Spiking neural network model construction, inference, analysis and applications

William Peer Berg

Master of Philosophy

Institute for Adaptive and Neural Computation
School of Informatics
University of Edinburgh

2022

Abstract

Computational models have long been used as hypotheses to illuminate and unravel aspects of neural functioning, with seminal works including models such as the Hodgkin-Huxley model, which also exemplifies that model hypotheses may be tested with in vivo or vitro experiments. However, designing high-dimensional biologically realistic spiking models can be an arduous endeavour, and may require a large amount of resources in hand-engineering. Automating this arduous process could potentially greatly accelerate computational research within neuroscience. Therefore, statistical approaches have been used to try and aid in such modelling, with the current state-of-the-art being based on approximate Bayesian computation. However, this approach does not scale well with growing model size and complexity, i.e. both the number of neurons and the number of parameters. In the current data driven era, where deep learning is the prevalent state-of-the-art within machine learning, we investigate whether its key ingredient; gradient based optimisation (GBO) may be leveraged for spiking neural network (SNN) model inference. To this end we implement a modular gradient based optimisation framework on top of PyTorch, a modern ML Python library, and test to what extent GBO may be used for SNN inference, particularly because this approach scales well with network size. GBO is tested for a rate based loss metric, the van Rossum distance which also emphasises the timing of spiking, for the Bernoulli or Poisson negative log-likelihood for probabilistic stochastic models, over different classes of SNNs, including generalised versions of leaky integrate-and-fire, non-leaky integrate-and-fire, and probabilistic general integrate-and-fire spiking models, with extensions to subthreshold synaptic current models as the readout signal used during optimisation instead of a surrogate over the membrane potential, which greatly increases optimisation performance. The results show that due to the temporal state-dependence of the membrane potentials and thus spikes in SNNs, the van Rossum distance metric and emphasis on the precise timing of spiking may obscure the gradient signal. As for the spike trains themselves produced by the models, we test whether higher-order statistics are captured to a greater extent in the SNNs irrespective of the loss variability, and compare model fits to generalised linear models as a baseline model. When factorising the spike trains by using non-negative matrix factorisation (NMF), the resulting ensembles reveal similar functional ensembles for the synthetic data - however, the SNNs capture the functional NMF ensembles to a greater extent, reflected in a higher NMF module similarity. When testing whether this was due to inductive bias by SNN model definition by also fitting to biological data, we

found that indeed the similarity was greater for the SNN models than for GLMs, suggesting that GBO may be leveraged to some extent for SNN inference with regards to capturing higher-order spike train statistics. Further, the parameter landscapes formed by the rate-based metric contains no global saddle point, and at best a frontier as the global minima. These observations and results are reflected in the inferred parameters when compared to the ground-truth models for the synthetically generated data, where the inferred SNN likely retrieved a different parameter configuration, while yet capturing the higher-order spike statistics, or solving the task at hand. When introducing a lower-dimensional intput-output task and using a readout of a continuous subthreshold synapse current model as the output, we were able to show that optimisation convergence was robust, and that the tasks were solved with low error - albeit yet with completely different final SNN parameter values. This emphasises the point that the biological brain may elegantly solve different tasks with completely different configurations, as is the case in every individual, and may suggest that we should not seek to retrieve some target parameter values, as much as higher-order spike statistics.

Lay summary

Computational models are a highly useful tool, employed in nearly all domains of science and engineering. A model is in itself a hypothesis of a system that we're studying - whether in biology, physics, or psychology. Some of these models, particularly within computational neuroscience, are more complex than others. As such, they may better and in a more detailed way describe what is going on in the biological system that we're modeling - by in fact being a closer match to the system, such as for instance a collection of neurons from which we have recorded signals using probes. A fundamental issue with these models is however that we do not currently have a satisfactory way of automatically inferring and attaining them, i.e. there is not algorithm or procedure that allows us to infer such model, no matter how much data we may have. Therefore, researchers often spend a great deal of time and resources in hand-engineering such models, which also typically requires expert knowledge. As such, a central challenge in computational neuroscience, which could greatly aid in modeling research, is how one might go about inferring more biologically realistic models. With the recent widespread success of the algorithmics applied in deep learning, which has given rise to successful applications and better models in a myriad of different disciplines, we investigate whether we may combine the technique employed within deep learning, namely gradient based optimisation, with inference of a more biologically realistic class of neural networks; spiking neural networks. These networks are more realistic in that each neuron models a membrane potential, and potentially more parameters that represent biological properties. We look at what has been done up until now for this model type, including a range of different such spiking models, and implement inference using the aforementioned gradient based optimisation methodology in a modern machine learning library called PyTorch. This lets us test to what extent the model reproduces and behaves like the data that we feed it during training/inference, as well as whether the model itself looks similar to what we would expect when comparing to either another model that has generated data, or to some biological properties. Further, we can look at whether the neurons spike in similar patterns and ensembles on a network-level. It turns out that we may find models that behave similarly on a network-level, but not at all at a single-neuron level. This may be expected if one considers that there are many different ways in which a network may capture similar patterns, potentially representing the same feature, or learning to perform some task. This may suggest that like in the biological brain, there is a myriad of possible solu-

tions, and the precise configuration may not matter as much as commonly suggested. Arguably the precise wiring and neural 'configuration' varies greatly between individuals - what is crucial is that we functionally arrive at solving the task at hand well. Furthermore, when we introduced a synapse (the cell that connects different neurons, and relays electric signals between them) model that varies continuously with the neurons' excitation, as well as presented the model with a simpler training signal, both training and task performance were excellent. This might suggest a role in the slower waves observed in biological brain waves in learning, which also contains a parallel to the sleep research literature in that sleep is thought to also facilitate learning, and that we have this synchronous slow-wave mode of brain-wide activity during sleep.

Acknowledgements

I would like to thank my supervisory team for offering their advice throughout my research. I would also like to thank members of my lab group whom offered invaluable support and advice on everything from my project work to mastering stress and finding a flat in Edinburgh. I also sincerely appreciated input relating to mathematics and philosophy from two special friends in Ediburgh whom are outside of my lab group. I would also like to thank friends and family at home, whose support was important to me throughout my research. Lastly, I would like to sincerely thank the head of our collaborative lab at the University of Strathclyde whom gave us access to in vivo data from the brainstem that was analysed with regards to sleep regulation. It was truly inspiring and exciting to be able to see their experimental lab from early on in my project work.

Declaration

I declare that the thesis has been composed by myself and that the work has not been submitted for any other degree or professional qualification. I confirm that the work submitted is my own, except where otherwise stated. My contribution and that of collaborations have been referenced explicitly both in the text, and the collaborations are indicated below. I confirm that appropriate credit has been given within this thesis where reference has been made to the work of others.

Part of the work presented in Chapter 6, namely that in section 6.2 was previously published in the referenced paper: Tsunematsu, T., Patel, A. A., Onken, A., & Sakata, S. (2019). State-dependent pontine ensemble dynamics and interactions with cortex across sleep states. BioRxiv, 752683. DOI [116], in which my principal supervisor is a co-author. I did not contribute to this publication, but was granted access to the data used in the publication after its submission. The data access was granted via the collaboration setup by my principal supervisor Arno Onken with the experimental lab led by Shuzo Sakata at the University of Strathclyde in Glasgow, Scotland. I replicated some of the findings of the original study before proceeding to employ the data in the unrelated gradient based optimisation work presented in this thesis, and have stated that I have replicated findings where this is the case.

Table of Contents

List of Acronyms

GBO Gradient based optimisation

SNN Spiking neural network

NMF Non-negative matrix factorisation

ML Machine learning

RNN Recurrent neural network

DNN Deep neural network

LIF Leaky integrate-and-fire

GLIF Generalised leaky integrate-and-fire

NLIF Non-leaky integrate-and-fire

SGIF Stochastic general integrate-and-fire

GLM Generalised linear model

ABC Approximate bayesian computation

SBI Simulation-based inference

ODE Ordinary differential equation

BP Backpropagation

BPTT Backpropagation through time

GD Gradient descent

HH Hodgkin-Huxley

XOR Exclusive-or

GRU Gated recurrent unit

LDA Linear discriminant analysis

PDF Probability density function

MAP Maximum a posteriori

MLE Maximum likelihood estimation

TNGS Theory of neuronal group selection

REM Rapid eye-movement

NREM Non-rapid eye movement

Adam Adaptive moment estimation

KDE Kernel density estimate

SGD Stochastic gradient descent

frd Firing rate distance

vrd van Rossum distance

STDP Spike-time dependent plasticity

MCMC Markov-chain Monte-Carlo

SNPE Sequential Neural Posterior Estimation

NLL Negative log-likelihood

RMSE Root mean squared error

WN White noise

OU Ornstein-Uhlenbeck

PPT/LDT Pedunculopontine and laterodorsal tegmental areas

AE Auto-encoding

GPE General predictive encoding

DB Database

Chapter 1

Introduction

Spiking neural networks (SNNs) contain variables that represent biological properties such as the membrane potential, cell membrane time constant relating to the type of cell and ion channels, and potentially other neurotransmitter dynamics [58]. Their definition results in dynamics resembling that of biological neurons, including the release of action potentials upon reaching a certain threshold. Due largely to the biological plausibility of this class of neural network models, they are appealing to study in computational neuroscience, as they maintain these biological parallels to an extent which feed-forward networks in deep learning do not [123, 106, 105]. However, SNN inference research is currently limited, and faces a number of challenges due to the temporal nature of SNNs, i.e. the dependence on past and current neuronal activations, which all have effects on the current membrane potentials and spikes, as well as the high complexity associated with this state-dependence. This can result in significantly different neuronal behaviour, and thus also potentially complicate traversing gradients in order to perform model inference. On the other hand, machine learning has seen a recent surge of interest due to the large success of employing gradient based optimisation for recurrent neural networks (RNNs), in part enabled by the increase in available computational power and data allowing to increase the network size (which deep learning typically denotes) and complexity, but also by algorithmic advances, such as by using adaptive moment estimation during gradient descent [97, 6]. Numerous fields have replaced their state-of-the-art models with surrogate neural network models that capture the structure in the data [68, 69, 112, 21, 38]. Thus, it is only natural to ask whether the foundational methodology of gradient descent may be applied in other fields, and arguably of particular interest when considering computational neuroscience, which is after all the origin of neural networks in machine learning, loosely inspired in the

networks of coupled nodes, thought to represent neurons, which can be traced back to [74]. Some of the ongoing research in computational neuroscience is now in fact focused on whether deep learning may inspire model inference within a domain closer to its origin; namely for biologically plausible neural network models [40, 98, 105]. However, these approaches either greatly constrain model fitting by making statistical assumptions, inferring models with a low number of nodes, or using statistical representations to try and learn the relation between SNN parameters and spike outputs. This work is focussed mainly on more direct in-place inference of SNN models using gradient based optimisation, with the aim of accelerating research towards and extending the scalable inference algorithm of gradient based optimisation also to this model class, namely that of biologically plausible and interpretable spiking neural network models.

As more and more data, and data of a higher resolution, is becoming available from neural brain recordings, the potential benefits of developing a scalable approach for model inference only grows. We here revisit the state-of-the art for inferring SNN models using both surrogate gradient based optimisation and some of the most prominent spike metrics, as well as deep neural network amortized learning and approximate Bayesian computation (ABC) [70, 68], and compare how these approaches may be employed for SNN inference. Further, we hypothesise that recent ML techniques may be leveraged for successful and efficient model inference using gradient descent based optimisation, and test this both for leaky integrate-and-fire (LIF) models [93, 24, 88, 13], generalised leaky integrate-and-fire (GLIF) models [1, 111], a probabilistic type of stochastic general integrate-and-fire (SGIF) models [92, 98], and non-leaky integrate-and-fire (NLIF) models [53]. We find that while ABC may be successful for lower-dimensional population-level SNN models, as has been shown recently in the literature, its algorithmic and computational complexity and cost thereof limits the methodology to models with a low number of nodes and complexity. On the contrary, when using a surrogate gradient approach, we find that model inference for larger networks is made possible, albeit only retrieving local minima, and that the inferred models capture the functional ensembles in the spike data even better than coupled Poisson generalised linear models, both for synthetic as well as biological spike data. We perform model inference using the aforementioned LIF, GLIF, and SGIF using both synthetic and biological data, and demonstrate that higher order statistics may be captured (to some extent) even when performing neuron-level model inference over a mixed neuron-type network. The results are verified by comparing geodesic similarities of inferred model

spike train outputs with predicted spike trains produced by fitting generalised linear models (GLMs) (see section 2.5.2) to the target data. Our findings there illuminate that the inference procedure is in fact capable of capturing the higher-order statistics to the same extent as a coupled Poisson GLM. However, we observe that it is highly unlikely to aim for retrieving ground-truth parameters in SNN models when using gradient based optimisation. This is in part due to the issue of constructing a well-defined loss metric, but also importantly due to that there are multiple configurations that may well capture the statistics of the data. Further, we test fitting LIF and NLIF SNNs to lower-dimensional data with a continuous subthreshold synaptic currents model, and find that both model classes; leaky and non-leaky converge and perform well for the different task-instances and associated data sets. This demonstrates that successful gradient based optimisation setup is possible for SNNs, but illuminates that a different training signal than the more high-dimensional and stochastic spike train input-output may be required. The explanation for the success in training using the auto-encoding and general predictive coding tasks may be attributed to a more well-constrained parameter landscape given the lower-dimensional signal, with the signal being far less noisy. However, the signal varies over time, and the network activity itself requires a high degree of synchrony. This begs the question of whether such signals could support, or even be crucial, in learning in the biological brain. After all, we do observe rhythms in the neural activities, or brain waves.

To try and give a more general an intuition about the point pertaining to the loss metric; if we think about deep neural networks (DNNs) in the machine learning (ML) domain - these may approximate arbitrary data well, given that they are universal function approximators [50]. However, this requires that the data are of such a structure that it may be represented and captured by a function which the network may learn, i.e. which may be learnt as a combination of sequential (usually linear pattern) transformations. However, in the domain of SNNs, we introduce several new crucial and non-linear properties, which result in significantly different, if not distinct model dynamics. Each neuron now has a state, which depends on its previous state, and the behaviour is modelled typically as a system of ordinary differential equations (ODEs), containing parameters representing the membrane potential, membrane time constant relating to a refractory period, transmitter interaction, and other biological properties. While this makes the model biologically interpretable with direct parallels between parameters and biological and cellular counterparts, it also changes the entire system's behaviour. The system now has activity independently of input perturbation. The transformation

of an input signal is no longer deterministic in the sense that it will result in one given output given a set of initial model parameters - it now depends on the current system's state, which again depends on the previous state. As such, gradients calculated in a manner similar to that of for DNNs in ML will also depend on the system's and neurons' state, greatly increasing complexity. In a way, one may still maintain the parallel to backpropagation through time for DNNs. However, there is a crucial difference in that a neuron's output is now of a much more binary nature; a spike or no spike, and in that the time series evolution of spikes, whilst potentially encoding much more information, now is a series of spike evolving over time, and thus the target signal cannot be regarded in the same way as in DNNs. This being said, like noted by other authors such as [101] that researchers are looking for a connection between the field of machine learning and computational neuroscience, the former having had its subfield of DNNs created based on inspiration from the brain, but now in a reverse way in which we may leverage the advances from the domain of ML in model construction. Even though we are not there yet, the search for a connection here may both be highly beneficial for the community, and spawn a new sub-field of dynamical systems modelling using inference in computational neuroscience.

SNN inference using GBO could potentially be a great tool in the field, but there are some key issues that we have identified and only partly been able to address in our work that remain to be solved. Some of these are outlined in the subsections below.

SNNs are hard to optimise

In machine learning, we typically only consider approximating a spatial transformation of the input data with our model. Not only does this allow for higher parallelisation of the training algorithm, but it may also greatly constrain the parameter landscape in the sense that it does not depend on the past activity of the network itself, which vastly increases both the complexity and stochasticity of parameter inference. This is an issue for SNNs, where current work for applying optimisation is limited in successfully inferring models using in-place signals. Typically surrogate signals are used [78, 107], and the model complexity and performance is limited. In an attempt to facilitate gradient based optimisation, [52, 53] rewrite the standard LIF-formulation to a non-leaky formulation, essentially removing the leakage term, which then allow us to also calculate the exact gradients of the system given a set of inputs and expected outputs. Since we may computationally set the system's initial conditions and random seed, the

resulting computations have little to no stochastic variability and fluctuations. Further, with no leakage, gradient propagation may be computed exactly according to the error signal. Interestingly, we find in this work that we may achieve the same performance even with leaky models when studying the synaptic model of [53] for their outlined tasks - the crucial part for the model inference performance observed seems to be the continuous subthreshold current synapse model signal, which may provide a much better suited training signal for parameter inference by gradient based optimisation.

In any case, one may hypothesise that there should exist solutions to the set of coupled ODEs that maximise a given loss metric. The question would then be how large the solution space is, and to what extent the different regions are reachable. Furthermore, we are not guaranteed that the solution(s) given by our metric arrives at the ”true” parameters, i.e. most correct given either a synthetic model or biological parameters. Nevertheless, since BPTT [95] and optimisation within ML and deep learning has been shown to be able to be able to infer highly complex data sets [65], it is our goal to study the extent to which a similar methodology may be applied in SNNs, to evaluate the performance of this novel algorithmic design, and to illuminate where work that bears potential may lie. Therefore, we test SNN inference using GBO for surrogate gradients using spike train data as the target data, and compare this with the state-of-the-art procedures in approximate Bayesian computation, and also quantify and evaluate the extent to which we may capture both the spike train data, as well as potentially the ground-truth parameters when the target data is synthetically generated. Further, we also test SNN inference using GBO with the subthreshold model of [53] after observing that we could capture the spike statistics for spike trains, but not the generative model parameters. This was in part to assess whether we would then arrive at the ground-truth parameters with a more continuous synapse model and thus training signal, and also to test whether a more well-defined partly known input could facilitate this. As it turns out, we cannot expect to retrieve the ground-truth parameters in any of the cases, but using lower-dimensional training signals, as well as the gated subthreshold synaptic currents, greatly facilitate gradient based optimisation. This is despite that the error landscape remains very similar, which may indicate that some more stable input-output is required or beneficial computationally in spiking networks.

The binary nature of spiking

Spikes are quite binary events, firing very rapid pulses of action potential, or releasing neurotransmitters, upon reaching a certain threshold. However, binary signals aren't very suitable for optimisation, which is an issue when aiming to employ optimisation for spiking network model inference. As such, a common approach when using gradient descent based optimisation has been to instead optimise over a surrogate signal, such as for a function of the membrane potential instead. Using a function such as the sigmoidal function, this then lets us compare the surrogate signal directly with a target spike train - which we may also choose to transform to a shape or signal that will lie closer to the surrogate signal. One such example is the van Rossum distance [118], in which each spike pulse is convolved with an exponentially decaying kernel, such that we may operate over a smoother signal. However, this signal transformation transforms each spike pulse the same, and doesn't really provide us with more than just spike-time information (although in the optimisation procedure, the idea is that the signal is more widely and continuously available during inference - i.e. that it allows to 'align' two spike trains if one is just slightly shifted). If, however, some of the timing information is relevant - as is arguably the case biologically - such a metric should provide a better training signal for optimisation than simply a purely rate-based metric. This is however based on the assumption that emphasising the timing of spiking is meaningful also from a gradient based perspective, and not just a general information theory perspective. In this work we therefore test using the van Rossum distance as a loss metric when performing adaptive moment estimation gradient descent with varying temporal kernels which effectively lets us vary the metric emphasis between a rate based (a wide kernel) and a precise spike-time based (a slim kernel). In both cases each spike will represent a peak in the error landscape, and we will traverse the error landscape such that we try to align these peaks (and spikes) between the model and target spike trains.

Another way of addressing the binary nature issue is by incorporating the signal below spike-threshold, i.e. the membrane potential, and have this generate a signal inside of an active zone, such as when the potential is above 0 [52]. It can be argued that this is more biologically realistic, whilst also offering the advantage of a continuous signal centred around potential spike pulses, or action potentials. As such it also provides a gradient signal for spike generation even when the potential may be below threshold, still making the excitation 'visible' to the optimiser, rendering the sub-threshold gat-

ing synapse model a more fine-grained candidate than surrogate membrane potential signals, as it contains richer information.

