happened when most of the drawn samples lie outside of the support of the prior and need

the inference, we utilized the algorithm developed by Greenberg, Nonnenmacher, and Macke 2019, which is implemented in the Python package sbi[8] (Tejero-Canteroe et al. 2020).

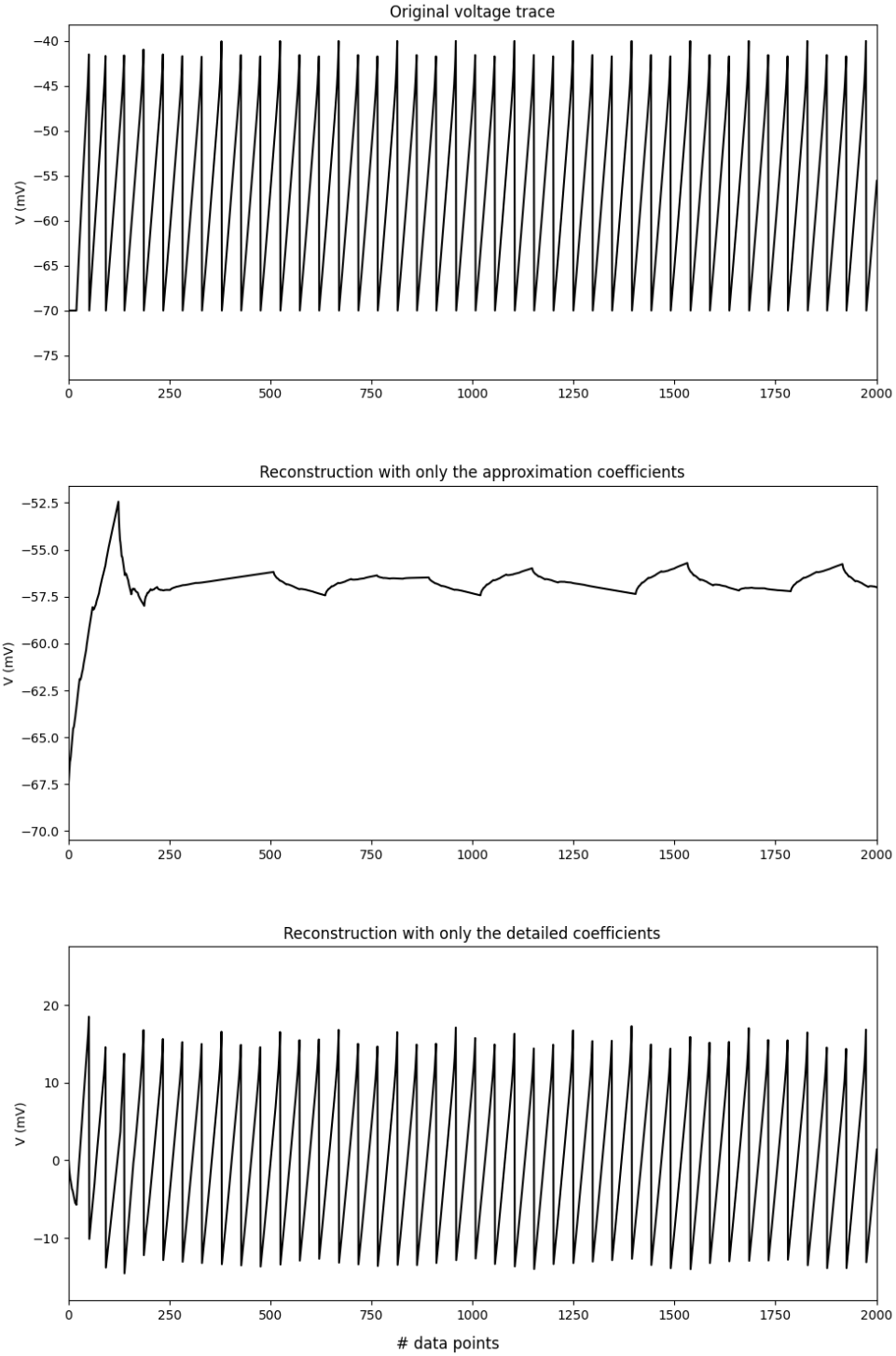

Figure 9: DWT on a voltage trace with a decomposition level of seven.

to be discarded.

[8] https://github.com/mackelab/sbi.

4.4 Results

First, we inferred only one parameter, namely b. By performing dimensionality reduction with the one-dimensional DWT, we were able to approximate the posterior quite well with three inference rounds (see Figure 10).