The parameter landscape contains a frontier of local minima (at best)

Even when addressing the issues above with the proposed approaches, and even if fixing all model parameters but the weight matrices, there are multiple weight configurations that may result in very similar behaviour and outputs. As noted by other researchers in the DNN as well as SNN literature; the initial configuration may have a profound effect on the resulting inferred model [115, 4, 104]. This also holds true for SNNs, where the previous model state also has an effect on the future state, and the model neurons may be brought to exert different modes of behaviour, depending on the previous and current input and state. In any case, this begs the question of how hard we want to pursue the retrieval of the ground-truth parameters. In fact, is retrieving the ground-truth at all something we should strive for, or necessary? When it comes to learning meaningful representations in a network, biologically speaking there are multiple configurations in the motor cortex that may elegantly solve a motor task at hand [72]. The question of the relevance of the ground-truth parameters is one that has been particularly problematic for me throughout my research. No matter how hard we tried to shape the parameter error landscape, there was always a myriad of parameter combinations, and a rich parameter solution space, that would allow for modeling and capturing the spike train data. This is in line with the point discussed below in 1 about sloppiness on the network level, and the observed flat regions in the parameter error landscape, and raises the question of whether it may at all be possible to infer ground-truth parameters, as well as whether it is meaningful to do so. After all, the goal of in this thesis is primarily to model the target data, but as a side-goal we wish to test whether and to what extent the parameters match those of the synthetic models that generated the data (if the target data was generated in silico).

Optimisation may aid in automating model construction by inference

More complex models, therein SNN models, often require expert resources during implementation and/or model design, in which certain parameters have to be constrained

or calibrated according to a desired target regime of behaviour. This may to some extent result in more detailed models that may demonstrate and thereby hypothesise some of the dynamics that are at play in generating some observed data. However, in the case of SNNs this may not be sufficient for capturing the network organisation, or other desired higher-order statistics such as functional ensembles. In any case, model inference may be a better and more importantly less time-consuming alternative, which may also be used easily in conjunction with more manual processes if so desired.

In sum

This work surveys the literature on GBO related to SNNs and introduces the model definitions, loss metrics, evaluation procedures, and algorithmic approaches in chapter 2, demonstrates a way in which GBO may be implemented and performed scalably for these various SNN model classes, and tests the limitations of the procedure, as well as connects it with and compares it with the state-of-the-art for a specific stochastic model type, and for a prominent simulation-based inference by approximate Bayesian computation approach in chapter 5 (with an introduction to the simpler leaky integrate-and-fire model in chapter 4). The goal is to (1) test whether we may capture spike train data (both biological and synthetic) in SNN models using gradient-based optimisation (GBO), and to test whether the parameters can be assumed to be close to the ground-truth by comparing with the ground-truth parameters when fitting to synthetically generated data, and (2) based on the finding in (1) that we may capture the spike statistics, but not parameters, to test whether we may capture the parameters by greatly constraining the target and training signal to lower-dimensional tasks. Note that these different setups also allows us to assess the performance of GBO under different network input-output conditions. Whilst we find that the neuronal parametrisations vary greatly in both cases, we find that performance is excellent for low-dimensional non-noisy signals, but not for spike trains, for which the input is also assumed to be unknown.

Further, using the insights gained in chapter 5, and mainly to also test the framework and approach on biological data, as well as with the goal of assessing functioning relating to the site from which the data stems, we apply the inference methodology and framework to spike train data from the rodent brainstem in chapter 6, and assess the extent to which we may capture higher-order spike train statistics by non-negative matrix factorisation analysis and comparison. Due to the rich number of potential model

configurations/solutions, as illuminated in both chapter 5 and 6 - particularly with the parameter error landscape plots - what can be hypothesised from the specific inferred parameter values is highly limited. Lastly, before our concluding chapter, inspired by the limitation pertaining to exact parameter estimation, as well as convergence and training issues with and difficulties of constructing the loss metrics for SNNs when fitting to spike train data, we extend the work of [53], and interestingly find that using their synapse model allows for excellent task performance and convergence rates also for leaky models (thus the exact gradient calculation is not a strict requirement for the procedure to be successful). This demonstrates that synapse models containing richer subthreshold information may be key to enabling scalable and robust SNN inference using GBO, and also highlights that while there are multiple, different configurations that may be inferred by the approach, they may be equally elegant in solving the task at hand - as has also been noted by neuroscientists when studying data relating to the motor cortex [72]. On this note, we also briefly discuss the possible implications of this work for representational drift and sloppiness in this chapter, which is compatible with both observations in that the error landscape is observed to be ”sloppy” on the network level with flat regions in the parameter hyperspace, but drifting on the neuron-level, as seen in part by that their parametrisations and responses vary drastically between inference runs.

To sum up, GBO may be leveraged for a more scalable SNN model inference procedure, which may capture the data that we are fitting to, but not the ”true” generative model parameters. While the inferred SNNs may thus not be used for ”ground-truth” parameter estimation, it may yet be analysed and probed functionally to asses characteristics and dynamics of the site which the spike data stems from, or as a starting point for model construction. When testing this hypothesis on biological data, we find that the same as observed for GD SNN inference on synthetic data holds. Namely that we may still attain a higher geodesic NMF module similarity for spike trains predicted by inferred SNN models than by coupled Poisson GLMs.

Chapter 2

Background: Biologically plausible computational modeling

In disciplines pertaining to biology, maintaining a more direct link in hypothesised models by incorporating a greater level of detail may allow for correspondingly more illuminating findings [54]. However, this comes at the cost of increased complexity, often resulting in that construction of such models by sheer hand-engineering is too time-consuming, as the combinatorial expansion of possible solutions to model parametrisations quickly rises to one above where manual search is out of the question. Of course, hand-engineering often involves both expert system and domain knowledge for constraining the semi-manual parameter search, and model proposals, and some type of programmatic model search. However, inference of spiking neural networks remains an unsolved task [105, 20]. With the recent advent of widespread success of deep learning models for arbitrary data sets, we here revisit applying a similar inference procedure for automating inference of biologically plausible models, too - namely spiking neural networks.

There are multiple model definitions and topologies one may consider when studying neural network architectures, which in turn affects network behaviour and inference. Ranging in complexity from the original perceptron [74, 94] to the Hodgkin-Huxley (HH) model [47], and from single-neuron to multi-layer networks. On the one hand, the HH model is complex enough that fitting the parameters of a single node using electrophysiological data requires a significant amount of computational resources, which in turn allows the model to accurately replicate biological membrane potentials and spikes. Perceptrons, on the other hand, are far less costly to train and fit, and of a different nature entirely, requiring only straightforward matrix multiplications for

weight inference, with the nodes' values being either 0 or 1 - however, the behaviour these networks may exert is highly limited and implausible. A common mentioned example to illustrate this is that a two-layer network consisting of perceptrons cannot learn or perform the XOR-task, as hidden layers are required to perform this non-linear separation. More complex models, however, may incorporate non-linearities also in single nodes. The question is whether we may learn from learning and inference algorithms that have been successful in one domain of neural networks and apply it on more complex classes of network models, and if so what the limitations are, and how one may address these. More specifically, we wish to address the question: Can we employ gradient-based optimisation (GBO) for inference of spiking neural networks (SNNs), and how well does it perform for this temporal, more complex class of neural network models? What are the limitations, and do there appear to be any ways of addressing these?

These are some of the questions that we have kept in mind when focussing on inference of spiking neural networks (SNNs), along with that they a maintain a higher level of biological plausibility. The latter point is relevant when considering biological data. To elaborate on this point: When evaluating the performance of GBO for SNN inference, we wish to study both how well models can capture target data, as well as whether we can expect inferred model parameters to lie within close regions of the ground-truth parameters (when these are available, i.e. when training using synthetic data). When it comes to whether we may capture and model target data, we study this along two strands: (1) with the aim of capturing target spike-trains in order to be able to model functional organisation on the network-level, and (2) with the aim of performing a simpler task - motivated by the finding that we cannot expect to retrieve ground-truth parameters, but that we may capture higher-order spike-train statistics. The second strand is also an interesting one with respect to capturing ground-truth parameters, as we also remove noise and train on a simpler signal - which then strengthens the hypothesis that we cannot aim to retrieve ground-truth parameters via GBO. In fact, it raises a more philosophical question about whether this is at all meaningful. More on this later on in the discussions of the various chapters.

When it comes to GBO for SNN inference there are different approaches for tackling the associated increased complexity, as well as binary nature of spiking itself, resulting in non-linear changes of the internal neuronal state and associated variables [56, 105, 113, 108, 3, 78, 8, 82, 107, 75, 34]. On the one hand the temporal unfolding resulting from the internal state changes requires loss metrics that are adapted to also

measure this non-linear temporally dependent effect, and on the other hand the limited information in biological data poses a constraint on the extent to which we may emphasise such information. Traditionally a type of rate-based encoding has largely been the preferred rate-metric for SNNs due to its simplicity and robustness to sudden spike-changes, however this comes at the cost of being more or less agnostic to, and not making use of the rich information available in the precise timing of spiking. Another reason why rate-based metrics has been used is that it has been hypothesised that the brain may employ a rate-based encoding scheme. However, it has more recently been accepted that this scheme is not succinct in capturing the rich information processed by the brain, neither on an information content level, nor on a processing speed level (i.e. quick neural computation and responses, on the order of milliseconds), as has been shown by information theorists [10]. Therefore, it is our aim in this work to incorporate temporal information into the spike metrics used for gradient-based optimisation, in order to try and model not only the neuronal rates, but also the timing of spiking. If successful, non-negative matrix factorisation should infer spatial modules of similarly temporally co-active ensembles of neurons in the model as in the target data. Note however that this does not measure to what extent we have captured spike-patterns unfolding in space and time in the model. However, this is arguably far out of scope in this attempt to combine GBO and SNN inference, as this would require a highly precise model, which we cannot hope to achieve by a best-effort GBO methodology that currently lies in between a rate and precise timing encoding in the loss metric.

Population-encoding [92] is another type of approach which may make readouts and learning more stable in a network, but also imposes a significant information bottleneck on the amount of information that may be processed and encoded by the network. It is likely that the brain employs an encoding scheme in which the rich information contained in the timing of spikes is relevant [15]. However, as we shall see throughout the work studied in this thesis, it is not possible to do so simply by means of a global gradient descent procedure, which seems to be neither sufficient for precise timing emphasis, nor is biologically plausible [51, 118, 40]. This being said, the scenarios we have tested only include conditions where the input is unknown, and only the probabilistic nature of it is incorporated. We also only study using error back-propagation by using rate based loss metrics, which was found to be the best-performing procedure when doing gradient descent when fitting to spike train data. Surprisingly, when fitting to a lower-dimensional signal, such as a linear combination of a sum of sine-modulated white noise inputs, we may retrieve a model configuration

that almost perfectly learns the input to output mapping, by extending the work of [52]. There are a number of key aspects for why this works, including greatly constraining the parameter landscape by using a lower-dimensional target signal, by using a well-defined, non-noisy signal, and by using a means of gradient-descent that in fact may be said to approximate a type of STDP, as we are using sub-threshold synapse currents instead of a surrogate over the spike signals - which allows for distinct sub-threshold signals instead of a uniform signal surrounding each spike. For further reading on different learning procedures currently studied in the context of SNNs, we refer the reader particularly to [105].

2.1 Fundamental differences between SNNs and spike train PDF models

In this thesis, we have mainly focussed on spiking neural network models, which we regard as a set of neurons that are each modelled by a set of ordinary differential equations (ODEs), and coupled by a set of synapses, which are in turn modelled by a set of ODEs, ultimately producing spikes, or spike trains when considering intervals of activity. These spikes may either be completely binary as in the spike trains decoded from in vivo LFP recordings, or continuous in the synthetic model formulations where we use continuous sub-threshold synapse currents as the spike readouts. In the case of target data, however, we have chosen to always consider binary spike trains as the target data when fitting to spike trains - except for in the chapter on inference using continuous sub-threshold synapse models and fitting to a lower-dimensional target signal (i.e. encoding tasks), where we investigate whether some of the issues identified when fitting to the arguably less informative binary spike trains could be ameliorated. In order to place our work on SNN inference using GBO into the context of the current state-of-the-art and field of current research, we have also chosen to include probabilistic spike train models, in which each neuron is modelled by assuming that it produces spikes according to a probability density function, such as a Poisson probability distribution and corresponding density function, or a Bernoulli probability density function. This also lends itself to GBO, albeit by using maximum likelihood estimation over the PDF parameters, i.e. by minimising the negative log-likelihood given by the PDF. We compare both our SNN GBO results with the PDF-based SNN GBO results, as well as with published results on PDF SNN inference [92].

As mentioned above, a spike train can be modeled by drawing from a probability distribution $P(X;\lambda)$, such as a Poisson distribution, where X here denotes the number of observed spikes, and λ the rate. This is the case in the generalised linear model (GLM), which is a fairly simple, yet robust spike train model. Due to that spike trains can be considered to be drawn from a Poisson probability density function, i.e. distributed in a Poisson distribution, we may expect to capture neuronal rates if simply using a set of random Poisson variables for each neuron. If we take this one step further, and consider a coupled Poisson model, in which each node is coupled with the others by a weight matrix, and the activities are propagated using a link function, this forms a model that should also capture some of the ensemble activity as seen in the target data. Since such coupled Poisson GLMs have been successfully used in other modelling work, we also fit this model class to our target data and use it as a baseline model for comparison with models inferred using GBO. GLMs may capture spike correlations and rates well, but not exert different neuronal modes of behaviour in the way that SNNs may (see [54] for different modes of behaviour exhibited by different SNN formulations). I.e. it is a spatio-temporal model with the temporal signatures being limited to static functions over spike-responses, which may be somewhat affected by input, but not fundamentally change its dynamics and behaviour, as opposed to in SNNs, which may exert distinct modes of behaviour. The golden standard should as such be to infer models that also capture different modes of behaviour, thus also the dynamics that are at play and may be exerted under different input conditions. This may be extremely hard to capture using current inference methodologies, and has to the best of my knowledge not been successfully attained by the research community. In this work, we use non-negative matrix factorisation (NMF) [100, 99], described in more detail later in this chapter, to assess the functional ensembles captured by models, as it has shown to be able to capture ensembles sufficient for predicting future brain state almost as well as the raw neuron-signals themselves, suggesting that the factorised modules is a suitable representation of potentially functional ensembles, and may well pertain to the dynamics observed in the data. Further, different input and stimulus conditions is in fact something that may be tested both synthetically and in vitro or vivo by stimulation e.g. patch-clamp tissue stimulation or optogenetically [117].

2.2 The link to ML

As mentioned previously, SNNs are systems in which each node is modeled by a set of ordinary differential equations (ODEs), where each variable or parameter may have a more or less direct biological parallel, such as representing the rest potential E_L, the spike threshold V, and other properties. Together, these result in a model whose membrane potential v_t has a temporal trajectory similar to that of biological neurons. The main idea is that under the right conditions, SNNs may mimic biology so closely, that we can in fact use them to study what might be going on in the modeled brain area; the model becomes the hypothesis. The stronger the link between the model and the brain area, the stronger the hypothesis, and associated predictions made by perturbing and probing the model under specific conditions.

In RNNs, however, the criterion of each node or neuron being a system of ODEs that maintain biological parallels is relaxed or no longer valid. Each node may simply be one value that is some sum over synaptic input that has been transformed by a transfer function, or it is more commonly some slightly more complex unit such as the gated recurrent unit (GRU) [5, 17] or long short-term memory (LSTM) unit, engineered for application in the domain of machine learning [46, 97]. When dealing with nodes that are commonly used in machine learning, the temporal dependence is more limited than in more complex SNNs and neuronal ODE systems. Further, RNNs are usually trained on tasks in which the input is known, and is some specific transformation of inputs to outputs, such as is the case in image processing data for CNNs, which forms the data set. In other words, the data modeled is usually less noisy and of a more straightforward nature than (decoded) brain recordings such as spike trains, which it might be argued are more heterogeneous from an information theoretic input-output mapping. At the very least, the mapping is unknown, including in our work when using spike trains the input. These aspects result in that work on RNNs consider a setup more suitable for chaining the error gradients backwards in time, which may be done over each time-step for each error gradient by a modification to the backprop-algorithm by chaining the error gradients, known as back-propagation through time (BPTT) [95]. Note also that RNNs typically deal with continuous, smooth signals in each node, whereas SNNs incorporate non-linear value-changes upon binary spike-events, which also greatly complicates training, let alone inference of model parameters. [78] describe the connection between RNNs and SNNs well, including the efforts that have been made to employ (surrogate) gradient descent for spiking neural networks. In or-

der to give an intuition about the differences, between SNNs and RNNs, the definition of the GRU unit is included below, and may be written as,

$$\mathbf{h}_t^j = f(W^{i,j}\mathbf{h}_t^i) + \sum_{i=1}^{L} g^{i,j} U^{i,j} \mathbf{h}_{t-1}^i \tag{2.1}$$

where h is the hidden units, W the weights, and g a gating function,

$$g^{i,j} = \sigma(\mathbf{w}_g^{i,j}\mathbf{h}_t^i + \mathbf{u}_g^{i,j}\mathbf{h}_{t-1}^*) \tag{2.2}$$

basically treating the network instead of a conventional RNN which may be regarded as a process in which each hidden layer has gated recurrent connections determined by a reset gate function, which allows the ignoring of previous hidden states by transition through the reset gate function, which greatly simplifies optimisation when compared to SNNs, where the gradients necessarily are chained backwards in time for all previous state. The reset gate function weights may also be updated by the same methodology; estimating a probability distribution over sequences, factorising the probability of the sequences, and training an RNN by means of negative log-likelihood minimisation of the training sequences:

$$p(x_1,...,x_N) = p(x_1)p(x_2|x_1)...p(x_N|x_1,...,x_{N-1}), p(x_N|x_1,...,x_{N-1}) = g(\mathbf{h}_t) \tag{2.3}$$

Note that this bears resemblance to SNNs, and is brought somewhat closer particularly by the probabilistic model version of [92], which we have based a spike time and gradient compatible model implementation on in section 2.6.4. There, a spike train is assumed to be a Bernoulli or Poisson distributed spike history, which may be estimated with the ODE system and its recurrent connections. However, connections in a GRU [17] system are optimised in a more straightforward manner, with the other parameters and internal dynamics instead being what decides on the ”gating”. Also, the weights are typically the only inferred parameter in ML models using some gradient based optimisation algorithm. As such, it might be argued that the link between RNNs and SNNs is somewhat weak - but we have chosen to include a short section on RNNs for the sake of consistency, and to more clearly outline and contextualise the GBO methodology for SNNs by describing the model class that is arguably the most closely related to SNNs and also importantly for which GBO is successfully applied for inference.

There exists work in which SNNs are transformed such that they are more compatible with BPTT, including probabilistic models in which each node projects a probability of spiking for each time step or interval, and models in which synaptic signals are made continuous and smooth, either by constructing a surrogate gradient signal such as over the membrane potential, or by modeling the signals as a function over the sub-threshold membrane potential, such as for instance when it is above zero, or within a given interval, or by modeling sub-threshold synaptic currents. The same often holds for SNNs in which surrogates are constructed, due to elements such as the initial model configuration, initial conditions, or model perturbation. Works such as [57] employ a rate-based error metric for data and readouts pertaining to image classification tasks. By considering how these complex systems behave differently under different conditions, and that they may be heavily affected by their current and thus past state, it may be seen that designing an error-metric that emphasises the timing of spiking is problematic. As such, approaches pertaining to rate codes have seemed to largely be the extent to which it is possible to incorporate temporal precision whilst successfully optimising and inferring models. While these works are valuable in studying how one might combine optimisation and spiking models, they often are limited not only to rate-based encoding [**?**, 11] and/or loss metrics, but also to non-noisy ML domain types of data. However, the sole basis for studying SNNs in this work is to maintain the biological parallel, such that the model may be used as an interpretable system and hypothesis of recorded site dynamics, and studied as such. Therefore, it is important to study how the precise timing of spiking may be made relevant in the models, as that is arguably not only a relevant source of information, but a rich encoding scheme employed by the neural circuitry in the biological brain. While this was initially debated, it has been argued both by information theorists, neuroscientists and computer scientists alike recently that the precise timing of spiking is not only relevant, but is in fact necessary in order to encode all of the information that has shown to be processed by the brain, i.e. a rate-coding is not enough to represent a task which is solved by a neural network.

In our work we have studied employing gradient based optimisation as well as approximate Bayesian computation using amortised learning for both a probabilistic class of stochastic spiking neural networks [92], for leaky types of integrate-and-fire SNNs [1], and for closed-form exact-gradient types of non-leaky integrate-and-fire SNNs [52]. We have also replicated and analysed Izhikevich SNNs, but chosen not to apply GBO to this model class due to planes of chaos in the parameter landscape, sim-

ply resulting in that the Izhikevich neuron ODEs are a lower-dimensional projection of the Hodgkin-Huxley ODEs - resulting in that while they are of lower complexity simulation-wise, the collapse of several variables into fewer results in variable intervals for which model behaviour is unrealistic and/or chaotic, making GBO impossible.

2.3 Bayesian approaches

We assume that the reader is familiar with Bayesian statistics. However, just to give an outline of our notation to the extent that we may also outline the SBI approach used, we may define a sample of observations O given priors p, making the posterior over the parameters given an observation,

$$P(\theta|X_o) = \frac{P(x_o|\theta)P(x_o)}{P(\theta)}, \tag{2.4}$$

Where $P(\theta|X_o)$ is the probability of the (model) parameters θ given the observations X_o (spike train), here written out using Bayes' rule, which is used to infer a posterior over the prior, given specific observations (spike data x_o). One way of doing so is to generate observations with a simulator, for which we may generate samples over priors, enabling us to estimate a full posterior over the parameters for a given observation [70, 68]. Note that this does not avoid the difficulty of modeling the more 'external' conditions, including the model perturbation and external input. Further, the computational cost requires sampling a great number of samples in order to estimate the posterior over the parameters, thus increasing drastically with the model dimensionality and number of parameters. The approach does, however, come with the benefit of being able to assess and directly measure the certainty by usual statistical metrics around the centres of the posterior distributions, as well as that the full posterior distribution contains samples across all parameters, dependently, which may capture dependencies not so easily captured when estimating using GBO.

2.3.1 Amortised learning with DNNs

[68] propose using sequential neural posterior estimation, in which a simulator network to generate samples, and a deep neural network to approximate the posterior over the parameters given the sampled observations. This can then be used to estimate the posterior given a target observation, which may be outlined as,

$O : \Phi(\theta) \sim X, \quad DNN = F_\Phi(\theta; X) \approx P(\theta|X)$

It seems that DL is an excellent candidate for this type of function approximation, and that in this regard GBO works just as well as it does in ML, depending on how well-defined the posterior distribution in fact is in the data. We find in our work that particularly when working with models that have larger spaces which may be thought of as ”frontiers” in the parameter landscape for which the loss is approximately equal, the posterior is similarly not condensed around fixed points. [69], however find that the posterior approximation still contains the relationship, i.e. dependence, over the parameters. In other words, while they cannot be sampled independently for marginals, sampling over the full posterior still generates realistic data, even if necessarily still suffering from the same local minima samples from the posterior, due to the input-output ambiguity.

2.4 Other key works

2.4.1 Stringer et al., (2016)

In this section, I have included some key relevant published works, presenting some of their key findings, advantages, and limitations, and how they relate to the work in this thesis.

It should be mentioned that whilst pursuing our line of research on GBO for SNN inference I was unaware of one key publication, which is highly related to the work in this thesis, namely that of [103]. In this publication the authors fit quadratic integrate-and-fire spiking models to multi-neuron recordings from different rodent species and brain modalities across different brain states, which is essentially what my goal was initially, and is to a large extent what is attempted in chapter 6, albeit for only one rodent species, but across several individuals. However, [103] use non-linear Monte Carlo based approaches for model fitting in conjunction with first sequentially identifying meaningful parameter regimes for each parameter that had a significantly pronounced effect on model behaviour, in contrast to our gradient-based ”in-place” optimisation approach, i.e. adjusting parameters of the model throughout inference.

Notably, the authors of [103] state in their introduction that: ”For our results to provide direct insights into physiological mechanisms, we required a model with several properties: (1) the model must be able to internally generate the complex intrinsic dynamics of cortical networks, (2) it must be possible to fit the model parameters directly to spiking activity from individual multi-neuron recordings, and (3) the model must

be biophysically interpretable and enable predictions that can be tested experimentally.", [103], which is highly similar to what our goals were when setting out to devise a methodology for SNN inference. In our work however, we assumed some external model perturbation, upon which the model was dependent, and found that we could not retrieve the ground-truth parameters. That being said, the authors of [103] employed statistical methods to look at distributions of parameters, constrained after sequential parameter searches that were found to pass qualitative similarity, in which sequential "wide parameter sweeps" over the model parameters were performed and used to identify parameters that may exert a significant effect on model behaviour and meaningful intervals for them. They also observed different modes of behaviour similar to those reported in [54], and chose parameter intervals for which activity was observed to be qualitatively similar to that as observed in the target data, hinting that ground-truth was not a goal, but behaviour was. Further, there is no mention of the ground-truth parameter distance. However, the authors reported accurately capturing diverse patterns of activity without requiring external stimulus in the models, due largely to the intrinsic dynamics generated by inhibition, and the different strengths in feedback inhibition. One question that arises based upon these results is whether they have identified behavioural regimes, with corresponding regions in parameter hyperspace - particularly because the Izhikevich model, for which a subset of the parameters may form such regions (see chapter 3 for more on this). In any case, these findings may provide some clues as to what may be captured in spiking models per se, as well as what is needed in order for the system to more robustly model and capture the spike statistics and precise patterns of activity; namely a range of inhibitory dynamics, which is the main focus in [103]. This is notably in line with other works on that feedback inhibition drives synchronisation [89]. Further, the synchronised population patterns of activity bears resemblance to my complementary poster on the Izhikevich SNN model 3. It would be interesting to test whether some of the synchrony could also stem from a subset of the model variables, under some stable inhibitory configuration, or if it is simply due to the hypothesised dynamics associated with the variability in the feedback inhibition along with the internal model variables' states. It is also worth mentioning that the authors outline the link between brain states and inhibition, and specifically mention acetylcholine and norepinephrine, whose activity are particularly seen as increased during states with higher desynchronisation, such as during wakefulness and REM sleep. In [103] the parallel is mostly drawn to cortical neurons, for which the neuromodulatory effects on excitation and inhibition is not yet well understood. The same holds for

the brainstem, from which PPT/LDT neurons were recorded in the target data used in chatper 6, for which both the role of the neurons as well as the neuromodulatory effects remains highly debated and unexplained, particularly wrt sleep regulation.

Algorithmically, the authors of [103] developed a novel computational technique for fitting model parameters, involving a non-linear parametrisation, as well as non-linear loss functions over the parameters. Due to the non-linearities, the authors employed a non-gradient based method, and used Monte Carlo simulations, lending itself well to various statistical methods that is based on sampling to estimate a posterior. To make up for the increased cost associated with this, they wrote a GPU-compatible implementation, and also used Gibbs sampling with simulated annealing to further scale up computational speed. Fundamentally, however, the computational cost associated with Monte Carlo methods cannot scale very well to models of increasing size and complexity, although GPU-implementation and various algorithmic tricks such as Gibbs sampling are important contributions for amending the issue of algorithmic computational complexity and cost. Therefore, investigating GBO for model inference remains relevant also in terms of scalability.

Lastly, the authors of [103] scaled the stimulus such that the model response spike rate matched that of in the spike recordings. This is a clever trick that one may consider doing in future research using GBO for SNN inference, although it may also limit parameter exploration somewhat due to constraining the neuronal response(s).

2.4.2 Zenke & Ganguli, (2018)

The authors of [124] propose a framework and model called SuperSpike, in which the van Rossum distance is used to calculate gradients for deterministic integrate-and-fire neurons, but using a continuous auxiliary function over the membrane potential as the surrogate output, rather than the spike outputs themselves, resulting in a three-factor multiplicative voltage based nonlinear Hebbian-like learning rule with STDP-properties.

Interestingly, the authors used two types of stimuli, depending on the task; frozen Poisson noise as the input for tasks where the precise timing was important, or pseudo-random stimulus with similar firing rates, but different firing time offsets. In the first case, this fixes the input such that the output may be better used for convergence due to the deterministic model (and target data produced by the model), and is one way of addressing the issue pertaining to the input being unknown, and should drastically

improve the applicability of the van Rossum distance metric, specifically. Because we aimed to perform model inference for biological spike train data for which the input was unknown, we chose not to fix the input to frozen Poisson noise. In retrospect, however, it may be argued that a transformation over a frozen mapping, if converged towards an accurate spike timing of the target output, may be as informative, or even more informative, than a less accurate model fit not using frozen Poisson input as its perturbation, although less realistic in its learning scheme. However, it may also be argued that on the contrary, freezing the Poisson noise may not only be highly unrealistic, but also lead to a highly skewed inference procedure, which may introduce issues including only exploring a subset of the parameter landscape given the less variable, frozen input, or potentially break down entirely under different perturbation schemes that may bring the model into an unstable state. At the core of our concerns in freezing the input was not exploring the parameter landscape very well. However, as it turns out, this trade-off may have resulted in even more catastrophic effects on parameter exploration than the frozen Poisson noise input scheme, since we could not employ the van Rossum distance as the loss metric due to the high variability in the input, which then correspondingly varied the output spikes, and thus rendered an emphasis on the precise timing of spiking less meaningful, or even misguiding, as observed in many of our experiments.

The synaptic model, eligibility traces, error signals, and corresponding parameter learning all have biological parallels, which makes the model interesting not only from an inference perspective, but also with regards to biological learning. It is unclear however whether more parameters than the weights were inferred, and whether these were at all close to ground-truth parameters in the case of using synthetic target data.

In any case, the authors managed to notably capture complex, precise-timing of output spikes in single-neuron, multi-neuron, and multi-layer deterministic LIF models, by using frozen Poisson noise as input. It would be interesting to design a setup where one looks at gradually increasing input variability, and how this might impact the output, van Rossum distance, and model convergence.

2.4.3 Nicola & Clopath, (2016)

In [81], the authors use the ML method of FORCE-training for learning complex patterns in spiking neural networks, such as a songbird song (using the spectrogram over sound waves as the input), a Beethoven song (using a transformation of the notes as

the input), or complex spike pattern and replay of a movie (with parallels drawn to hippocampal memory replay).

Methodologically, the FORCE-method is implemented as to decompose the weights into the sum of a linear combination of the weights and some static matrix, and a learnt output decoder, which is learnt by recursive least squares. While the FORCE-method may be biologically implausible, the successful application to SNNs for capturing the complex spike outputs is promising in modeling, and may well be competitive with the methods put forth in this thesis on GBO. However, the approach only pertains to learning the model weights. It is also interesting that the authors note that the Izhikevich model was the one observed to lend itself best to FORCE-learning. One speculation why this might be is that they can naturally become oscillatory pattern generators, as seen in [83], and in chapter 3, which in turn should provide a model that lends itself well to generating signals that may then be decoded into some target task signal, particularly for the cyclic data and tasks described above. Arguably, songs with notes, sound waves, and short movie replay is highly cyclic. It would be interesting to see the FORCE-method applied to all SNN parameters, as well as to less cyclic data, such as for in vivo activity recorded throughout different brain states.

2.4.4 In sum

Different works on SNN inference incorporate different methods, including gradient-based error backpropagation, or the FORCE-method. The main focus is on capturing the spike output by adjusting the parameters of the model, which is mostly done by adjusting the weights, as is typical in ML models. Modeling and capturing the output is attained with varying degrees of constraints being put on the input, and in some without external perturbation. Notably, the goal is NOT to capture the ground-truth model parameters that are inferred, but to capture and model the output. Therefore, parameter inference is fixed to the ones that are necessary in doing so, such as the weights. This may be an important point to consider, as it might be argued that one cannot hope to fit all model parameters in parallel, let alone that they will converge towards the parameters of the model that generated the target data. The reason for this is arguably that there is a myriad of parameter configurations that may produce similar outputs, and thus we may converge towards an entirely different set of parameters.

Another point to make is regarding the scalability of learning and inference algorithms for SNNs. SNN simulation in itself requires on-line simulation due to the state-

dependence on for instance the membrane potential, and can only be partly ameliorated by batch-simulation, where the idea is that averaging over batches in parallel approximates the same GBO inference as when doing inference completely sequentially per model. Batch-training for SNNs is to the best of our knowledge a novel proposal in its combination of batching and SNNs put forth in this work. When it comes to statistical methods, they may also be parallellised similarly, but require sampling over a large number of simulations, which scales very poorly with network size and complexity, as is the case in [103]. In other works where a type of error-backpropagation is performed, precise timing may be successfully emphasised and spike patterns learnt by learning model weights when freezing input drawn from a Poisson PDF as in [124], or by using FORCE-learning over some sum over an output-decoding and a combination of a set of fixed input data and model weights in an Izhikevich model, in conjunction with parameters that were determined to allow for meaningful weights inference.

2.5 Evaluating SNN models

While it is non-trivial to exactly evaluate the information processing of two networks, i.e. the precise functions they compute, we may use statistical measures such as spike correlations, or dimensionality reduction methods to identify prominent features, or modules of co-active neurons. As mentioned previously in the introduction, we use non-negative matrix factorisation (NMF) [100, 99] in order to identify these sets of co-active neurons in a spike history, which have been found to be good low-dimensional abstractions of functional behaviour, further strengthened by the finding that factorised modules are as good state predictors as the raw neuron-signals when applying linear discriminant analysis (LDA) or random-forest regression, as done in [84], and replicated in 6.

2.5.1 Non-negative matrix factorisation (NMF)

The NMF factorization may be written as,

$$M \approx WH, \tag{2.5}$$

where the original data set M of dimension $n \times t$ is factorised into two matrices of dimensionality $m \times n$ and $m \times t$. What makes this dimensionality reduction method particularly suited for spike train data is its non-negativity constraint, which naturally

leads to a parts-based representation, and which importantly holds for all parts when it comes to neural spike activity.

While NMF may be used to discover spatial and temporal firing patterns in spike data, the geodesic similarity measure between the modules may be used to assess how well these features were recovered in a fitted model. The geodesic distance o between two matrices A and B of dimensionality $m \times n$ may be written as,

$$o(A,B) = (1 - \frac{2}{\pi}\cos^{-1}(A^T B)), \tag{2.6}$$

where m is the number of modules, and n is the number of nodes in the data set. This then quantifies a similarity between the modules that are a factorised representation of the spike trains. In this work quantify the geodesic distance between the spatial modules W as a one-dimensional value. One interpretation of this measure since it is a similarity measure between the (factorised) spatial modules is as functional similarity, and to which extent this may have been captured in the model.

2.5.2 Generalised Linear Models

In order to assess the extent to which the spike statistics are encompassed by inferred models, a suitable baseline model is the generalised linear model (GLM) [79, 31], as this model is well-suited for capturing the statistics of spike trains.

For binned spike trains, the spike counts may be assumed to be Poisson distributed, and may thus be approximated by a Poisson PDF. Correspondingly, we may model each neuron as a Poisson PDF, with the rate parameter λ giving a probability for spiking, or an approximate number of spikes for an interval. In the GLM model, this is extended by linking the observed spike counts μ with the probability distribution's (here Poisson) corresponding density function - or rather the inverse of it, assuming that the observed variable is a linear combination of a set of parameters, which in turn lets us numerically fit the GLMs to a spike history. Further, we may also choose to couple each node's spike response with the other's activities, thus also assuming dependence on these. This may be written as,

$$y_t | X_t, w \sim Poiss(y_t; f(x_t^T w)\Delta), \tag{2.7}$$

i.e. the predictor y_t (via the Poisson PDF link function) given the observed activity x_t and a set of weighted couplings w between the other nodes. This allows us to write

models that may capture the spatiotemporal statistics that may be inferred using maximum likelihood estimation (numerically as indicated by Δ) with linked Poisson PDFs, by (a) link function f, given the observed spike histories x_t, weighted also on the other nodes' estimations w. This gives the log-likelihood of

$$logp(y|X,w) = \sum_{t=1}^{T} logp(y_t|x_t,w) = \sum_{t=1}^{T} (y_t log f(x_t^T w) - f(x_t^T w)\Delta), \tag{2.8}$$

where p is the probability of the predicted spike responses y given the observed spike responses x_t and the weights w, f is a link function, and Δ is the step size, which may be approximated using MAP estimates by numerical methods.

2.6 Model definitions

2.6.1 LIF

The neuron model we employed in this work is the leaky integrate-and-fire (LIF) model, as described in among other works [93], and also in 4. For the sake of consistency, we include a brief description here in this chapter. The LIF model may be formally outlined as,

$$\frac{dv}{dt} = \frac{E_L - v_t + R_I I_t}{\tau_m}, \tag{2.9}$$

where E_L is the rest potential, v_t is the membrane potential at time t, R_I is the membrane resistance, and I_t is the synaptic current. The modelled networks are non-transitively fully connected (i.e. all-to-all connected, but with no self-recurrent connections), with the weights normally distributed between $w \in [-1,1]$. The synapses are modelled as exponentially decaying post-synaptic currents, where a conductance variable g models the conductance for the neurons,

$$\frac{dg}{dt} = -\frac{g}{\tau_g}, \tag{2.10}$$

where the synaptic input current to a neuron j is modelled as $I_{syn,j} = \sum_i w_{i,j} I_{i,j}$.

While forward passes are done according to a spike-threshold function with non-linear resetting of the membrane potential upon spiking, the backward pass is made possible by separately defining a differentiable soft-threshold function over the membrane potential which is used solely for back-propagating the error gradients obtained

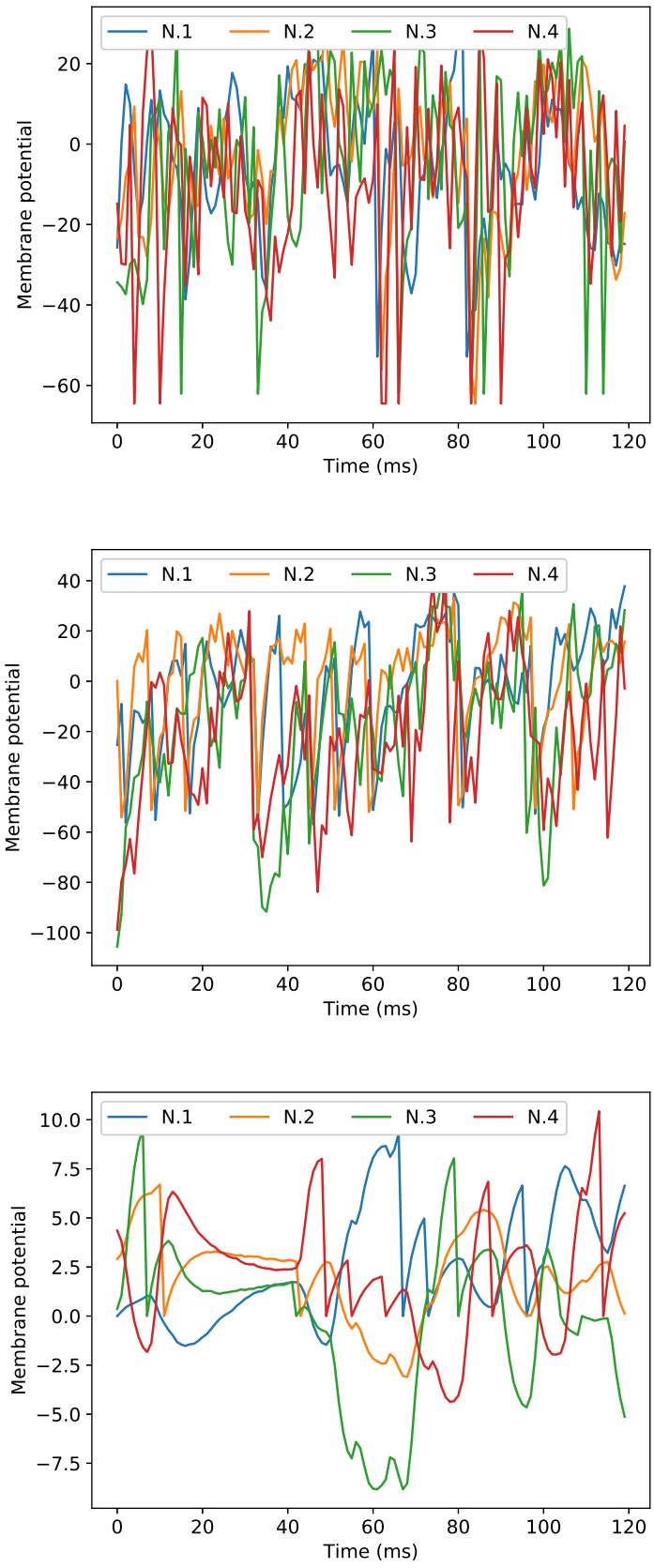

Figure 2.1: Sample membrane potentials for a LIF, GLIF, and SGIF model - top left, right, and bottom, respectively, illustrating the slightly different nature of the time series across the model systems. Each line represents the membrane potential of a single neuron.

by minimizing the loss functions. We use the commonly applied sigmoidal function for this purpose,

$$s_t(v) = \frac{1}{1+e^{-v_t}}, \tag{2.11}$$

where s_t denotes whether a neuron spikes at time t with a membrane potential v_t. This results in that optimization of defined loss metrics always back-propagates error gradients through the soft-thresholded membrane potential, in effect treating this as a continuous spike value. In other words, the distance metrics calculate the distance between a continuous spike train to a binary target spike train. Note that this also results in that sub-threshold values result in a spike signal during optimization.