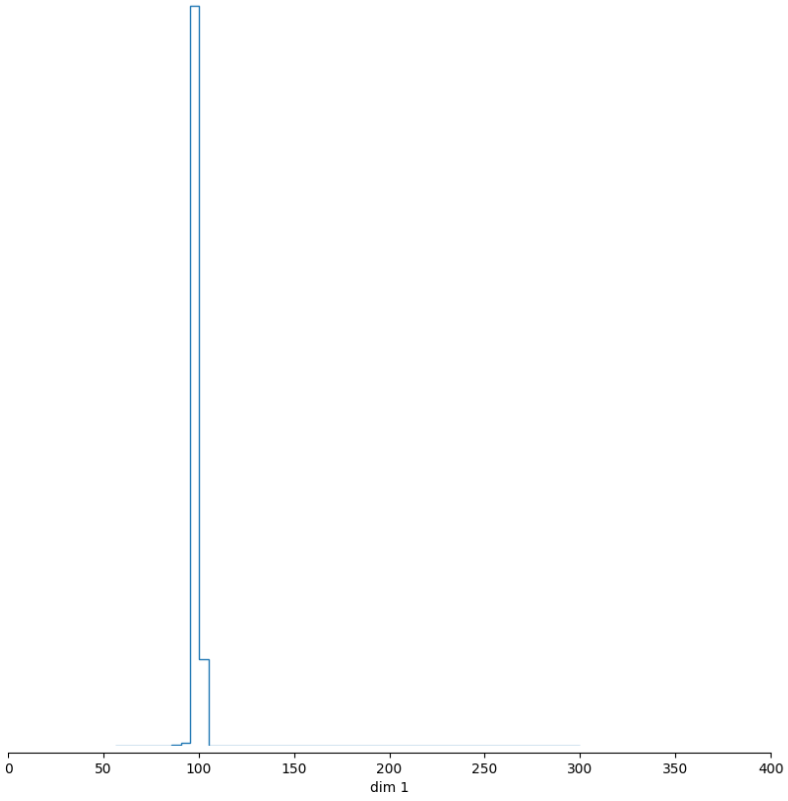

Figure 10: Normalized approximated posterior with one-dimensional DWT embedding using the "db2" wavelet. Three rounds of inference with 1000 simulations each were performed. $b_{true} = 100\,\text{pA}$ was arbitrarily chosen.

We chose $b_{true} = 100$ pA arbitrary as the true value that generated the observation $\hat{x}$. However, when we used two-dimensional DWT for the embedding with four inference rounds, our posterior resulted in two peaks (see Figure 11).

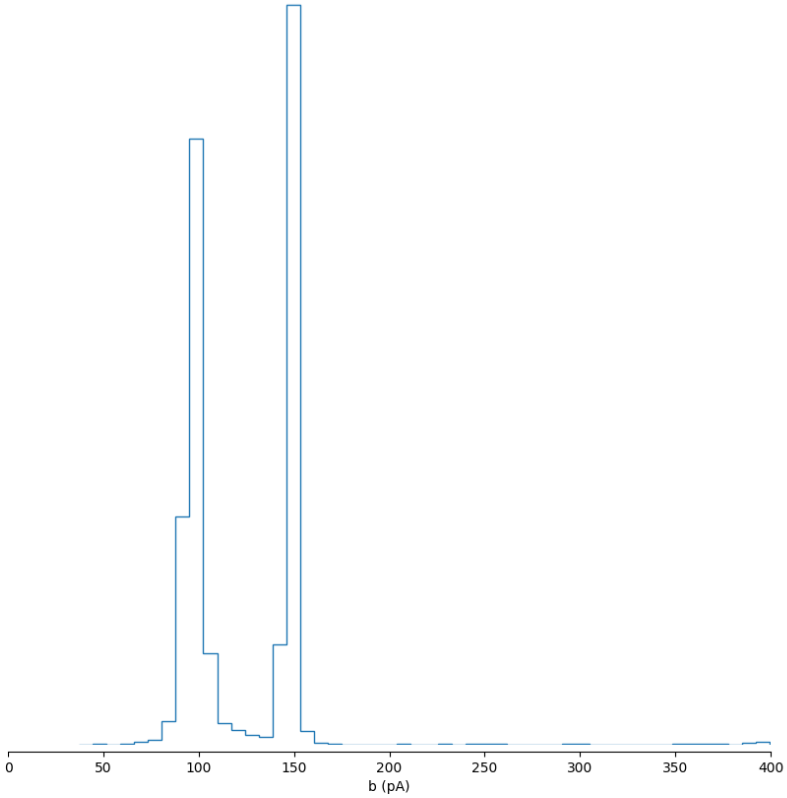

Figure 11: Normalized approximated posterior with two-dimensional DWT embedding using the "db2" wavelet. Four rounds of inference with 1000 simulations each were performed. $b_{true} = 100\,\mathrm{pA}$ was arbitrarily chosen.

Since the two-dimensional DWT is simply the application of the one-dimensional DWT on both the horizontal and vertical axes, it seems evident that this approach might not work well. This is because we only have two points in one direction: one for the voltage trace and one for the adaptation current trace. To investigate that posterior, we randomly drew samples from the left and right peaks of the posterior and plotted their simulations against the true voltage trace, generated with b_{true} (see Figures 12 and 13).