All parameters were held as free parameters during optimization, resulting in 4 N-dimensional free parameters, and one N^2-dimensional parameter, where N is the number of neurons in the network.

2.6.2 Generalised leaky integrate-and-fire (GLIF)

Our GLIF implementation is based on the whitepaper of [1], and may be defined as

$$\frac{dv}{dt} = \frac{g(E_L - v) + R_I I_{syn}}{C_m}, \tag{2.12}$$

where v is the membrane potential, g the conductance, E_L the reversal potential, R_I membrane resistance, I_syn the total incoming synaptic current, and C_m the membrane capacitance. While the neuron model differential equations are linear, spiking is highly non-linear, and determined by the composite spike threshold $\theta_v + \theta_s$, whose differential equations are

$$\frac{d\theta_v}{dt} = a_v(v - E_L) - b_v(\theta_v - \theta_{inf}) \tag{2.13}$$

$$\frac{d\theta_s}{dt} = -b_s\theta_s \tag{2.14}$$

where θ_v is a membrane potential dependent spike threshold, and θ_s is an exponentially decaying threshold, a_v is an adaptation factor for θ_v, b_v is a voltage-induced threshold time constant, and b_s is a spike-induced threshold time constant. Introducing these thresholds makes spiking highly adaptive, which may mimic the temporal dynamics of the sodium-potassium pump, whilst remaining linear and differentiable.

Upon $v \geq \theta_v + \theta_s$ spiking occurs, which results in non-linear variable resetting, defined as,

$$v_{\text{reset}} = E_L + f_v(v - E_L) - \Delta_V \tag{2.15}$$

$$\theta_{s,\text{reset}} = (1 - b_s)\theta_s + \delta_{\theta_s} \tag{2.16}$$

where f_v is the pre-spike voltage fraction influence on the reset potential, and Δ_V is voltage addition following reset.

Each neuron is connected to every other neuron in the network, the weights being normally distributed with values $w \in [-1, 1]$. The synaptic currents decay with a factor f_I with an additive after-spike current I_A. This may be written as,

$$I_{syn} = \begin{cases} (1 - f_I)I_{syn} + I_A, & \text{for } v \geq \theta_v + \theta_s \\ (1 - f_I)I_{syn}, & \text{otherwise} \end{cases} \tag{2.17}$$

where the synaptic input current to a neuron j is modelled as $I_{syn,j} = \sum_i w_{i,j} I_{i,j}$.

GLIF neurons certainly allow for more biological realism in that they may exert most types of behaviour as outlined in the table in figure 2. in [55]. As such, their information processing is theoretically greater, and the types of spike patterns they may exert are richer than when compared to LIF SNNs.

2.6.3 Non-leaky integrate-and-fire (NLIF)

Interestingly, using sub-threshold synaptic currents as the spike signal, rather than a surrogate function over the membrane potential, results in not only a continuous optimisation signal, but also one that has a specific temporal signature, depending on neuronal excitation. Extending the work of [52], we implement first their non-leaky integrate-and-fire (NLIF) neuron model, with the aforementioned sub-threshold synapse model, which is defined using a gating-function which essentially results in a synaptic current when the potential is inside of an active zone.

$$\int s dt = \int g dv = 1, \tag{2.18}$$

where s is the synaptic currents, and g is the current gating function, and the synapse model is,

$$\tau_s \frac{ds}{dt} = -s + g\frac{dv}{dt} \tag{2.19}$$

Combined with a lower-dimensional target signal this enables perfectly learning to perform the task, albeit with the inferred model parameters depending largely on the initial configuration and random seed. We extend their work to also test their methodology on LIF SNNs (see chapter 5, and find that although this results in non-exact gradient calculation, the setup is sufficient for inferring models with near-perfect performance in reproducing the target signal for LIF SNNs, too.

2.6.4 Stochastic integrate-and-fire models (SGIF)

Lastly, to tie my work with a current state-of-the-art procedure, we have implemented the cortical microcolumn-inspired (based on [98], adapted to a lower set of neurons) stochastic integrate-and-fire (SGIF) population model of [92], and extended it such that it is compatible with in-place gradient based optimisation. We compare the results on both population- and neuron-level model fits with previously reported results, and also propose that the GBO procedure may be used to directly infer heterogeneous neuron-level models, and test this by fitting models to a full-size SNN SGIF target model. We also test fitting SGIF models to biological data, and evaluate the goodness of fit using loss metrics and geodesic similarities between the NMF modules.

While we refer the reader to [92] for the full model definition, we wish to outline the crucial parts to making the models in-place differentiable - not requiring using approximate Bayesian computation over model samples in order to estimate a posterior. The synapse model may be defined as,

$$\tau_s I_{syn}(t) = W_{syn}\varepsilon_s(t), \tag{2.20}$$

$$\tau_s \varepsilon_s(t) = (1+\tanh(t_s-\Delta_s))e^{-\frac{t_s-\Delta_s}{\tau_s}}, \tag{2.21}$$

where t_s is the time since the previous spike, Δ_s is the delay before synaptic transmission after spiking, W_{syn} are the synaptic weights, and ε_s is the synaptic kernel, or spike-transmission model. Note that the key difference between this formulation and the one in [92] is using the tanh-function to incorporate the transmission delay, rather than by using a Heaviside function Θ, as this allows us to differentiate the function, and thus backpropagate error gradients to calculate parameter gradients. Thus, we may use negative log likelihood estimation over the spike probabilities, i.e. maximising the likelihood that a set of spike probabilities produce the target spike trains when drawing

from a Bernoulli distribution with the probabilities, which is equivalent to minimising the negative log likelihood of the target spike train given simulated probabilities.

2.7 Loss metrics

Designing suitable loss metrics is one of the main challenges in applying gradient based optimisation to SNNs, as it is what defines the gradients, and thus the parameter space that we traverse during and throughout optimisation.

We performed optimization using (1) the firing rate distance, (2) the van Rossum distance, and (3) the negative log-likelihood assuming either a Bernoulli or Poisson PDF for the probabilistic models. Further, we performed initial testing using an additive combination of the van Rossum and firing rate distance, a Pearson correlation metric distance, the Fano Factor [110, 64], and the mean squared error, but found empirically that none of these were able to produce good results in terms of convergence or performance, and thus abandoned testing the metrics further for this project. Reasons for why these we unsuccessful may partly be explaiend by the reasons outliend in the van Rossum distance section below.

2.7.1 Firing rate distance

We define the firing rate distance as the Euclidean distance between each neuron's average firing rate for a given spike interval, which may be written as,

$$d_r(S_1, S_2) = \sum_i^N \frac{\sqrt{(s_1^i - s_2^i)^2}}{\Delta t}, \tag{2.22}$$

where s^i is the number of spikes for neuron i in a given interval Δt.

2.7.2 van Rossum distance

One thing is quantifying distances in terms of spike metrics. Another is designing a metric that is well-defined for gradient-descent. While the van Rossum metric describes the distance between two spike trains both in terms of rates and timing, due to its sensitivity to exact timing the gradient signal might be obscured for non-matching neurons, and more well-defined where the rates better match.

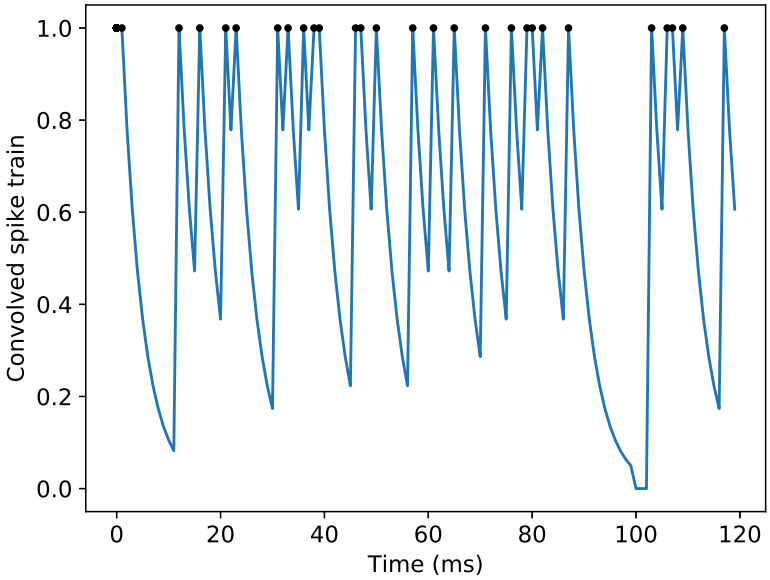

Figure 2.2: An illustration of the van Rossum convolution over a sample spike train.

The van Rossum distance [118] may be defined as the Euclidean distance between two spike trains where each spike is convolved with an exponential kernel, see figure 2.2, forward in time,

$$f_{conv}(t, t_{i,spike}) = e^{\frac{-\Delta t_i}{\tau_{\text{vr}}}} \tag{2.23}$$

where $\Delta t_i = (t - t_{i,spike})$, $t \geq t_{i,spike}$, with t being the current time, $t_{i,spike}$ the most recent time of spiking for neuron i, τ_{vr} a time constant (set to $\tau_{\text{vr}} \in \{10.0, 20.0, 100.0\}$ms in the experiments, and Δt_i the time since the neuron's last spike, $\Delta t_i = (t - t_{i,spike})$, the distance between two spike trains is then given by the Euclidean distance between two convolved spike trains,

$$d_v = E(S_1, S_2) = \sum_{t=0}^{t=N} \sqrt{(f_{conv}(S_1) - f_{conv}(S_2))^2} \tag{2.24}$$

Our initial hypothesis was that using the van rossum distance, we may better 'guide' the parameter inference search in the parameter traversal in the error landscape (then provided by the van Rossum distance metric), allowing gradient-based methods to converge more often, and to better solutions. The basis for the hypothesis was that using the van-rossum distance as a loss function for the spike trains, we attain a more continuous signal, since the discrete spike train is convolved with a time-based kernel, whic may make the gradients more informed via the loss function in larger search areas. However, our findings contrast this hypothesis, and we find "*wandering gradi-*

ents" for the van Rossum distance. Wandering gradients may be defined as gradients that point in no consistent direction, but results in that parameters "wander" arbitrarily during optimisation, in our work due to being in a region for which the loss is relatively constant and uniform. Upon closer inspection, the spike train data sets have a too high variability for an emphasis on the precise timing to improve optimisation performance - and we end up only obscuring the error signal further by using the van Rossum distance. I.e. it breaks down for the stochastic signals, and only works well for NLIF, or when we increase the time-constant such that the metric approaches a rate-based metric. And even then the firing rate distance might outperform the van Rossum distance metric for the surrogate gradient descent approaches.

2.7.3 Maximum Likelihood Estimation

By assuming either a Bernoulli or Poisson distribution for the observations, or spike trains, we can maximise the likelihood of producing that same data by using numerical methods. As usual, it is easier to work with a sum of products, rather than exponents, when doing numerical differentiation over a probability distribution to find the maximum likelihood estimate (MLE). Therefore we maximise the log-likelihood over the distribution, as it lets us work with a sum of products instead of a product of exponential functions, and is equivalent to maximising the likelihood itself. In fact, we also instead minimise the negative log-likelihood, which remains equivalent to maximising the likelihood, as this lends itself directly to optimisation as a loss metric and error signal which may be used for gradient-calculation.

$$\mathcal{L}(\theta|x) :\propto -\log \mathcal{L}(\theta|x) = -\log \prod_{n=1}^{N} pdf(\theta;X) = -\sum_{n=1}^{N} \log(pdf(\theta;X)) \tag{2.25}$$

Assuming a Bernoulli distribution, $pdf = Bernoulli(\theta;x)$, or assuming a Poisson distribution, $Poiss(\theta;x)$, these let us estimate the maximum likelihood over θ.

2.8 Summary

This chapter outlines SNN models and gradient based optimisation, and contextualises the model class and inference methodology combination in the field by drawing parallels to probabilistic models, RNNs, and machine learning. A few selected key works is briefly presented and discussed, before background material relevant for SNN model

evaluation is presented, along with a proposed baseline model; the GLM. Further, various SNN models are presented and formally defined, as well as the probabilistic SGIF model, which allows us to draw parallels to published work that used ABC and MCMC sampling for the SGIF SNN. Lastly, the two main loss metrics used in GBO for SNNs are presented, along with MLE for probabilistic models as is widely applied in ML. This provides the basis for implementation of the outlined models and procedures, which is presented throughout the remaining chapters of this thesis.

The goal in this thesis is to test the hypothesis that gradient-based optimisation may be used for spiking neural networks to capture higher-order statistics, including ensembles of activity on the network level, and to test whether ground-truth parameters can be retrieved using the approach. We implement a modular GBO framework in PyTorch, along with several different types of SNNs and loss metrics, which lets us test the aforementioned hypotheses, as well as to perform further implementation of the subthreshold currents synapse model of [52] in order to test whether ground-truth could not be retrieved due to unknown model input or training pattern complexity, as well as whether this more continuous model output implementation would provide a better basis for GBO. The latter turned out to be the case, with excellent convergence and task performance was observed, also for a leaky IF-implementation where exact gradients were thus not computed for BP in the model. However, ground-truth parameters were still not retrieved.

To sum up, some of the key contributions of this thesis includes:

- Showing that there is a near linear relationship between the recovery variables of the Izhikevich model and the resulting SNN firing rate
- Showing that LIF SNN inference using GBO is attainable, and that Adam may be a better optimisation strategy than SGD
- Finding that the van Rossum distance may make optimisation diverge in the case of using noisy, uknown input, due to the emphasis on precise timing, which is not meaningful in this case
- Demonstrating and providing a modern, modular and efficient implementation of SNN inference in PyTorch, which is modular wrt model, loss metric, and optimiser
- Proposing batching as a means of ameliorating the on-line constraint of SNNs due to parameter state dependence

- Coupling our SNN model inference work with published work on SGIF models and ABC
- Coupling SNN inference with SNPE, finding that posterior-estimation over all parameters might not be meaningful and not well capture the ground-truth means, due to the multiple solutions in parameter-space
- Finding that we cannot hope to retrieve the ground-truth parameters, and that this might not be a meaningful goal when aspiring to capture the target spike data
- Finding that we may capture higher-order statistics in terms of the NMF module similarity in our SNN models when using GBO, also beating or being on par with the proposed GLM baseline models
- Finding that even for the case of a non-noisy, fixed, lower-dimensional encoding-task, we converge towards completely different SNN model parameters, all with excellent task performance, even when employing continuous sub-threshold synaptic current models, strengthening the hypothesis that aiming to retrieve the ground-truth parameters might not be meaningful
- Finding that the methodology of [52] is applicable to leaky models, by extending their work on non-leaky integrate-and-fire models

Chapter 3

Complementary poster: Izhikevich recovery variable parameters, oscillations and resulting network rates

In our initial SNN research, we replicated [83] to show that the resulting rates could be formulated more or less as a linear function, resulting from sub-threshold oscillations, and that the model is highly sensitive to its initial parameter values. One of the motivations for doing this was that the model class may exert all neuronal modes of behaviour as observed in biology [55]. However, as hinted to above, this results in a highly irregular parameter landscape, only further complicated by the previously discussed stochasticity present in the model inference scheme that we are studying. To give an intuition about why this is, consider that the Izhikevich model is a 2-dimensional projection of a 4-dimensional ODE system - in fact based upon the original HH model [47]. As such, there are regions for which unstable, or even chaotic behaviour, may emerge - i.e. we end up with parameter regions for which the model is highly unrealistic, or not well-defined. Therefore, we limit our research to the previously outlined models, and only include parameter landscape plots illustrating the aforementioned, as well as our work on the effect of the recovery variables on resulting subthreshold oscillatory behaviour.

3.1 Poster: *The effect of the recovery variable parameters on oscillating Izhikevich networks*

The effect of the recovery variable parameters on oscillating Izhikevich networks

Synchrony in a Strongly Connected Spiking Network

William Peer Berg & Arno Onken
University of Edinburgh,
the Institute for Adaptive and Neural Computation
william.berg@ed.ac.uk

The theory of neuronal group selection (TNGS) posits neuronal groups as functional units within the brain. Since the proposal of TNGS, studies have corroborated the existence of such units. With a myriad of different brain areas and rhythms, there are analogously various mechanisms for emergent synchrony that remain to be uncovered.

This work focuses on a subset of neuronal behaviours that occur within biology, by analysing a highly connected network of inhibitory and excitatory neurons, with a parallel to that of Central Pattern Generators (CPGs) in that the population bursts rhythmically. By extending the work of [2], and using Izhikevich neurons, we attain a computationally efficient implementation which captures the emergence of synchronised population bursting. We uncover and elaborate on why this synchronised bursting occurs in this specific model, and outline further interesting parallels.

Main Objectives

1. Test if population bursting depends on the single neuron sub-threshold oscillations
2. Test if single neuron firing can predict network bursting
3. Analyse the effect of the recovery variable parameter values on the neuron, and network

Materials and Methods

We extended the model of Oliveira et al. (2019) [2], by performing further exploration of the parameter-space, and a comparative analysis of the single neuron.

This led to observations which enabled us to explain the relationship between the model parameters and emergent firing rates. The Izhikevich neuron model employed is defined by,

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I_{\text{syn}} \quad (1) \qquad \frac{du}{dt} = a(bv - u) \quad (2)$$

$$\text{and if } v \geq 30\,\text{mV}, \text{ then } v \leftarrow c \text{ and } u \leftarrow u + d.$$

This work is primarily focused on the effect of the parameters a and b in regularly spiking networks, with two classes of neurons having $(c, d) = (-65, 8)$, and $(-65, 2)$, respectively.

The synaptic model is defined by a simplified differential equation,

$$\frac{dg}{dt} = -\frac{g}{\tau_g} \quad (3) \qquad I_{\text{syn}} = \sum_{i=1}^{N} g_i w \quad (4)$$

where g is reset to 1 upon spiking, and otherwise decays exponentially, as defined in Eq. 3. I_{syn} is modelled per neuron as the sum over each synaptic conductance times a synaptic weight constant $w \in [10, 20]$ (Eq. 4). At each time-step of the model simulation, a random neuron is stimulated by $I_{\text{external}} = 100\,\mu\text{A}$.

Results

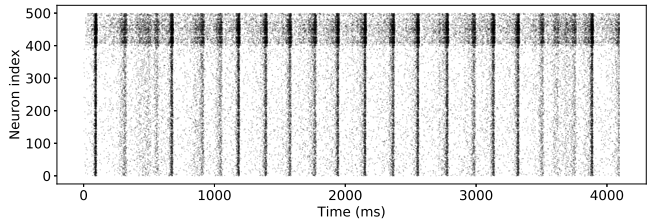

Figure 1: Raster plot for a network model consisting of $n = 500$ neurons, with parameter values $a = 0.005, b = 0.42$, a weight of $w = 11$, and $\tau_n = 5.0\,\text{ms}$.

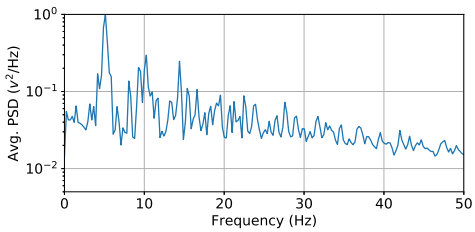

Figure 2: The power spectrum density reveals $\approx 4.15\,\text{Hz}$ as the most prominent burst rate.