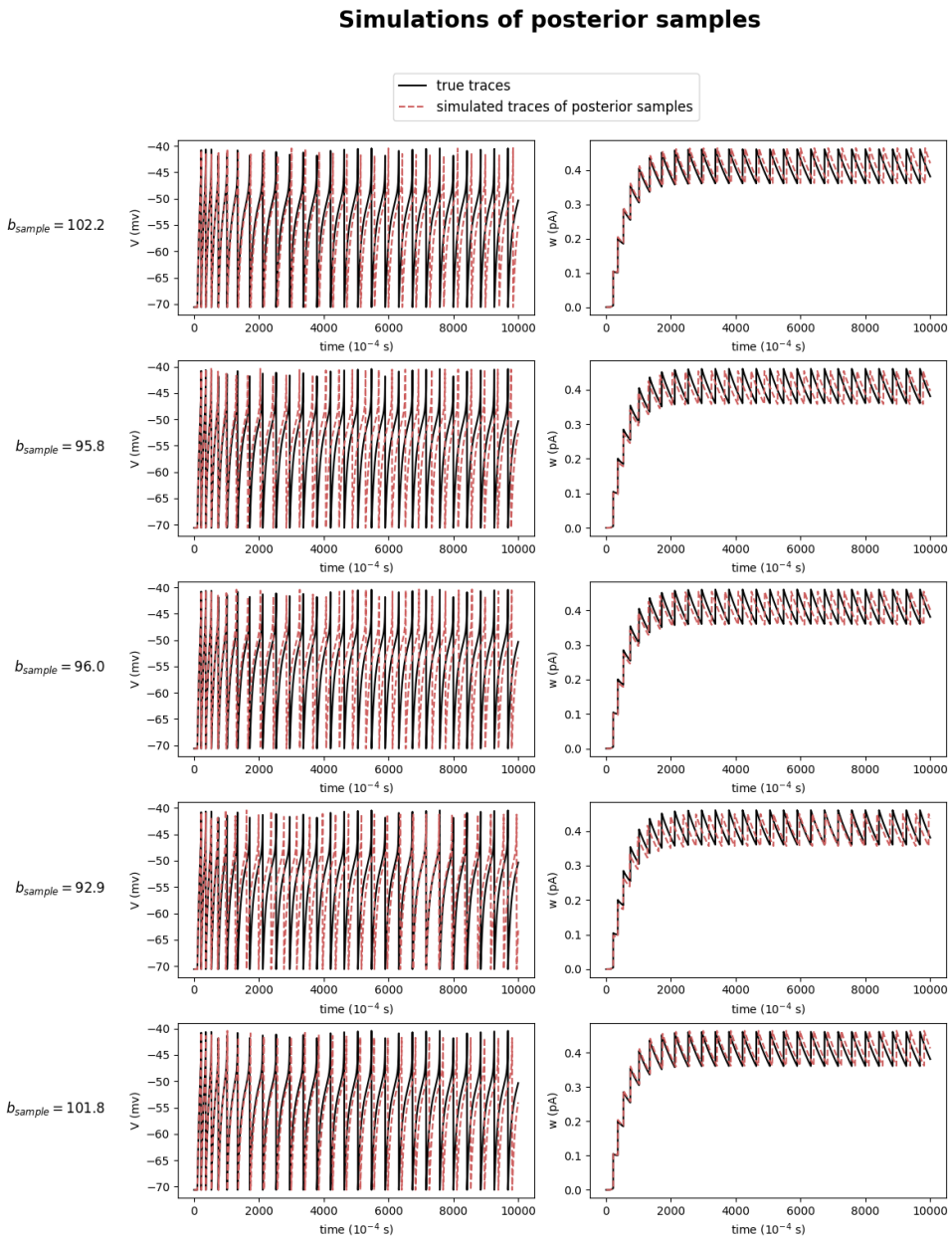

Figure 12: True voltage trace (with $b_{true} = 100$) vs. simulated traces from samples drawn from the left peak of the approximated posterior in Figure 11.

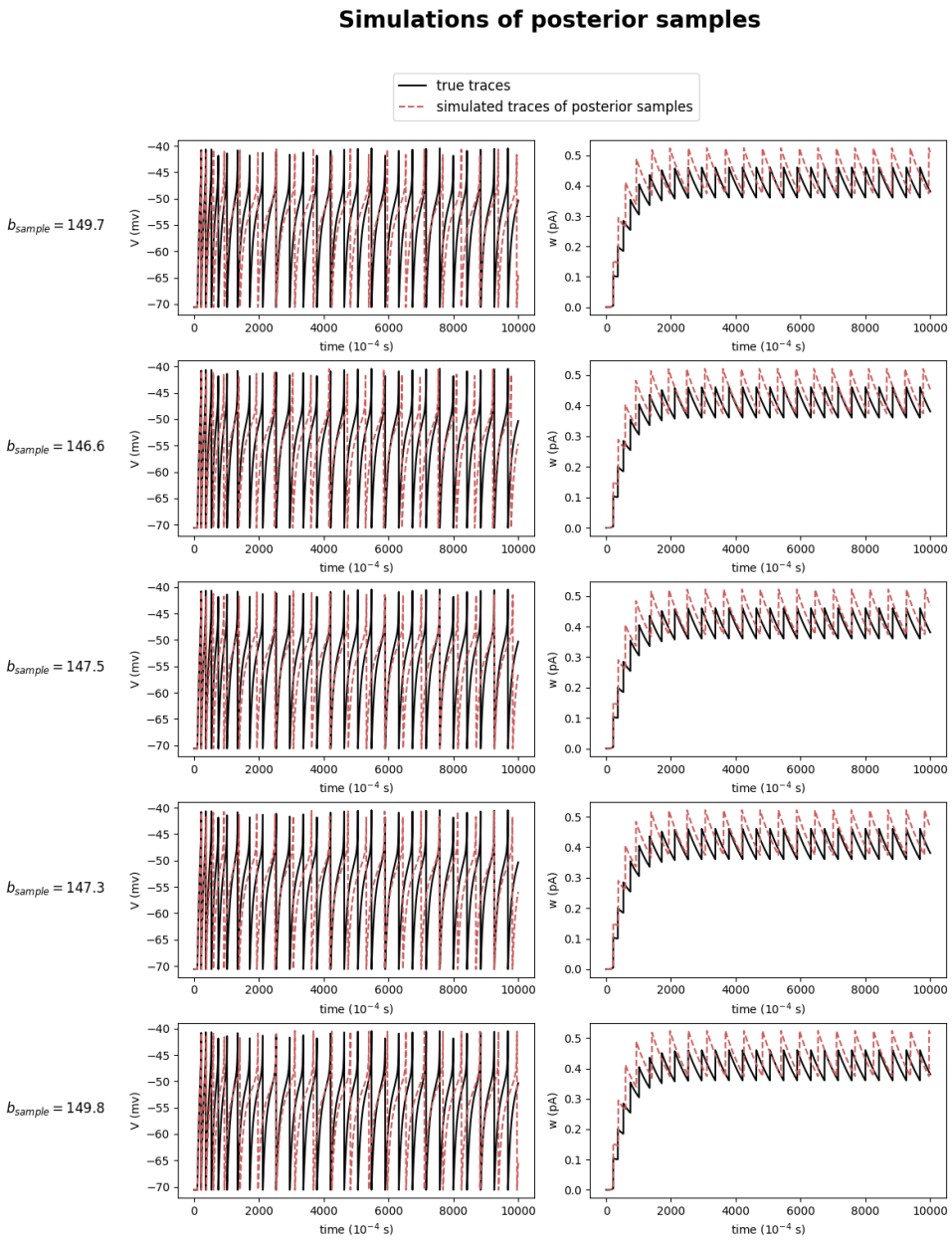

Figure 13: True voltage trace (with $b_{true} = 100$) vs. simulated traces from samples drawn from the right peak of the approximated posterior in Figure 11.

As we can see in Figure 12, the closer the sample values of b are to 100 pA, the better the agreement between the true and simulated traces of the sample becomes. The differences in b from the true value accumulate over time because b affects the system at every spike time (see formula (3)), resulting in a greater inconsistency of the agreement as more time passes.

Evident in Figure 13, the agreement is quite poor, with the simulated traces of the adaption current systematically shifted too high. This is to be expected for a higher value of b (see formula (3)). Hence, the two-dimensional DWT has failed to capture

these crucial features. Consequently, our second posterior, with its two peaks, is significantly more inaccurate than the first. This emphasizes the importance of selecting the correct embedding technique.

Next, we inferred two variables, b and V_r. With seven rounds of inference and using the one-dimensional DWT as an embedding, we obtained the posterior in Figure 14.