- Inhibitory neurons are close to their bifurcation point; a slight stimulus leads to spiking
- Excitatory neurons drive synchronisation
- a and b determine the neuronal sub-threshold oscillation rate
- Network burst rate depends on the sub-threshold oscillation rate, and parameters a and b
- Population burst rate may be predicted by single excitatory neuron firing rate
- There is a minor effect of inhibitory neurons' negative stimulus on spike frequency
- Inhibitory neurons generally burst more due to: (1) a shorter refractory period (a higher a-value), and (2) a lower after-spike reset for the recovery variable (lower d)
- Bursting occurs due to the large input current, which is a result of the dense connectivity
- For lower weights w, I_{syn} is lower, and the network desynchronises

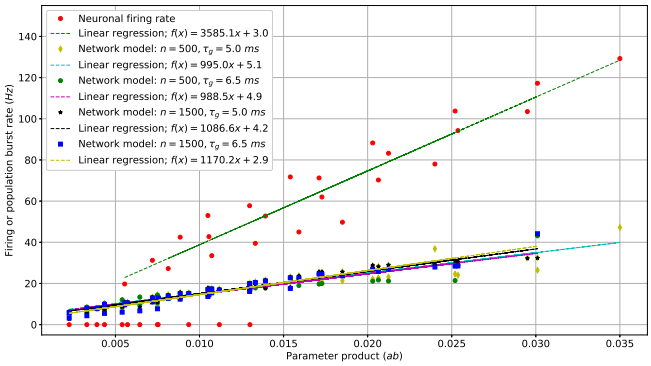

Figure 3: Neuronal sub-threshold oscillation rate and synchronised population burst rate reveal a linear relationship between ab and the burst rates for converged (synchronous) models. Data points for non-synchronous models are not included. Convergence to synchronous bursting is less likely for higher higher parameter-products and firing rates.

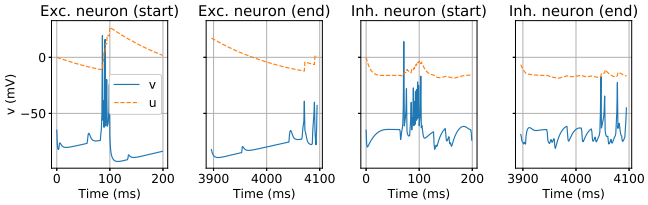

Figure 4: Voltage traces of two randomly chosen neurons for two time intervals. Inhibitory neurons fire longer bursts, due to their parametrisation of $(a, d) = (0.1, 2)$.

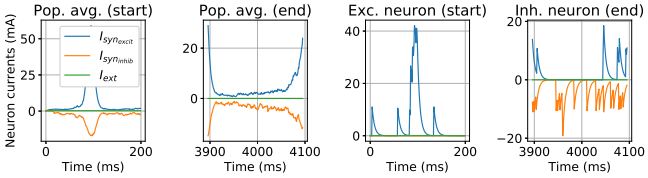

Figure 5: Currents at two time intervals for the population average, and two random neurons. Excitatory and inhibitory currents mirror one another on average due to synchronicity.

- With weights $w \in [10, 20]$, incrementing of u by $u \leftarrow (u + d)$ does not induce a refractory period for the neuron, as the signal I is still greater than the subtracted value u in $\frac{dv}{dt}$. This leads to a series of consecutive spikes and increments of u by d, or a burst, eventually inducing a refractory period.
- Because the inhibitory neurons have a slightly higher time scale a for the recovery variable u, they may impose a slightly higher negative current on the population, inducing a slightly longer build-up of the membrane potential v.
- Note that weights $w \geq 1$ is not necessarily biologically implausible, as they also incorporate a constant for the difference between the reverse and membrane potential (typically denoted $(E_{syn} - V_{post})$).

Forthcoming Research

(1) Using spiking Izhikevich networks to capture emergent neural assemblies as observed in vivo within the PPT/LDT (brainstem) of mice during wakefulness, REM and NREM sleep. **(2)** Analysing phase synchronisation in weakly coupled spiking networks with different rhythms. **(3)** Looking at the effect of brain topology on spiking Izhikevich network behaviour.

References

[1] Eugene M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572, nov 2003.

[2] Lucas D.R. Oliveira, Rogerio M. Gomes, Bruno A. Santos, and Henrique E. Borges. Effects of the parameters on the oscillation frequency of Izhikevich spiking neural networks. *Neurocomputing*, feb 2019.

Acknowledgements

We would like to express our gratitude towards Dr. Nina Kudryashova for her invaluable input, particularly on interpretation of neurophysiological parallels.

In sum, although the Izhikevich model is capable of almost all types of behaviour as seen in biology with only 2-dimensional ODEs per neuron, it comes at the cost of also being able to exhibit highly unrealistic, or chaotic behaviour, for which GBO would diverge, as gradient would be completely un-informative. Further, there is a strong dependence on a sub-set of the model parameters, which also poses an issue for GBO; these are the only parameters likely to be inferred using GBO unless specific design and care is taken to calibrate the other parameters, such as for instance sequentially after initial inference of the others - however, this would then render the other inferred values completely dependent on the variables inferred prior in the sequence. In some preliminary experiments we confirmed empirically that divergence indeed did occur due to chaotic behaviour, which was not easily avoided due to several regions of parameter hyperspace for which such behaviour can occur. As such, Izhikevich SNNs are not used further in thiswork where we study SNN inference using GBO.

Chapter 4

Gradient-descent based LIF SNN inference

4.1 Poster: *Spiking Network Inference Using Gradient Based Optimisation*

Before beginning on the next chapter, which describes in more detail the state-of-the-art of SNN inference using both gradient-based optimisation and to some extent approximate Bayesian computation by simulation-based inference, we would like to present the work that we have done specifically for and constrained to leaky integrate-and-fire neurons using gradient-descent for parameter-inference, before expanding on the methodology and applying it to a wider range of model types and classes in the following chapter 5. The poster contains the essentials pertaining to the post-inference NMF analysis, the van Rossum distance metric, as well as single-neuron level rate analysis, along with inferred kernel density estimates over the parameters across experiments.

These experiments seemed highly promising with regards to ground-truth parameter retrieval - however, they are in part due to smaller initial parameter intervals, and the fact that the LIF-model is relatively low-dimensional. This becomes clearer in the more extensive report on LIF inference when considering the relative (normalised) parameter distance before and after training, which was done following the work presented in the poster. It is nevertheless interesting that the inferred kernel density estimates are somewhat centred around the true means for several of the LIF-parameters. While we hypothesise that this won't hold for the more high-dimensional models, such as for the

stochastic general integrate-and-fire (SGIF) [92] and generalised leaky integrate-and-fire (GLIF) models [1], it is certainly something that it would be interesting to pursue in future research.

Spiking network inference using a modular gradient-based optimisation framework

William Peer Berg (william.berg@ed.ac.uk) and Arno Onken – University of Edinburgh

Inferring biologically meaningful spiking neural network models (SNNs) using only spike train data presents several significant challenges, including traversing a high-dimensional parameter state space, as well as using spike trains as the only signal. We address these issues by implementing a modular SNN inference framework in Pytorch using gradient-based optimisation, inspired by René et al. (2020)[1]. We generate target data spike trains by hand-engineering a biologically plausible spiking sleep regulation model, and show that we may recover the model's parameters, even under strong Poisson input noise, starting from different parameter configurations. Further, inferred models have neuronal ensembles that are similar to the original model's, and also similar firing rates, as well as spike patterns.

This work addresses the issue of scarcity of computational models in the field by providing a basis for a general modular spiking model inference framework using only spike trains as the target signal.

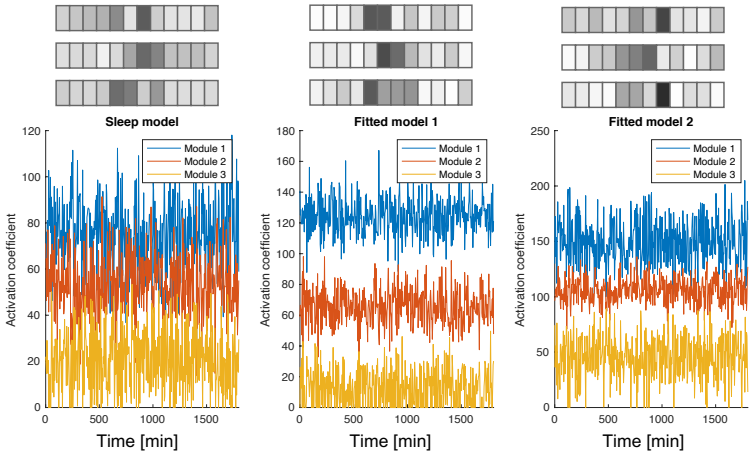

Figure 1: (Top) factorised spike train modules, and (bottom) their corresponding activation coefficients.

We hand-engineer two spiking models inspired by the rate based sleep model of Héricé et al. (2020)[2], attaining models with spike features that are qualitatively similar to biological spike trains. To evaluate and compare the inferred models with the sleep models, we use **non-negative matrix factorisation** (NMF) on the spike trains. This is an external statistical analysis, which yields a parts-based representation naturally due to its non-negativity constraint, leading to neuronal ensemble identification, which may be seen in figure 1.

Formally, NMF may be outlined as

$$X = WH$$

where X is an $n \times m$ spike train matrix, W is an $n \times k$ activation coefficient matrix, and H a $k \times m$ matrix of the modules.

This work is mainly focussed on fitting **leaky integrate-and-fire** (LIF) models, which may be outlined as follows,

$$\frac{dv}{dt} = \frac{v_{rest} - v + I}{\tau_m}$$

where v is the membrane potential, v_{rest} is the rest potential, I is the synaptic current, and τ_m is a membrane time constant. Additionally, we also implement and test Izhikevich neurons. In both cases, we used an exponentially decaying **conductance based synapse model**, which may be written as

$$I = Wg + I_{ext}, \qquad dg = -\frac{g}{\tau_g}$$

where W is a weight matrix, g is synaptic conductance I_{ext} is extenal input (e.g. Poisson input), and τ_g is a conductance decay time constant.

The **Van Rossum distance** [3] D_R is the Euclidean distance between two transformed spike trains convolved with an exponential kernel. This may be written as

$$D_R(\tau_R) = \sqrt{\left(\frac{1}{\tau_R}\int_0^\infty [\tilde{x}(t) - \tilde{y}(t)]^2\right)}, \qquad f(t) = e^{-\frac{t}{\tau_R}}, t > 0$$

where τ_R is a time constant, and $\tilde{x}$ and $\tilde{y}$ are the spike trains x and y convolved with f, transforming each spike to an exponentially decaying signal.

Goal: We demonstrate retrieval of model parameters close and far from the intial setting.

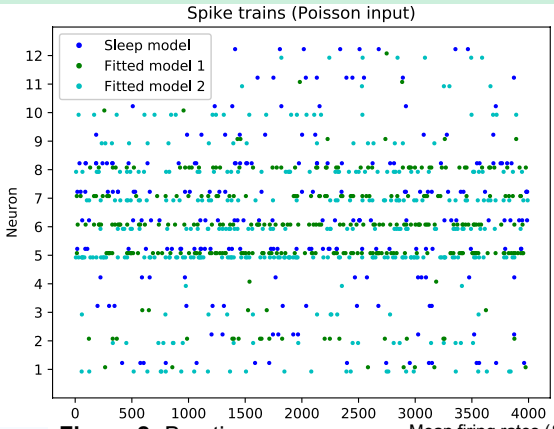

Figure 2: Bursting, sparse and regular firing.

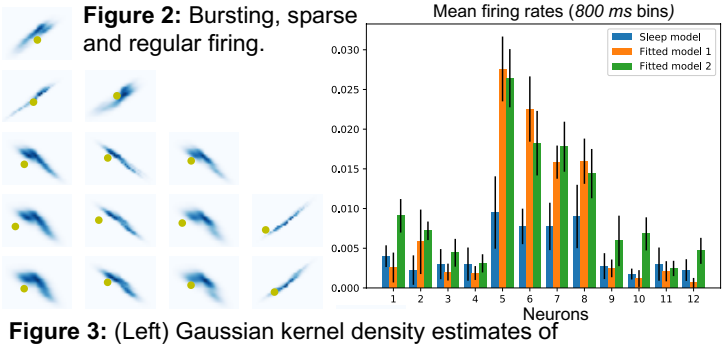

Figure 3: (Left) Gaussian kernel density estimates of fitted model parameters across training iteration (blue), with the true values (yellow dots).

In sum, spiking model inference using spike train data may allow inferring models capturing biologically realistc spike patterns. This provides a basis for assessing the dynamics of an area by model analysis, which when coupled with NMF allows comparable neuronal ensemble identification.

Future work includes addressing limitations of gradient-based methods in parameter space exploration. This will include looking into using adaptive optimisation methods for each parameter, which might also reduce the required overhead and framework complexity.

[1] René, et al. (2020). Inference of a Mesoscopic Population Model from Population Spike Trains. *Neural Computation*, 1–51.
[2] Héricé & Sakata. (2019). Pathway-Dependent Regulation of Sleep Dynamics in a Network Model of the Sleep–Wake Cycle. *Frontiers in Neuroscience*, *13*, 1380.
[3] Van Rossum. (2001). A novel spike distance. *Neural Computation*, *13*(4), 751–763.

4.2 Report/paper draft: *Parallel spiking neural network parameter inference using gradient based optimization*

The report addresses different combinations of the van Rossum distance and firing rate distance metrics, as well as different optimisation schemes. These did not vary significantly for the LIF model, at least not when considering the relative (normalised) parameter distance and geodesic NMF model similarities between stochastic gradient descent (SGD) and adaptive moment estimation (Adam) optimisation, nor between the van Rossum distance and firing rate distance. This may suggest that the lower dimensionality of the LIF model results in that all setups are rendered ”good enough” for the model inference. As we shall see in the following chapter, specifically in the subsection on error and parameter landscapes 5.2.1, the reason for this is that the loss metrics result in regions in the parameter error landscapes for which the error and thus gradient is unchanged. However, during our work in the next chapters, we observed empirically in initial experiments that Adam was significantly better in terms of both convergence and the number of training iterations for the more complex models (SGIF and GLIF), as well as in the gated subthreshold synaptic current synapse models in chapter 7 [52]. One possible explanation for this is simply the same as in the ML domain; the adaptive moment allows for an adaptive response to the gradients, and is lowered when the gradient rate of change does not. In our case, i.e. for LIF models, this does in effect only gain the benefit of being able to converge without too much calibration of the learning rate, as this is adapted throughout inference. In effect this may be seen as a type of simulated annealing, only relative to each parameter and their gradients.

Parallel spiking neural network parameter inference using gradient based optimization

William Peer Berg [1] **Arno Onken** [1]

Abstract

A common goal in computational neuroscience is to model biologically observed neural activity in order to capture dynamics relating to computation and functional organization. Despite the recent success of gradient based optimization in deep learning, application to spiking models has been limited due to the non-differentiability and complexity of the systems. Using leaky integrate-and-fire neurons, we transform neurons' spike output into continuous, differentiable signals by using soft thresholding, and devise a methodology for spiking neural network (SNN) model parameter inference using gradient based optimization, fitting to spike trains as the target data. To assess goodness of fits, we compare first and second-order statistics, as well as non-negative matrix factorization modules, which are independent to the optimization procedure. The latter is calculated by evaluating the geodesic distance between modules of fitted models and those of the original data sets. Our results show that spike patterns may be captured through gradient based optimization, and further that higher-order spike statistics may be replicated by fitted models. This approach (1) demonstrates a methodology for SNN inference using gradient based optimization and a common data set type, (2) compares two common optimizers for the outlined experiments, and (3) provides a framework for accelerating SNN model inference.

1. Introduction

Fitting single neuron models to patch-clamp recordings has been a common methodology for constructing computational models ever since the seminal work of (Hodgkin & Huxley, 1952), which also happened to capture temporal dynamics relating to the sodium-potassium pump before its discovery later that decade (Skou, 1957). Advancements in recording techniques now make it possible to record from large populations of neurons simultaneously (Jun et al., 2017). These recordings may be used to form spike train data sets indicating which neuron spiked at what time. These signals may be analysed to attain information about the recorded area. However, the use of such data in spiking neural network (SNN) model inference has been limited in part by the non-differentiability of SNN systems, despite the recent success of gradient based methods in deep learning (Huh & Sejnowski, 2017; Tavanaei et al., 2019). Modelling more biologically realistic and thus complex spiking networks, let alone for larger populations of neurons, would thus require methodological advancements over current approaches to make inference tractable.

Several recent studies have tried to address this problem. (Rule et al., 2019) note in their work that recent advances in recording techniques may in conjunction with methodological advances allow for new theoretical connections. Further, they show that intrinsic neuronal states in neural field models may be inferred based on spike train data. (René et al., 2020) demonstrate Bayesian inference for sequential parameter estimation of both leaky integrate-and-fire (LIF) and generalized leaky integrate-and-fire models (Allen Institute for Brain Science, 2017) using spike train data. Nevertheless, these works employ computationally costly methods and do not make use of the work horse for deep learning optimization, which is gradient based.

In this work, we explore the utility of gradient based optimization for parallel parameter inference from spike train data, where parameter inference is defiend as maximising the likelihood that a spike train was produced by the current model by adjusting its parameter values corresponding to minimising the error gradients defined by the studied loss metrics. To this end we implement a modular gradient based optimization framework on top of Pytorch, utilizing its autograd-feature to minimize the loss defined by the outlined distance metrics. We implement leaky integrate-and-fire (LIF) neurons, and crucially define spiking via a soft-threshold function, such that the system is differentiable. For the loss functions, we explore three options: (1) the firing rate distance as a baseline loss function, (2) the van Rossum distance (Van Rossum, 2001) to capture both temporal and spatial spike aspects (relating to both frequency and timing of spiking), and (3) the firing rate distance and the van Rossum distance in conjunction.

We investigate whether the inferred models reproduce spike trains capturing properties of the original data set, and address the questions of tractability and computational cost relating to parallel parameter optimization. In order to assess the performance of our approach, we perform non-negative matrix factorization (NMF) (Seung & Lee, 1999), and assess the geodesic distance between the modules of spike trains produced by the inferred and generative models. NMF has the advantage of naturally resulting in a parts-based representation due to its non-negativity constraint, which has been found to be suitable for natural data and spike train analysis (Seung & Lee, 1999). Further, this functional network characteristic is independent of the optimization-procedure, making it a suitable additional metric to assess the outlined methodology.

Overall, our findings suggest that spike train data may be used for model inference, and that inferred models can reproduce functional network characteristics of the original data. The distance between the NMF modules of the inferred model spike trains and the target spike trains is reduced substantially throughout optimization, with the closest fits falling very close to the ground-truth. While convergence toward true model parameter values cannot be guaranteed, we also observe that the most prominent parameters fall within close proximity of the ground-truth values for the closest model fits. We also observe that fitted model spike trains become more correlated with the target spike trains in all experiment configurations.

2. Methods

The goal in this work is to demonstrate that successful machine learning methods of gradient based optimization may also be applied in spiking network model inference using leaky integrate-and-fire neuron models. Further, we aim to assess quantitatively (1) to what extent ground-truth parameters are inferred, (2) to what extent dependent functional network characteristics of the spike trains are captured, and (3) to what extent independent functional ensembles are captured by and emerge from the inferred models. The neuron model we employed in this work is the leaky integrate-and-fire (LIF) model, as described in among other works (Rolls & Treves, 1998),

$$\frac{dv}{dt} = \frac{E_L - v_t + R_I I_t}{\tau_m}, \tag{1}$$

where E_L is the rest potential, v_t is the membrane potential at time t, R_I is the membrane resistance, and I_t is the synaptic current. The modelled networks are non-transitively fully connected, with the weights normally distributed between $w \in [-1, 1]$. The synapses are modelled as exponentially decaying post-synaptic currents, where a conductance variable g models the conductance for the neurons,

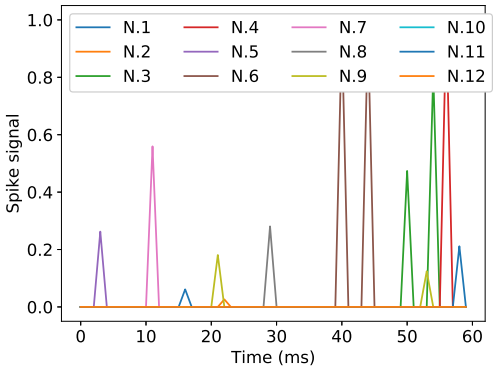

Figure 1. The soft-thresholded spike signal $s_t(v)$ for 60ms backward passes during model simulation for 12 neurons.

$$\frac{dg}{dt} = -\frac{g}{\tau_g}, \tag{2}$$

where the synaptic input current to a neuron j is modelled as $I_{syn,j} = \sum_i w_{i,j} I_{i,j}$.

While forward passes are done according to a spike-threshold function with non-linear resetting of the membrane potential upon spiking, the backward pass is made possible by separately defining a differentiable soft-threshold function over the membrane potential which is used solely for back-propagating the error gradients obtained by minimizing the loss functions. We use the commonly applied sigmoidal function for this purpose,

$$s_t(v) = \frac{1}{1 + e^{-(v_t))}}, \tag{3}$$

where s_t denotes whether a neuron spikes at time t with a membrane potential v_t. This results in that optimization of defined loss metrics always back-propagates error gradients through the soft-thresholded membrane potential, in effect treating this as a continuous spike value. In other words, the distance metrics calculate the distance between a continuous spike train to a binary target spike train. Note that this also results in that sub-threshold values result in a spike signal during optimization.

All parameters were held as free parameters during optimization, resulting in 4 N-dimensional free parameters, and one N^2-dimensional parameter, where N is the number of neurons in the network.

In addition to the previously mentioned parameters, a Poisson input perturbation rate was also optimized as a free scalar variable, r_p. This is used to generate a Poisson input perturbation to the model for each time-step, drawn from a Poisson distribution with a rate r_p. As such, the input current model may be written as,

$$I(t) = I_S + I_{\text{ext}}, \tag{4}$$

where $I_S^j = \sum_i^N w_{ij} g_i$, and $I_{\text{ext}}^j \sim P\{k; r, t\} = \frac{(r_p t)^k e^{-rt}}{k!}$, where t is the time interval, and k is the number of spikes, $P\{k; r, t\}$ denoting drawing k spikes (pseudo-)randomly from the Poisson distribution.

2.1. Gradient based optimization

We implement a LIF SNN optimization framework on top of Pytorch (Paszke et al., 2019) and utilize Pytorch's autograd-feature for backpropagation of error gradients. The architecture studied consists of a mixture of excitatory and inhibitory neurons, with a fully, non-transitively connected network of network size $N = 12$, with 8 excitatory and 4 inhibitory neurons.

To perform optimization, an set of initial model parameter values is drawn uniformly from pre-defined intervals. These intervals are constrained both to meaningful parameter ranges, and also to ensure non-silent models upon initialization. These models are then perturbed with Poisson input, drawn pseudo-randomly from a Poisson distribution given by the current rate parameter (initially $r_p = 10\text{Hz}$). See table 1 under section 4for more details. The model output is then compared with the output of a target data, and the loss is calculated between the model and target output spike trains by using the loss functions defined in the following sub-section. Training is then done for a static number of 20 training iterations. Note, however, that this could easily be modified to depend upon some convergence criterion. However, for comparability, we chose to keep this number constant.

2.1.1. Loss metrics

We performed optimization using (1) the firing rate distance, (2) the van Rossum distance, (3) an additive combination of the van Rossum and firing rate distance, and (4) by an adaptive combination of the two loss functions, emphasising mainly the firing rate in earlier training iterations, and conversely mainly the van Rossum distance in later training iterations. We define the firing rate distance as the Euclidean distance between each neuron's average firing rate for a given spike interval, which may be written as,

$$d_r(S_1, S_2) = \sum_i^N \frac{\sqrt{(s_1^i - s_2^i)^2}}{\Delta t}, \tag{5}$$

where s^i is the number of spikes for neuron i in a given interval Δt.

The van Rossum distance (Van Rossum, 2001) may be defined as the Euclidean distance between two spike trains where each spike is convolved with an exponential kernel forward in time,

$$f_{conv}(t, t_{i,spike}) = e^{\frac{-\Delta t_i}{\tau_{\text{vr}}}} \tag{6}$$

where $\Delta t_i = (t - t_{i,spike}),\ t \geq t_{i,spike}$, with t being the current time, $t_{i,spike}$ the most recent time of spiking for neuron i, τ_{vr} a time constant (set to $\tau_{\text{vr}} \in \{10.0, 20.0, 100.0\}\text{ms}$ in the experiments, and Δt_i the time since the neuron's last spike, $\Delta t_i = (t - t_{i,spike})$, the distance between two spike trains is then given by the Euclidean distance between two convolved spike trains,

$$d_v = E(S_1, S_2) = \sum_{t=0}^{t=N} \sqrt{(f_{conv}(S_1) - f_{conv}(S_2))^2} \tag{7}$$

and lastly, we also define two linear combinations of these distance metrics as loss functions, either simply as a static combination, $d_{\text{linear}} = 0.9 d_r + 0.1 d_v$, or as linearly decreasing or increasing for d_r and d_v, respectively,

$$d_A = (1 - 0.9T) d_r + (0.1 + 0.9T) d_v, \tag{8}$$

where $T = \frac{t}{t_{\text{max}}} \in [0, 1]$.

2.2. Assessment of functional characteristics

In order to assess functional organization in the neural data we employ non-negative matrix factorization (NMF) (Seung & Lee, 1999) to infer ensembles of coactive neurons. This factorization may be written as,

$$V \approx WH, \tag{9}$$

where the original data set V of dimension $n \times t$ is factorised into two matrices of dimensionality $m \times n$ and $m \times t$. What makes this dimensionality reduction method particularly suited for spike train data is its non-negativity constraint, which naturally leads to a parts-based representation.

While NMF may be used to discover spatial and temporal firing patterns in spike data, the geodesic similarity measure between the modules may be used to assess how well these features were recovered in a fitted model. The geodesic distance o between two matrices A and B of dimensionality $m \times n$ may be written as,

$$o(A, B) = (1 - \frac{2}{\pi} \cos^{-1}(A^T B)), \tag{10}$$

where m is the number of modules, and n is the number of nodes in the data set. We also calculate the spike correlation

between the generative (target) model and the fitted model by binning spikes into 20ms bins and then calculating the cross-correlation between the binned spike counts for in total $4\ s$ of data from each model, generated by using the same random seed as in the experiment for Poisson input generation. The results for the different optimization setups are shown in section 4.

Lastly, we plot inferred Gaussian kernel density estimates (KDEs) for parameter marginals (Rosenblatt, 1956; Parzen, 1962), and show a sample plot for this in figure 10. Largely parameters fall outside of the kernel-borders - however, some parameters of converged models fall closer to the true values. This cannot, however, be guaranteed, as can be seen in figures 9, and 10.

2.3. Synthetic data set

We simulate data sets with known ground truth to test the (SNN) inference algorithms on. The generative model was designed to have three distinct populations of neurons, with each population exhibiting fairly stable, yet distinct spike patterns. To generate the target data, we perturbed the hand-engineered LIF model with Poisson input with rates of 10Hz, producing target spike train data. To make our results more robust, however, we initialized four different models by normally distributing the parameter values around the pre-defined values, setting a distinct random seed for each initialization.

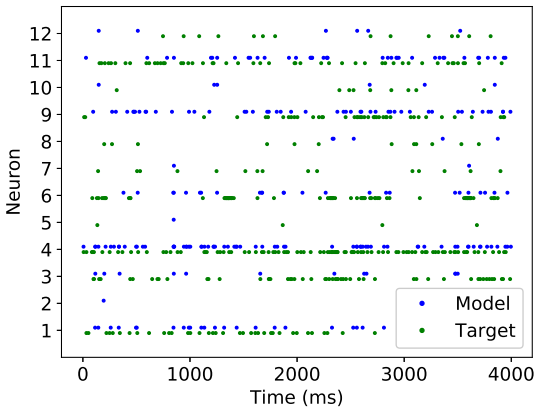

Figure 2. Spike trains after optimization for 20 training iterations, each over 10 batches of 400ms. Here the Poisson input to the generative and fitted models are drawn pseudo-randomly from Poisson distributions their respective rates, but with the same random seed.

2.4. Computational complexity

This research explores parallel parameter inference in order to reduce computational complexity to the extent that this is possible. One algorithmic bottleneck is the one imposed by the sequentiality associated with the temporal nature of spiking models. i.e. the temporal dependence its spike history and state, which mean that they have to be simulated in time. One may, however, relax simulation to intervals of simulation, each of which may be run in parallel. Parameter updates throughout training still poses a checkpoint where synchronization has to occur. This means that parallel parameter optimization would greatly reduce the computational cost, compared to other recent works where optimization is done sequentially (René et al., 2020), which with gradient-based methods quickly becomes intractable.

In our implementation we divide each training interval into a set of batches B that we simulate in parallel, and also optimize for all parameters in parallel. Training over intervals should result in more stable gradients, and may also reduce the number of training iterations required for convergence due to smoothing the parameter landscape **??**.

The complexity of simulation scales with the time frame simulated Δt_B for each batch, and the model complexity itself, which is $N^2 + PN$, where P is the number of N-dimensional parameters, and the weights ω are N^2-dimensional. This makes simulation $O(C_t \Delta t_B (N^2 + NP))$ where C_t is the number of training iterations, which has a lower bound of the reduced form $\Omega(N^2)$. Optimization additionally requires back-propagation through the system, including calculation of the parameter updates for each parameter. Yet, the lower bound is that of the model dimensionality, which is $\Theta(N^2 + NP)$ per time-step, and due to the sequential constraint of SNN simulation thus $\Theta(2C_t t_B(N^2 + NP)$, where C_t is the number of training iterations, and t_B is the length of each training batch. The temporal constraints relating to spiking neural network simulations make exploring parallel parameter optimization the least costly approach, but yet quadratic in complexity, whereas sequential optimisation would be exponential.

3. Related work

(Grün, 2010) discuss defining spike metrics to quantify the distance between two spike trains with regards to data analysis. More recently, (Huh & Sejnowski, 2017) applied gradient descent on a network of single-state neurons by reformulating the gate function into a continuous gate function, thus introducing a novel model that may be successfully used in gradient based optimization. In this work, however, we transform the more commonly studied leaky integrate-and-fire model which in contrast also has non-linear reset rules, into a differentiable system. This allows employing successful machine learning approaches for LIF SNN model inference.

Another related piece of work looking at parameter inference in SNNs is the work of (René et al., 2020), where the authors study using Bayesian inference for parameter estimation. This has the advantage of providing a confi-

dence intervals for the estimated parameters. However, the outlined approach requires estimation of a population-level model for Markov Chain Monte Carlo sampling for sequential neuron-level model parameter estimation, resulting in a computationally costly procedure. In this work, the aim is to address employing ML techniques for direct, parallel parameter inference of neuron-level models.

(Rule et al., 2019) perform intrinsic state inference of classical neural field models using spike train data by developing a moment closure based method. In doing so they connect single neuron dynamics with population level dynamics by using spike train data. In our work, we wish to encompass functional level behavior by more direct inference based on spike train data by optimizing spiking network models.

Optimization of deeper networks has only recently become successful, previously hindered by the non-linearities being more pronounced by deeper networks (Bengio et al., 2013). Similarly, as noted by (Bengio et al., 2013), these problems may apply to recurrent neural networks when unfolded through time, including complex recurrent networks such as spiking neural networks. In this work we study spiking neural networks that are recurrently, albeit not transitively, connected. As such, the issues relating to RNNs should also be considered in this work. We note that the leakage term of the neuron model may reduce vanishing gradients when simulating over periods of time, and also that different temporal properties of the nuclei may result in more separable patterns. Further, we also try to use adaptive momenta for the learning rate per parameter, and test whether this may improve the model fit. Overall, we utilize the advances of deep learning which has made optimization of deeper networks possible, and study whether this may improve, and let alone make SNN inference possible.

Generalized linear models have also been widely used to study spike train data due to its linear separability and convexity, guaranteeing convergence towards a global optimum for the defined filter functions given a well-defined data set, and for its interpretability by fitted filter inspection, which may reveal strong signatures of spike history dependency and/or neuronal couplings (Pillow et al., 2008; Nelder & Wedderburn, 1972). We also fit GLMs to the synthetic spike data in order to establish a baseline for our GD optimized LIF models.

4. Experiments

We perform the experiments for four different seeds and perturbations of carefully designed generative models, resulting in fitting 20 LIF models for each carefully designed generative model. The generative model was designed to mimic biological data, and to meet a set of criteria, including that it should contain three different sub-populations

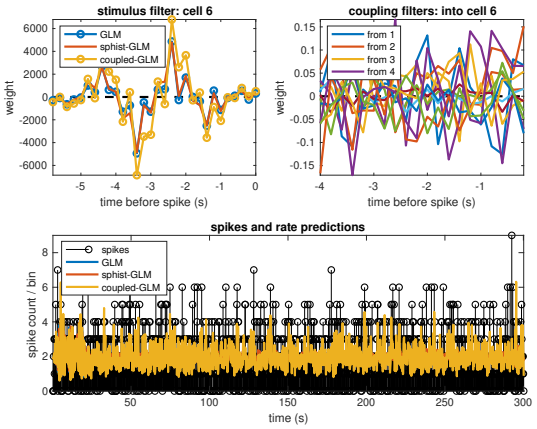

Figure 3. Stimulus and coupling filters (top) for GLMs fitted using a custom Matlab script, with filters for the spike history and neuron couplings, as well as predicted firing rates (bottom) per bin (400ms) when using the GLMs for prediction using Poisson noise ($\lambda = 10$) as stimulus.

with different properties, each of which should produce different spike patterns, all within realistic frequency ranges. For each population, we implemented different temporal qualities by adjusting parameters including membrane time constants, membrane resistances, synaptic current time constants, weights (with one having negative weights). For data set generation we perturbed all of the the parameter values slightly (with a standard deviation of 2.5 %), and instantiated the models by using random seeds 0 through 3 for each uniform pseudo-random model initialization, along with $\{\overline{\omega}, \omega_{\text{std}}\} = \{0.3,\ 0.2\}$ for the weights, and $r_p = 10\text{Hz}$ for the Poisson rate. Given that we consider the above criteria to be met by qualitatively studying the spike trains of our generative models, and by considering fitted generalized linear model (GLM) filters, we hypothesise that the methodology results are to generalizable to application to biological data.

We also performed a comparison with GLMs (Nelder & Wedderburn, 1972) by fitting these to the generative model spike trains by maximising the log-likelihood. We then verified that the spike and neuronal coupling filters of the fitted GLM looked comparable to that of when fitting to real data (we observed some cyclic and sinusoidal spike history filters, and some positive or negative cyclic bumps for neuron coupling filters). We also did the same test on fitted LIF models. Further, we predicted spike trains by using the GLM models, and performed NMF on the predicted spike trains. When using this as a baseline, we note that it performs worse than the best spike metrics, but better than the worse metrics that often diverge for this particular noisy (Poisson) input setup (figure 7).

To perform optimization, we minimize the loss over batches consisting of intervals of 400 ms of spike data, with learning rates $\alpha = \{1, 5\}$ %, and $\alpha = 5\%$ for SGD and Adam, respectively. The spike metrics we employ are as defined

in section 2, namely the van Rossum distance, the firing rate distance, and linear combinations of the metrics, after observing empirically that the firing rate distance was more stable than the van Rossum distance. We then fit five different pseudo-randomly initialized models using gradient descent and Adam for each combination of the loss functions, i.e. for the four combinations.

These were fitted over spike train intervals of 4s, divided into batches of 400ms, thus effectively normalising over 400ms time windows. The window of 400ms was chosen after empirically determining that it resulted in relatively good convergence in explorative work when using only one scalar value for each network-parameter.

Table 1. LIF network parameter intervals for pseudo-random uniform model initialization.

E_L	τ_m	τ_g	R_I
[-55, -45]	[1.2, 2.3]	[2.0, 3.5]	[135, 140]

In sum this resulted in a total of 20 fitted models per experiment configuration for a total of 10 configurations, consisting of 4 different pseudo-random generative model initializations, with 5 fitted models that were pseudo-randomly initialized and fitted for each loss function.

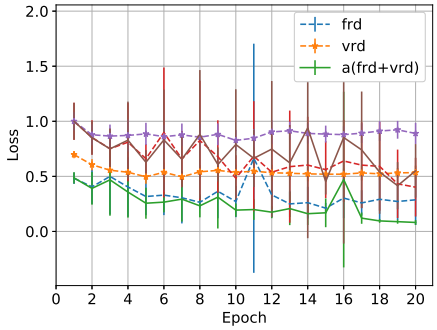

Figure 4. The average normalized training (lower) and test (upper) loss for the three loss metrics, normalised by the maximum test loss, where frd, vrd, and a(frd+vrd) denotes the firing rate distance, van Rossum distance, and the adaptive linear combination of the two, respectively.

Following optimization we evaluate functional network characteristics, including comparing the modules obtained by non-negative matrix factorization, and also look at second order statistics for the average firing rate, spike correlations between the fitted and generative models, and the spike variance and covariance. On average the geodesic distance is lowest for minimizing the firing rate distance using Adam (figure 7). This also holds for the best model, i.e. the model with the lowest geodesic distance. However, when considering the best fits, minimizing the adaptive combination of the firing rate and van Rossum distance using Adam, and the firing rate distance using SGD attains a model fit with the same distance. Note however that these two have higher average distances, with a higher deviation.

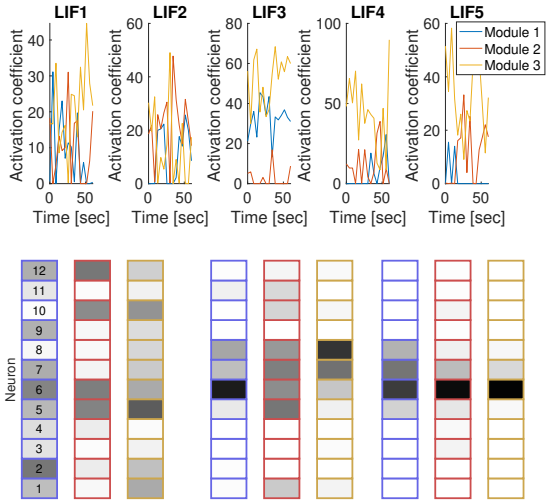

Figure 5. Non-negative matrix factorization activation coefficients (top panels) and modules (bottom panels); for the generative model with random seed 1 (left), and the fitted models in one experiment for random seeds 3 and 5 using the firing rate distance and Adam (middle, right). Note that the spatial modules are ordered and color coded after the activation coefficients, but that their spatial order may vary.

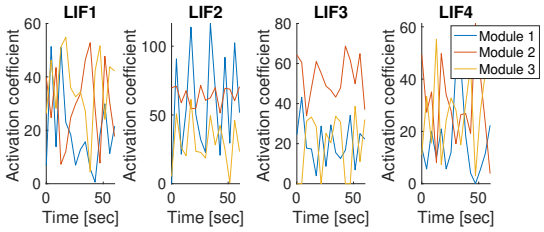

Figure 6. Activation coefficients of the three factorised modules for the generative model and each of its four different pseudo-random initializations around the pre-defined parameter means.

For all configurations, meaning combinations of optimization using either Adam or SGD, we also plot the average Euclidean distance between the inferred and target model parameters (figure 8). Interestingly, these were slightly further away from their true values when minimizing with Adam, but in all cases closer to the true parameter values.

5. Discussion

In this paper, we explored parallel parameter optimization for spiking neural network models with LIF neurons and exponentially decaying synapse models by using gradient based optimization for a set of different loss metrics. Our results indicate that the methodology can be used to replicate spiking behavior, and introduces a parallel methodology for spiking model inference.

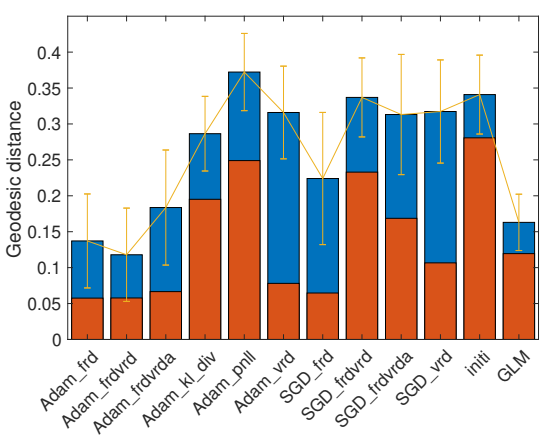

Figure 7. Average geodesic distances (blue, with error bars) between the non-negative matrix factorization modules of spike trains attained through optimization using the listed setups, and the factorized spike trains of the corresponding (target) generative model. The distances for the best model fits are shown in red (lower bars).