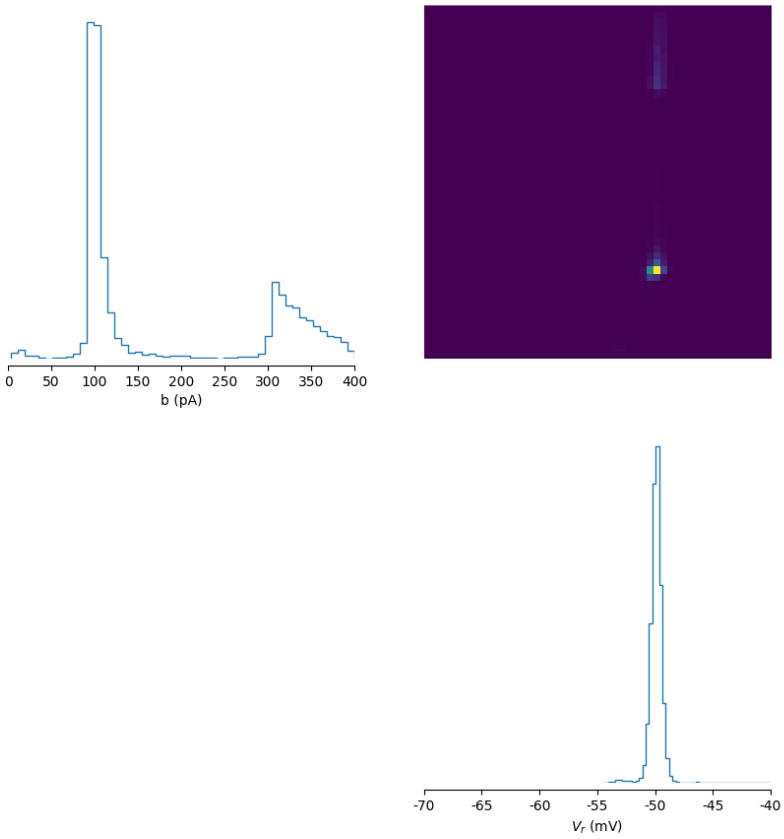

Figure 14: Normalized approximated posterior with two inferred variables. The "db2" wavelet was chosen for the one-dimensional DWT. Seven rounds of inference with 1000 simulations each were performed. $b_{true} = 100\,\text{pA}$ and $V_{r,true} = -50\,\text{mV}$ were arbitrarily chosen. Violet resembles (close to) zero and yellow high probability for the parameters to generate the given observation.

As we can see, the SNPE algorithm accurately predicted both variables, but there is an additional small peak in the marginal of the b parameter. Here, $V_{r,true} = -50$ mV was the true value for the reset potential. As a comparison, the posterior in Figure 15 was obtained using the same number of inference rounds, but with the data being embedded using the Haar-wavelet in the one-dimensional DWT. It is evident that the resulting posterior exhibits slightly better results compared to the previous result. In summary, the choice of wavelet for the DWT also has an impact on the resulting posterior.

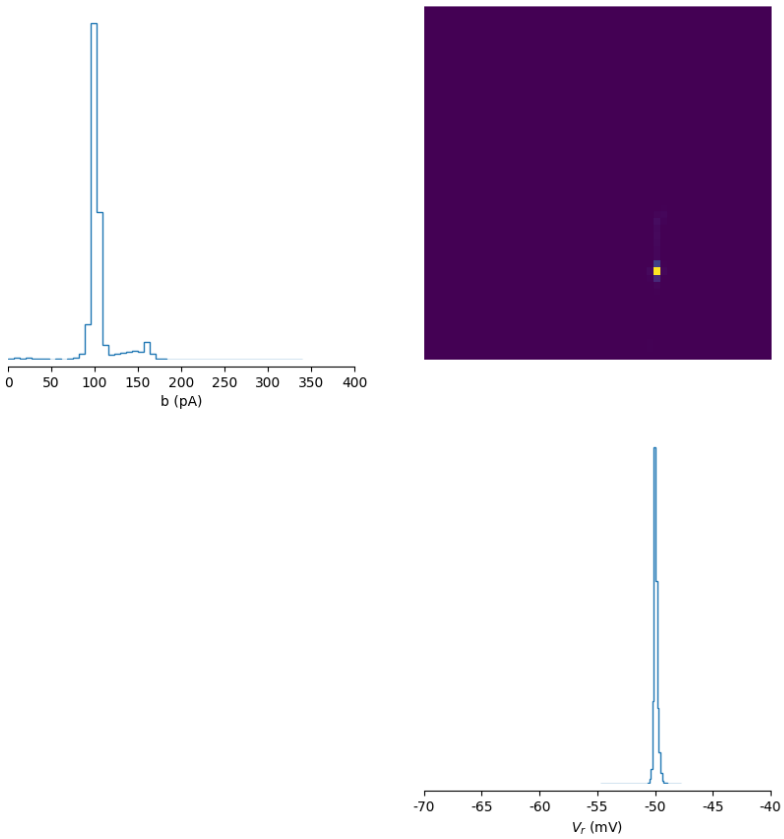

Figure 15: Normalized approximated posterior with two inferred variables. The "haar" wavelet was chosen for the one-dimensional DWT. Seven rounds of inference with 1000 simulations each were performed. $b_{true} = 100\,\mathrm{pA}$ and $V_{r,true} = -50\,\mathrm{mV}$ were arbitrarily chosen. Violet resembles (close to) zero and yellow high probability for the parameters to generate the given observation.