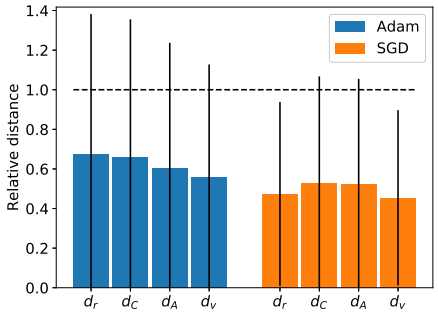

Figure 8. The average relative Euclidean distance across all experiments using the same loss metric for Adam (blue, left group) and SGD (orange, right group), relative to the the pseudo-randomly initialized model parameters' normalized distance to the true parameter values. The large deviance is a result of the large dimensionality (N or N^2) for each parameter, as this is a projection of the mean over all parameters.

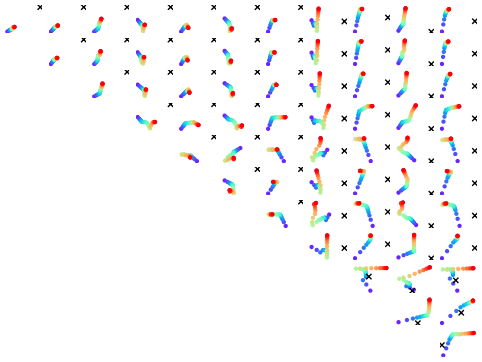

Figure 9. Parameter trajectories for R_I between each pair of neurons in the network model throughout optimization for one particular fit. Red denotes the first training iteration, moving throughout the color scale towards blue and violet.

Gradient based optimization methods are not guaranteed to find the global optimum. In line with this shortcoming, we found that convergence to local minima occurred. While perfect recovery of the true model parameter values cannot be guaranteed when minimizing the defined loss metrics due to the multitude of local minima in the parameter hyperspace arising from the high dimensionality of the model, we still observed that the more prominent model parameters fell close to their ground truth values when the rate based metric was minimized. Further, we observed that similar functional characteristics emerged in the inferred models, i.e high geodesic similarity between the fitted model spike trains and the target spike trains. The similarity was also greater than when compared to that obtained from predicting spike trains by using a GLM fitted to the sama synthetic spike data. This suggests that parallel optimization may be successful for the high-dimensional problem of spiking model inference by using spike trains as target data, and that inferred model can capture functional network characteristics of the modelled data. Furthermore, we observed that the inferred models when using the firing rate had a temporal sensitivity. This may be seen when considering that we fitted to sub-second intervals of spike data, which thus introduced local information. It is worth noting, however, that optimization using a metric that is commonly referred to as insensitive to temporal information may both reconstruct similar patterns in time, as well as functional network characteristics.

In some cases of model inference using spike trains as target data, properties about the input might be assumed, but in others little to no assumptions may be made. Therefore, we chose to model the input using a Poisson process, as this resembles the nature of neural activity to some extent (Grün, 2010). While perfect convergence may occur when using both the ground-truth input and outputs, it is possible that the input noise factor may have obscured of the gradient signal stemming from the van Rossum distance metric due to its temporal sensitivity. The firing rate metric is also more robust to local spike variability, since the expected rate over intervals of medium to longer length may be assumed to be roughly the same. This leads to the question of whether the van Rossum distance obscures the signal only when the model output is far from that of the target data. It turns out that a combination of the two metrics performs better than the van Rossum distance alone, but on average not as good as when only considering the firing rate. As such, our results suggest that the van Rossum distance should only

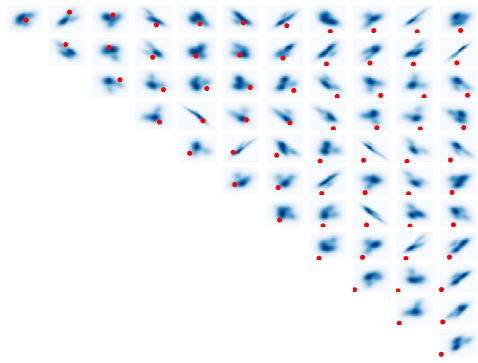

Figure 10. Gaussian kernel density estimates for R_I, inferred using the parameter values of all models fitted to the same generative model. Here using the adaptive combination of the firing rate distance d_r and the van Rossum distance d_v.

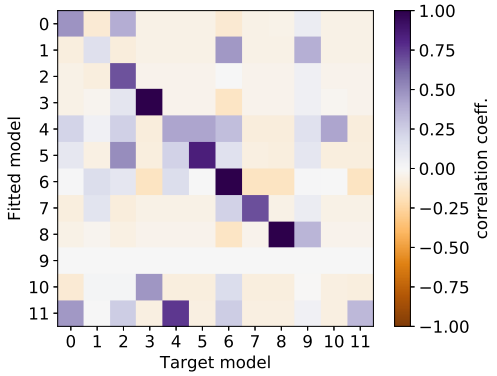

Figure 12. Spike cross-correlations between each pair of neurons for one particular fitted and generative model pair. In this run the adaptive combination of the firing rate distance d_r and the van Rossum distance d_v was minimized.

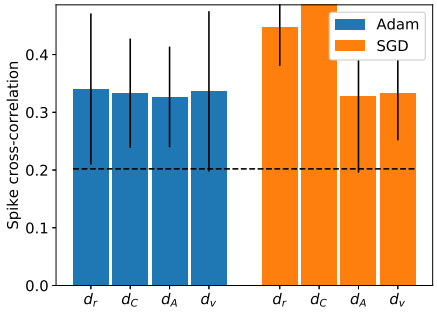

Figure 11. Average spike cross-correlation between fitted and target model spike trains for each loss function, and the two classes of optimizers.

be used when the signal to noise ratio is lower than in our experimental setup. Note that the above discussion is based on the results for the particular SNN models of this study, where the model is in part driven by noise. In scenarios where the signal-to-noise ratio is greater, or the model in itself is more strongly self-organizing, it might very well be that the loss metrics may have different effects.

Another aspect that may have a large effect on optimization is the continuity and stability of the model system. An SNN has non-linearities in its reset rules, and also may exhibit different types of spiking. Non-linearities may obscure the gradient signal during optimization in itself. When viewing this in conjunction with that we attempt to optimize all parameters in parallel, it might be expected that a signal that is more sensitive to local changes may be too chaotic for optimization to converge. As such, it might be fruitful to further constrain the search space such as by introducing linear constraints, or by performing sequential parameter optimization. We note that the average Euclidean distance between the fitted model parameters and the true model parameters always decreased throughout optimization, and was within the Gaussian kernel density estimates for most of the values of the parameters E_L, τ_m, and particularly R_I. From empirical observation, it seems that the implemented model is the most sensitive to the membrane resistance variable, R_I. Thus, if performing sequential optimization, this could be one variable to fix. Further, it may be shown that this value may be normalised relative to network size in order to generalize the outlined methodology to arbitrary network sizes. This would, however result in a slightly different shape of the term including the membrane resistance, potentially allowing for a wider interval of values for which similar model behavior may occur, ultimately changing the parameter landscape by the normalising of this parameter. In any case, when functional network characteristics are replicated in an artificial model, the model itself may be used for hypothesis testing and generation relating to the modelled area. Future work could address the generalizability of the approach to (1) biological spike train data, and (2) using more complex neuron models, such as the generalized leaky integrate-and-fire model (Allen Institute for Brain Science, 2017).

References

Allen Institute for Brain Science. Allen Cell Types Database, Technical White Paper: GLIF Models, 2017. URL http://help.brain-map.org/download/attachments/8323525/GLIFModels.pdf.

Bengio, Y., Boulanger-Lewandowski, N., and Pascanu, R. Advances in optimizing recurrent networks. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 8624–8628, 2013. ISSN 15206149. doi: 10.1109/ICASSP.2013.6639349.

Grün, Sonja; Rotter, S. *Analysis of Parallel Spike Trains*. Springer, 2010. ISBN 9781441956743. doi: 10.1007/978-1-4419-5675-0.

Hodgkin, A. L. and Huxley, A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117 (4):500–544, 8 1952. ISSN 0022-3751 (Print). doi: 10.1080/00062278.1939.10600645.

Huh, D. and Sejnowski, T. J. Gradient Descent for Spiking Neural Networks. Technical report, Salk Institute, 2017.

Jun, J. J., Steinmetz, N. A., Siegle, J. H., Denman, D. J., Bauza, M., Barbarits, B., Lee, A. K., Anastassiou, C. A., Andrei, A., Aydin, , Barbic, M., Blanche, T. J., Bonin, V., Couto, J., Dutta, B., Gratiy, S. L., Gutnisky, D. A., Häusser, M., Karsh, B., Ledochowitsch, P., Lopez, C. M., Mitelut, C., Musa, S., Okun, M., Pachitariu, M., Putzeys, J., Rich, P. D., Rossant, C., Sun, W. L., Svoboda, K., Carandini, M., Harris, K. D., Koch, C., O'Keefe, J., and Harris, T. D. Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551(7679):232–236, 11 2017. ISSN 14764687. doi: 10.1038/nature24636. URL https://www.nature.com/articles/nature24636.

Nelder, J. and Wedderburn, R. Composite Link Functions in Generalized Linear Models Author (s): R . Thompson and R . J . Baker Published by : Wiley for the Royal Statistical Society Stable URL : http://www.jstor.org/stable/2346381 Composite Link Functions in Generalized Linear Model. *Journal of the Royal Statistical Society*, 135(3):370–384, 1972.

Parzen, E. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3): 1065–1076, 9 1962. ISSN 0003-4851. doi: 10.1214/aoms/1177704472. URL https://projecteuclid.org/euclid.aoms/1177704472.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., Facebook, Z. D., Research, A. I., Lin, Z., Desmaison, A., Antiga, L., Srl, O., and Lerer, A. Automatic differentiation in PyTorch. In *Advances in Neural Information Processing Systems*, pp. 8024–8035. University of Warsaw, 10 2019.

Pillow, J. W., Shlens, J., Paninski, L., Sher, A., Litke, A. M., Chichilnisky, E. J., and Simoncelli, E. P. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995–999, 2008. ISSN 00280836. doi: 10.1038/nature07140.

René, A., Longtin, A., and Macke, J. H. Inference of a Mesoscopic Population Model from Population Spike Trains. *Neural Computation*, pp. 1–51, 6 2020. ISSN 0899-7667. doi: 10.1162/neco{_}a{_}01292. URL https://www.mitpressjournals.org/doi/abs/10.1162/neco_a_01292.

Rolls, E. T. and Treves, A. Introduction. In *Neural Networks and Brain Function*, chapter 1, pp. 418. Oxford University Press, Oxford, UK, 1998. ISBN 0198524323.

Rosenblatt, M. Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, 27(3):832–837, 9 1956. ISSN 0003-4851. doi: 10.1214/aoms/1177728190. URL https://projecteuclid.org/euclid.aoms/1177728190.

Rule, M. E., Schnoerr, D., Hennig, M. H., and Sanguinetti, G. Neural Field Models for Latent State Inference: Application to Large-Scale Neuronal Recordings. *bioRxiv*, pp. 543769, 2 2019. doi: 10.1101/543769. URL https://www.biorxiv.org/content/10.1101/543769v1.

Seung, Sebastian, H. and Lee, D. D. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999. ISSN 00280836. doi: 10.1038/44565. URL http://www.nature.com/doifinder/10.1038/44565.

Skou, J. C. The influence of some cations on an adenosine triphosphatase from peripheral nerves. *BBA - Biochimica et Biophysica Acta*, 23(C):394–401, 1 1957. ISSN 00063002. doi: 10.1016/0006-3002(57)90343-8.

Tavanaei, A., Ghodrati, M., Kheradpisheh, S. R., Masquelier, T., and Maida, A. Deep learning in spiking neural networks. *Neural Networks*, 111:47–63, 3 2019. ISSN 18792782. doi: 10.1016/j.neunet.2018.12.002. URL https://doi.org/10.1016/j.neunet.2018.12.002https://pubmed.ncbi.nlm.nih.gov/30682710/.

Van Rossum, M. C. A novel spike distance. *Neural Computation*, 13(4):751–763, 4 2001. ISSN 08997667. doi: 10.1162/089976601300014321. URL https://www.mitpressjournals.org/doix/abs/10.1162/089976601300014321.

This report provides the basis for SNN inference using GBO, different loss metrics and optimisers for the case of LIF SNNs, and also presents a way of evaluating the goodness of fit using NMF, comparing with GLMs as a baseline model. This is the foundation upon which the rest of the work in this thesis is based on, and expands on. In the rest of this thesis, we consider the case of using the Adam optimiser with different classes of SNNs and either a rate-based loss metric over intervals of spikes, the van Rossum distance, or the negative log-likelihood where considering probabilistic models. As no significant performance gain was observed when using a combination of the two metrics was observed, this is not used in further GBO work using other SNN models. In fact, as we shall see, when increasing the model complexity, the fact that the van Rossum distance may emphasise the timing of spikes in such a way that the gradient may be obscured, may be disastrous on inference, resulting in divergence.

Chapter 5

The frontier of SNN inference

In order to study how to infer spiking neural network models (SNNs) we need several ingredients: (1) a definition of the system, (2) an algorithm for model inference, and (3) an implementation of (1) and (2), where the algorithm for the case of gradient descent also necessarily contains (3) a way to measure and assess model performance, which is ideally comparable with existing methodologies and models.

Research on SNNs using gradient-based optimisation is fairly limited, and includes approximate Bayesian computation for smaller models, conversion learning in which more traditional ANNs are transformed to simpler SNNs after training, or surrogate gradient descent, in which a surrogate signal over the model spikes, usually as a function over the membrane potential, is used in order to obtain a differentiable output that may be used to optimise the model's parameters over a given loss metric by backpropagating the error signal. Notably, there exists some research on gradient descent for SNNs in which the model is defined such that its parameters may be used directly for error signal gradient backpropagation, including [52], [92]. This is slightly different from surrogate gradient descent in that the backpropagation is a more natural extension of the system, and thus albeit remaining unrealistic in terms of biology, it is closer to learning in a Hebbian and spike-time dependent plasticity (STDP) fashion, as noted by the authors in [52].

With the goal of accelerating SNN inference research, we looked at the most prominent state-of-the-art methodologies for SNN inference, which may be divided into (1) gradient based optimisation (GBO), and (2) approximate Bayesian computation (ABC) [68, 92, 22, 69]. GBO enables leveraging recent ML advances from deep learning, and albeit biologically implausible as a learning rule in its own right, allows for in-place inference of models that are biologically realistic. The computational cost of doing

gradient based computation is also exponentially less than that of doing ABC, as this involves doing a costly (Monte-Carlo) sampling step [92]. Therefore, GBO is the main focus of this work, and leveraging system definitions that may allow for more information to be captured and propagated by means of gradient descent types of optimisation is therefore also something that is desirable, and notably consistent with the works that have been extended.

As for ABC, Bayesian inference has until recently been intractable for SNNs due to their high-dimensionality. However, recent advances in deep learning, leading to the advent of sequential neural posterior estimation (SNPE) [39, 29, 34, 22], in which a DNN is trained to estimate a posterior over a prior given observations, has allowed for much more efficient sampling using this amortised approach; sampling from the DNN to perform posterior parameter estimation (which in turn uses GBO in the posterior inference, i.e. for fitting the DNN). Note however that this procedure still has a high cost due to the sampling procedure, and does not scale well with an increase in the network size. Using a Bayesian approach, we may however estimate the posterior over the model parameters, given our approximation of the prior. This also yields a type of uncertainty estimate around the posteriors. Note however that the posterior will be skewed by the prior approximation, in our case given by SNPE.

Using a surrogate gradient approach, we may fit the parameters of the system by error minimisation using a differentiable loss function. By working with biological spike train data, which is a very widely used format, we cannot however make many assumptions about the input, unless the recording somehow also contains information about this signal. Therefore, we have to model the input in some way, taking care in considering how assumptions may affect model behaviour during learning. As a rule of thumb, we want to have a signal which allows for the network to perform the task at hand, or to replicate the observed data, whilst remaining biologically reasonable. In other words, we want to have input that puts the system in a realistic mode of behaviour, and yet allows us to optimise its parameters. This means providing an informative signal, which is meaningfully transformed by the model with regards to output error minimisation, and thereby also usable in this regard.

5.1 GBO compatible SNN framework in PyTorch

In order to test the extent to which gradient based optimisation may be leveraged for spiking neural network model inference, we have developed and implemented a novel

framework in PyTorch [90, 91]. It allows performing gradient descent over a specified loss metric, for any differentiable SNN model definition, in a highly modular way. We have implemented a surrogate gradient approach over the membrane potential itself for LIF and GLIF neurons with rate-based metrics as well as spike-time dependent metrics, and over the spike probabilities by using the negative log-likelihood for SGIF neurons, as well as for non-leaky integrate-and-fire neurons by directly using the synaptic currents as readout signals, and considering a more high-dimensional output target signal. Notably, we extended the latter synaptic current model, as originally presented by [52], and combined it with leaky integrate-and-fire models, for which optimisation was highly successful. The work with the synaptic current modification is presented later in chapter 7.

As for hypothesising the extent to which GBO may be used to retrieve model parameters that are close to the ground-truth values, we may assess this by fitting to synthetic data. We hypothesise that we will retrieve local minima which is not close to the ground-truth values where this is accessible, but that the models will however capture the spike statistics to a large extent, potentially allowing for functional analysis and model probing for further hypothesising. This can be understood by considering the error landscapes formed by the loss metrics, which we shall plot later in this chapter in order to illuminate this. Further, we shall also test the extent to which we have captured similar ensembles in the spike trains of the fitted models, by comparing the modules attained by non-negative matrix factorisation, a dimensionality reduction technique well suited for neural spike train analysis due to its non-negativity constraint, irrespective of the parameter values of the fitted models.

As a baseline model for comparison in NMF module similarity, we use generalised linear models (GLMs) (see section 2.5.2), which are good candidates for capturing spike statistics in the NMF analysis due to their exponential family (and here Poissonian) formulation. However, we hypothesise that GBO for SNNs may enable capturing a higher factorised module similarity than the GLMs fitted by maximum likelihood estimation (MLE). If so, this could either be due to an inductive bias (by model definition), or due to the richer range of behaviours available to the SNN models.

Current implementations of SNNs in the field are commonly done in Brian 2 [36] or Matlab. Upon commencing work on SNNs using LIF neurons, the only recent work we found that supported optimisation in a supervised or semi-supervised manner was in Theano [7], a legacy Python library for symbolic programming. With recent advances in ML not only through increased computational resources and novel opti-

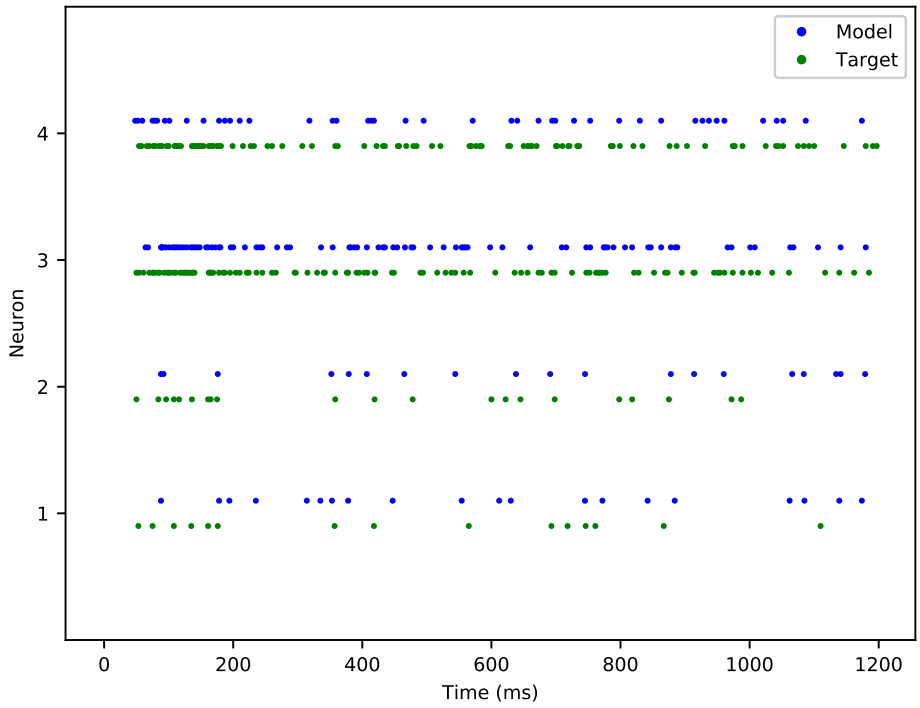

Figure 5.1: Target and fitted GLIF model spike trains for one of the GBO inference experiments when fitting to target spike train data generated by a hand-engineered GLIF SNN (green/bottom spikes in the spike train plot).

misation techniques, but also in programming libraries that also compile to and run on the GPU, and in either case contains highly optimised library code implemented in a lower-level programming language (typically C, C++, or Cuda), we wanted to make use of this aspect too, and after too many hours of fighting with at best poorly documented legacy Theano-library code, we decided to set out to implement SNNs in PyTorch. This led to the implementation of a modular framework and pipeline of SNN inference using GBO in PyTorch [90, 91]. Key to allowing using PyTorch, is maintaining a neuronal state throughout simulation, which we implemented by enforcing sequential input propagation throughout the model, such that the internal state is updated for each time-step. With this simple trick, intervals of input mutate the state with each input corresponding to some desired time-step constant, in my code set to 1ms for simplicity. However, it does result in that intervals of simulation need to be performed for each backward pass, i.e. updating the model parameters given error gradients.