5 Summary

This study explored the relationship between parameters and voltage and adaptation current traces of the AdEx model. Through simulations, we investigated the effects of parameter variations on the resulting firing patterns of the model. We successfully replicated the results reported in the paper by Naud et al. 2008. Plots were generated to identify regions in the parameter space where firing patterns remained consistent.

Additionally, we employed the SNPE algorithm to infer parameters. Data embedding was conducted using the Discrete Wavelet Transform (DWT). We found that a two-dimensional wavelet transform failed to capture essential features, resulting in an inaccurate posterior distribution. Conversely, two one-dimensional DWTs provided adequate embedding for inferring the parameters of interest.

Overall, this study highlights the complex relationship between the parameters of the AdEx differential equations and the resulting firing patterns. Furthermore, it underscores the effectiveness of SNPE in parameter inference. The choice of data embedding techniques, such as one-dimensional DWTs, significantly influences the accuracy of parameter estimation.

6 Future work

Our future objective is to apply the SNPE algorithm for parameterization on analog neuromorphic hardware. The goal is to identify suitable hardware parameters to accurately emulate biological processes. To achieve this, we will explore new methods of data encoding, including deep convolutional autoencoders and transformer architectures to effectively reduce the dimensionality of the data. Furthermore, since biological neuron data often contains noise, it is crucial to thoroughly test the robustness and noise resistance of our embedding and inference methods.

7 Appendix

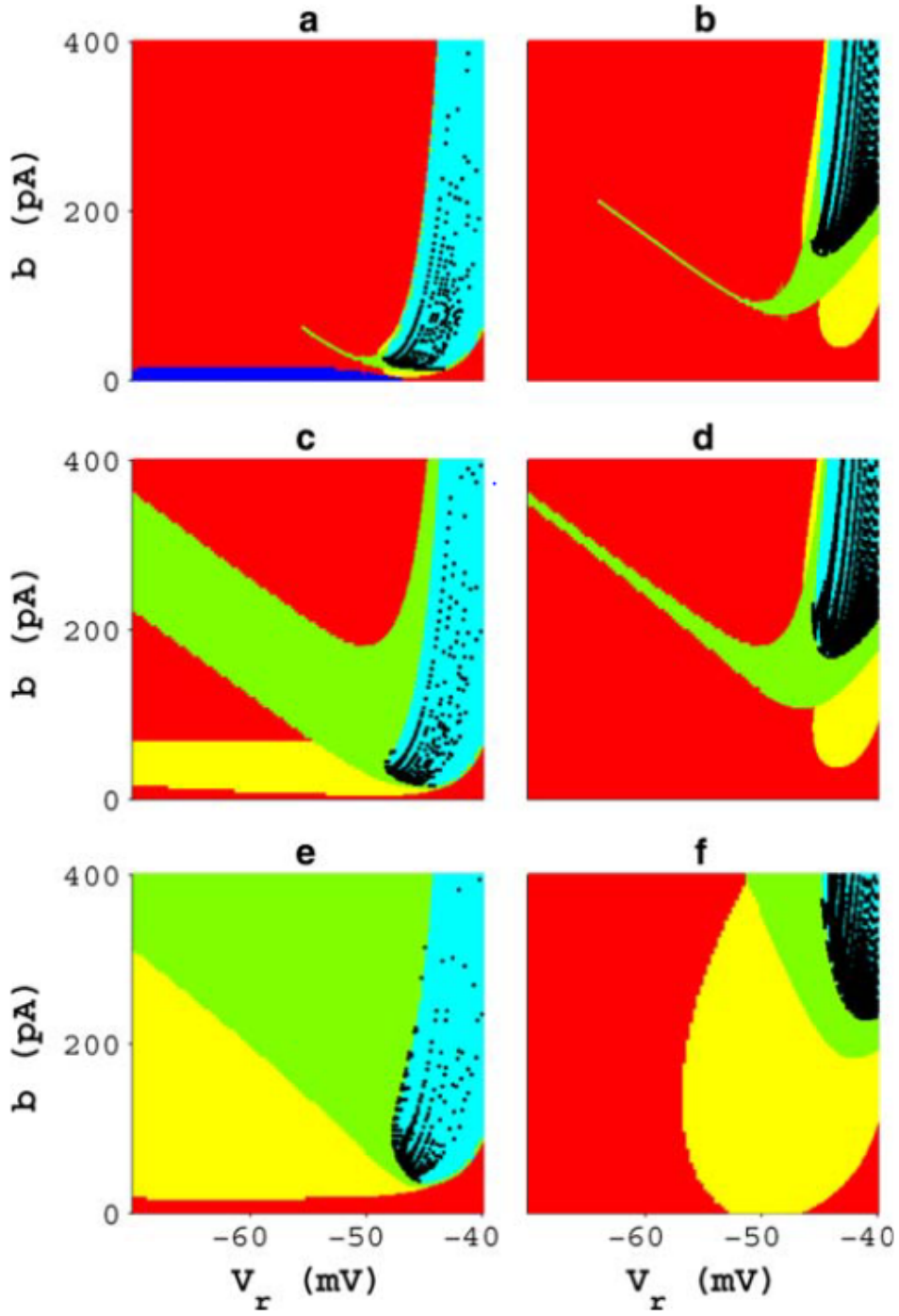

Figure 16: Figure 6 from Naud et al. 2008. Tonic spiking in red, adapting in yellow, initial bursting in green, regular bursting in cyan, irregular spiking in black, and accelerating in blue. The four-dimensional parameter space was reduced to six relevant planes: (a) $a = -5$ nS and $\tau_w = 100$ ms, (b) $a = -5$ nS and $\tau_w = 5$ ms, (c) $a = 0.001$ nS and $\tau_w = 100$ ms, (d) $a = -5$ nS and $\tau_w = 5$ ms, (e) $a = 30$ nS and $\tau_w = 100$ ms, (f) $a = 30$ nS and $\tau_w = 5$ ms.

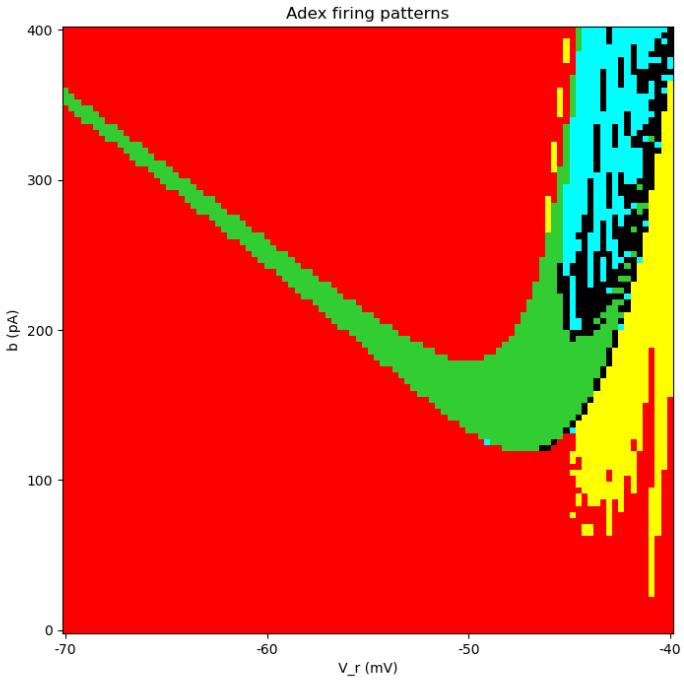

Figure 17: Same plot as in Figure 3 but with a time resolution of 0.1 ms instead of 0.001 ms. The results in the pattern classification are less accurate. We fixed $a = 0.001$ nS and $\tau_w = 5$ ms. The colors indicate different firing patterns: tonic spiking (red), adapting spiking (yellow), initial bursting (green), regular bursting (cyan), and irregular spiking (black).

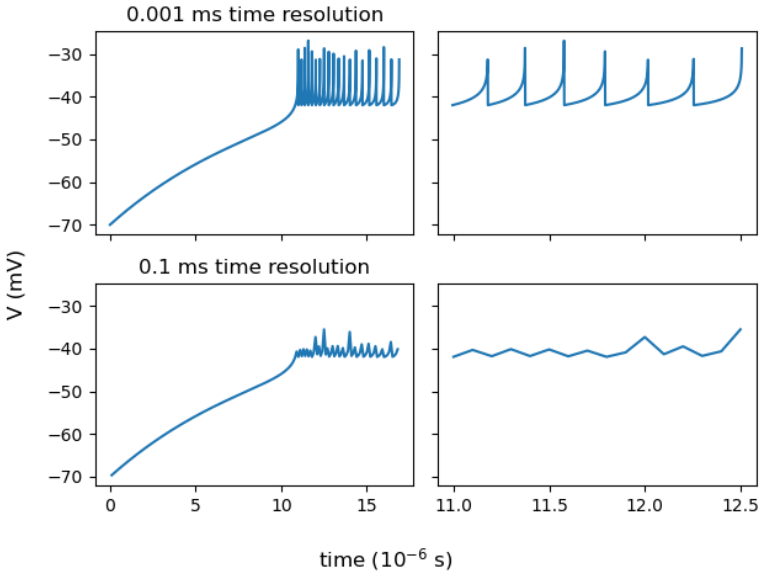

Figure 18: Simulated Voltage traces for a time resolution of 0.001 ms and 0.1 ms. The voltage traces differ even though the same AdEx parameters were used.

References

Alfaouri, Mikhled and Khaled Daqrouq (2008). "ECG Signal Denoising By Wavelet Transform Thresholding". In: *American Journal of Applied Sciences*.

B'charri, Oussama El et al. (2016). "The ECG Signal Compression Using an Efficient Algorithm Based on the DWT". In: *International Journal of Advanced Computer Science and Applications*. DOI: 10.14569/IJACSA.2016.070325.

Brette, Romain and Wulfram Gerstner (2005). "Adaptive Exponential Integrate-and-Fire Model as an Effective Description of Neuronal Activity". In: *J Neurophysiol*. DOI: 10.1152/jn.00686.2005.

Chaovalit, Pimwadee et al. (2011). "Discrete Wavelet Transform-Based Time Series Analysis and Mining". In: *ACM Comput. Surv.* DOI: 10.1145/1883612.1883613.

Cranmer, Kyle, Johann Brehmer, and Filles Louppe (2020). "The frontier of simulation-based inference". In: *Proceedings for the Sackler Colloquia at the US National Academy of Sciences*. DOI: 10.48550/arXiv.1911.01429.

Fourcaud-Trocmé, Nicolas et al. (2003). "How Spike Generation Mechanisms Determine the Neuronal Response to Fluctuating Inputs". In: *The Journal of Neuroscience*. DOI: 10.1523/JNEUROSCI.23-37-11628.2003.

Gerstner, Wulfram and Werner Kistler (2002). *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press. ISBN: 0 521 89079 9. URL: https://neuronaldynamics.epfl.ch/online/index.html.

Goncalves, Pedro J et al. (2020). "Training deep neural density estimators to identify mechanistic models of neural dynamics". In: *eLife*. DOI: 10.7554/eLife.56261.

Greenberg, David S., Marcel Nonnenmacher, and Jakob H. Macke (2019). "Automatic Posterior Transformation for Likelihood-free Inference". In: *Proceedings of the 36th International Conference on Machine Learning*. DOI: 10.48550/arXiv.1905.07488.

Izhikevich, Eugene M. (2003). "Simple Model of Spiking Neurons". In: *IEEE Transactions on Neuronal Networks*. DOI: 10.1109/TNN.2003.820440.

Jolivet, Renaud et al. (2008). "A benchmark test for a quantitative assessment of simple neuron models". In: *J Neurosci Methods*. DOI: 10.1016/j.jneumeth.2007.11.006.

Manikandan, M. Sabarimalai and S. Dandapat (2014). "Wavelet-based electrocardiogram signal compression methods and their performances: A prospective review". In: *Biomedical Signal Processing and Control*. DOI: 10.1016/j.bspc.2014.07.002.

Naud, Richard et al. (2008). "Firing patterns in the adaptive exponential integrate-and-fire model". In: *Biological Cybernetics*. DOI: 10.1007/s00422-008-0264-7.

Popivanov, Ivan and Renée J. Mille (2002). "Similarity Search Over Time-Series Data Using Wavelets". In: *Proceedings 18th International Conference on Data Engineering*. DOI: 10.1109/ICDE.2002.994711.

Qu, Yinsheng et al. (2003). "Data Reduction Using a Discrete Wavelet Transform in Discriminant Analysis of Very High Dimensionality Data". In: *Biometrics*. DOI: 10.1111/1541-0420.00017.

Srivastava, Varun and Ravindra Kumar Purwar (2017). "A Five-Level Wavelet Decomposition and Dimensional Reduction Approach for Feature Extraction and Classification of MR and CT Scan Images". In: *Applied Computational Intelligence and Soft Computing*. DOI: 10.1155/2017/9571262.

Tejero-Canteroe, Alvaro et al. (2020). "sbi: A toolkit for simulation-based inference". In: *Journal of Open Source Software*. DOI: 10.21105/joss.02505.

Touboul, Jonathan and Romain Brette (2008). "Dynamics and bifurcations of the adaptive exponential integrate-and-fire model". In: *Biological Cybernetics*. DOI: 10.1007/s00422-008-0267-4.

Wang, Yuehai, Yongzheng Yan, and Qinyong Wang (2016). "Wavelet-Based Feature Extraction in Fault Diagnosis for Biquad High-Pass Filter Circuit". In: *Mathematical Problems in Engineering*. DOI: 10.1155/2016/5682847.

Weeks, Michael and Magdy Bayoumi (2002). "Discrete Wavelet Transform: Architectures, Design and Performance Issues". In: *Journal of VLSI Signal Processing*. DOI: 10.1023/A:1023648531542.

Zhang, Hui et al. (2006). "Unsupervised Feature Extraction for Time Series Clustering Using Orthogonal Wavelet Transform". In: *Informatica*.

Zope-Chaudhari, Sangita, Parvatham Venkatachalam, and Krishna Mohan Buddhiraju (2015). "Assessment of distortion in watermarked geospatial vector data using different wavelets". In: *Geo-Spatial Information Science*. DOI: 10.1080/10095020.2015.1071064.