We implemented a general SNN optimisation framework on top of PyTorch [90, 91] and utilised PyTorch's autograd-feature for backpropagation of error gradients.

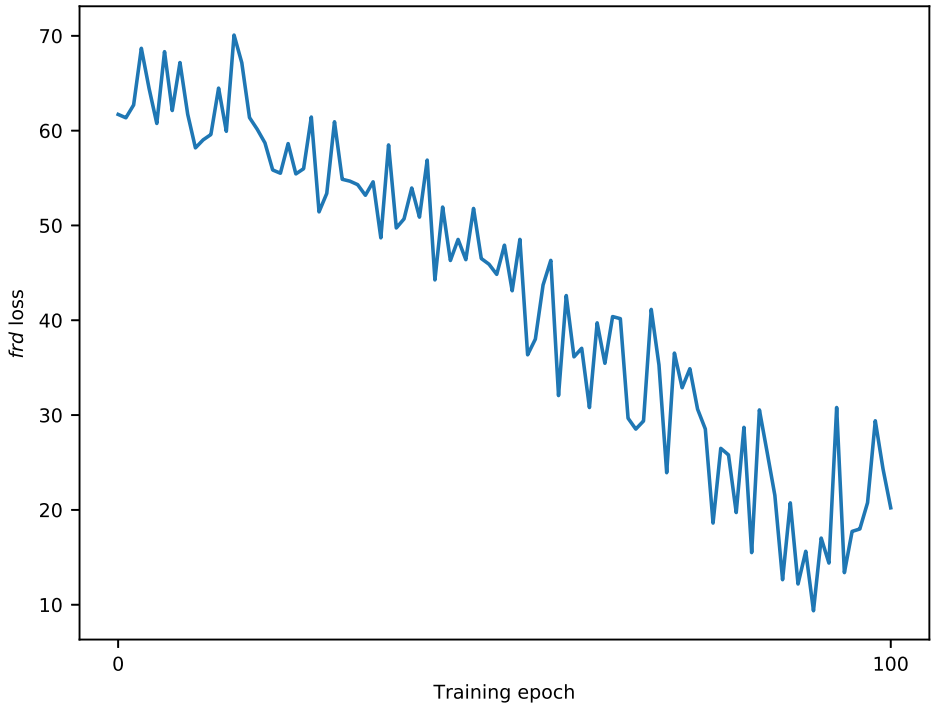

Figure 5.2: Loss per training epoch for one of the GBO inference experiments, with the final model and target intervals plotted in figure 5.1, and the membrane potentials in figure 5.3.

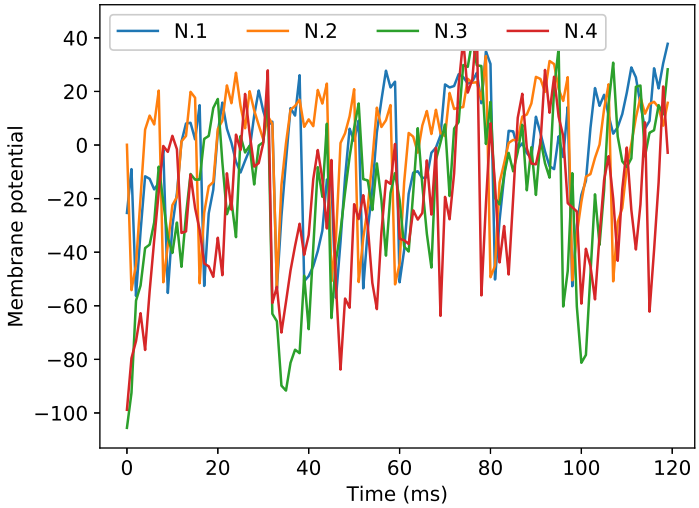

Figure 5.3: The membrane potentials for the fitted GLIF model in figures 5.1, 5.2.

Key to enabling backpropagation is defining a differentiable output signal. Traditionally, the sigmoid function is a common choice, i.e.

$$s_t(v) = \frac{1}{1 + e^{-(v_t - (\theta_v + \theta_s)}} \tag{5.1}$$

where s_t denotes whether a neuron spikes at time t with a membrane potential v_t. Or by considering the spiking to be a readout of a continuous parameter/variable such as the post-synaptic current from each neuron, as is done in the case of the continuous sub-threshold synaptic model in [52], or optionally the probability of spiking, in a probabilistic formulation of integrate-and-fire SNNs, as in [92].

To perform optimisation, an initial model parametrisation is drawn uniformly from parameter intervals that are constrained to meaningful parameter values, and then perturbed with input drawn from some input generator function. Note that the parameters of this function may also be fitted. The model output is then compared with the output of a target data set, and the specified loss metric and distance is calculated between the produced model output and target spike train, as illustrated in figures 5.1, 5.2, 5.3.

5.1.1 Batching and SNNs

Despite the need to do simulation in a sequential manner as described above, we implemented iteration over batches of activity, effectively allowing for a type of batch normalisation, and increasing the parallelisability of the algorithm with the number of batches run in parallel, allowing for better use of the computational resources at hand.

5.2 Experiments

Each experiment consists of fitting a model to a data set synthetically generated by a hand-engineered model of the same class. The target models and data were constructed such that they produced an array of behaviours and spike correlations, and which most importantly do not lie in a chaotic behavioural regime. This allows for comparing inference performance by both the loss metric and the average parameter distance, since the ground truth values of these are available. For a comparison across model classes fitted to the same data set, as well as for biological data, we report our work on this in chapter 6, in which we fit to biological data from in vivo recordings.

Returning to GBO for the synthetic data; for each experiment, a model is pseudo-randomly initialised by first setting a random seed (such that results may be repro-

duced) and then drawing initial parameter values uniformly from pre-defined parameter intervals, which are set such that the model is in a non-silent and non-chaotic mode of behaviour, with the ranges being set to biologically plausible values. Then, each model is fitted until either the average gradient has decreased to a fraction ($\approx 1\%$) of the initial average gradient, or for a set number of training iterations, $N_{exp} = 100$. Each training iteration consists of model simulation for an interval of $t = 1200$ms, and then updating the parameter values according to the gradients. The target data is synthetic data generated by the same model class, such that we may compare the retrieved parameters with the ground-truth. Data was generated as previously mentioned by hand-engineered models of the same class (i.e. LIF, GLIF, or SGIF). Note also that all of these model types are to biological spike train data in chapter 6. Also, for the SGIF class, we also calculate the Pearson correlation coefficient, as well as root mean squared error (RMSE) between the produced and target signal, in order to compare model performance with the results reported in [92].

Since some model parameters are more sensitive than others, we define linear constraints for each parameter, which PyTorch allows for incorporating into the computational graph in a simple manner by defining a hook for each backward pass over the parameters, i.e. for each gradient update, effectively clamping the gradients such that the parameters do not wander outside of realistic intervals. These intervals are naturally far wider than the initialisation intervals, but by considering what intervals are realistic, this is a simple and natural way to exclude unrealistic modes of model behaviour whilst constraining optimisation simultaneously. Note that the constraints do not pose small intervals of possible values for the parameters, but merely function as limiters enforcing realistic ranges for them, and as a side effect may help avoid divergence by over-adjusting a sub-set of the parameters, typically during the first epochs of optimisation.

Prior to the main experimental setup above, we simulated each model class for realistic intervals of parameters, whilst fixing the other parameters to values that we had empirically or analytically determined to be able to produce meaningful output, and plotted the loss and rate when compared to the hand-engineered target model of the same model class, essentially performing a grid search over parameters allowing to plot the marginals of the parameter landscape formed by the loss metric and target model signal. The results which are included below illuminate how optimisation traverses the gradients formed by the parameter landscape, and gives an idea about the extent to which the procedure is applicable to the model class.

5.2.1 Error and parameter landscapes

When performing gradient descent, we traverse an error landscape given by the loss metric, and iteratively update the parameter values by moving a certain amount (often called the step size) in the direction along the error gradient. This updates the parameters to values that would result in a lower loss for the current data interval at hand, for which the loss was computed. As may be seen from this more conceptual description; in order for gradient descent to work, the error signals need to be informative over the space of potential true values, and continuously defined for paths that lead to the regions that may contain the target parameter sets and values, or minima.

As we shall see below when plotting 2D projections of the loss across different parameter combinations, this isn't the case for all of the parameters and model types, especially when only considering a rate-based metric. Note however that this is to be in part expected particularly for a rate-based metric, as the loss signal is to some extent oblivious to the timing of spiking in the data, as it only considers the neuronal rates. This does not render the loss metric unusable, as we may expect to capture the rates well with a rate-based distance metric - however, the ambiguity of the optima is an issue wrt retrieving the true ground-truth values. The ambiguity of the error landscape formed by the rate based metrics is illustrated in figures 5.4, and 5.5 included in this section.

When it comes to the likelihood metrics that calculate the likelihood of producing the target spike train given the simulated probabilities, and assuming either a Bernoulli or Poisson distribution, over the model parameters, arguably seems to be somewhat more suited for retrieving sensible parameter-configurations. However, as illuminated in part by figure 5.6, these metrics are also ambiguous both in terms of retrieving the ground-truth, and defining a constrained set of configurations.

5.2.2 Gradient descent with Adam

To perform optimisation, we used Adam, the adaptive momentum implementation of gradient descent [61]. We note that the spaces for which error gradient given by the firing rate distance is ambiguous, i.e. the gradient may wander somewhat arbitrarily into the space of low to zero error, where it settles, and is thus not satisfactory in terms of defining a signal that will allow for retrieving the ground-truth. Further, for the van Rossum distance, these landscapes are even more obscure and noisy. Therefore, we have excluded using this metric in our experiments in this work. However, under

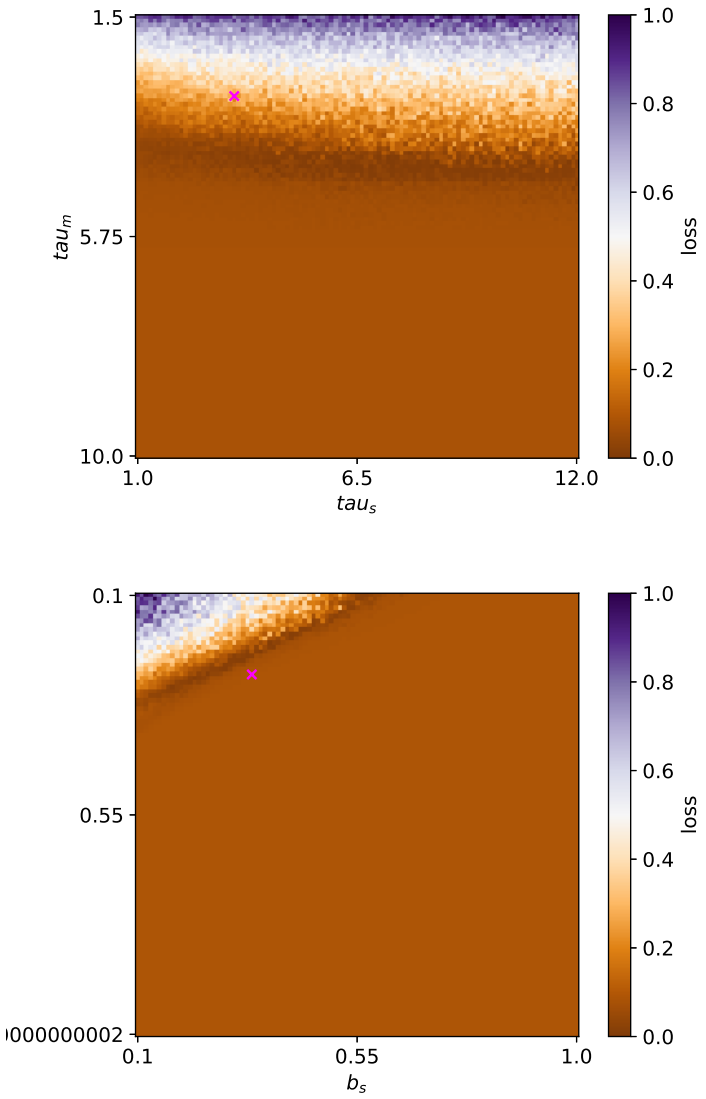

Figure 5.4: The parameter error landscape for a GLIF SNN for the parameters τ_s, τ_m (top), and b_s, a_v (bottom).

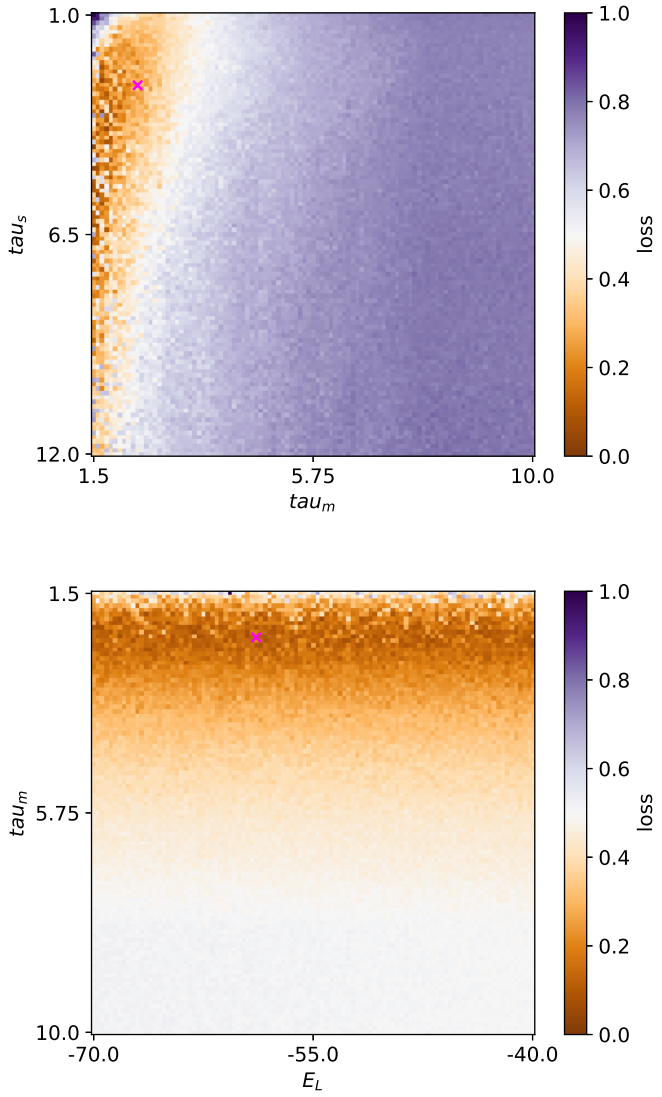

Figure 5.5: The parameter error landscape for a LIF SNN for the parameters τ_s, τ_m (top), and E_L, τ_m (bottom).

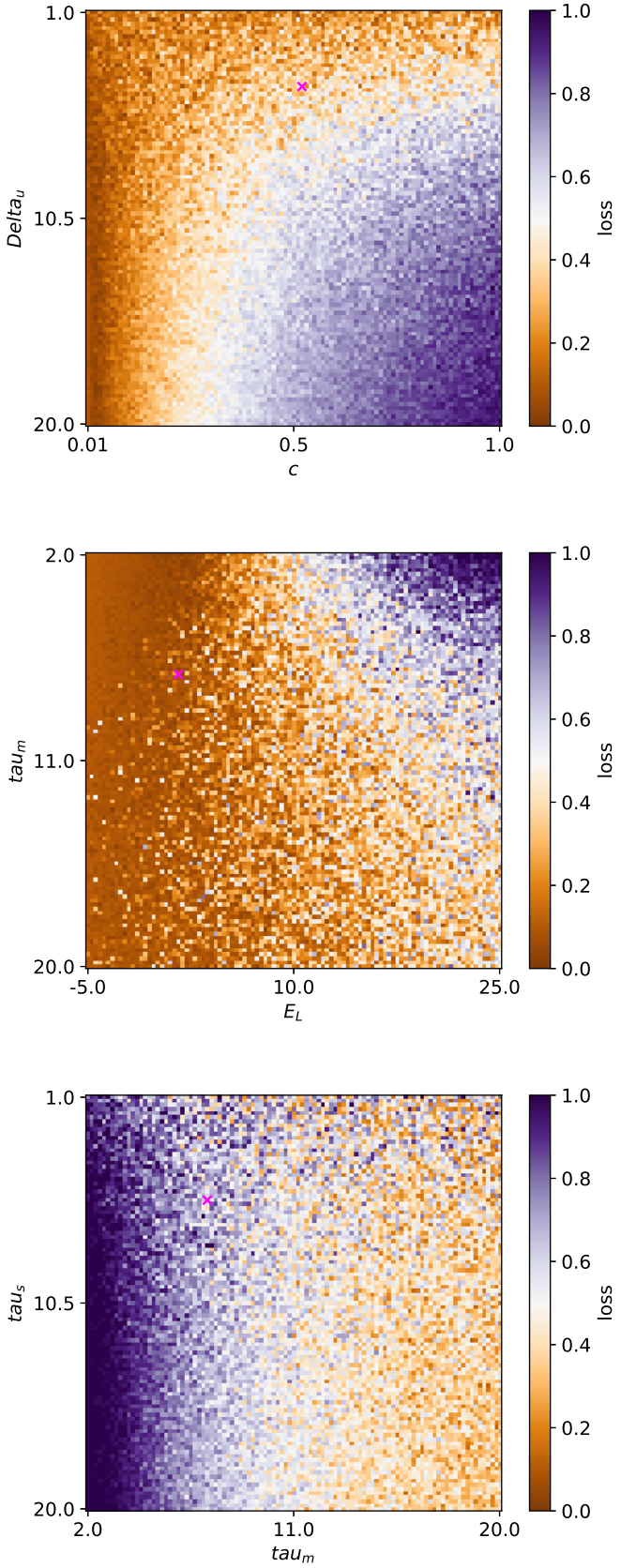

Figure 5.6: The parameter error landscape using the firing rate distance metric, for a population ($N = 4$) SGIF model.

more well-defined conditions, where the timing of individual spikes relative to the input signal and other neurons' signals is more informative than misleading, the van Rossum distance should theoretically outperform the firing rate distance.

For each model type, we used the firing rate distance from the target synthetic model outputs to the model being fitted, to optimise its parameters, as previously mentioned until the error gradients were diminishing and thus parameters settling into a local minimum, or for a certain number of training iterations ($t_{iter} = 100$. This was performed for 20 experiments, for each model class. We then assess the fitted model rates, parameter values, and the loss, and perform NMF analysis on fitted model as well as target model and data, assessing the geodesic similarity of the factorised modules.

Further, we perform direct neuron-level, i.e. full-scale, model inference of a heterogeneous mixture of stochastic general integrate-and-fire (SGIF) neurons, and find that albeit converging to local minima, the stochastic formulation and the optimisation over the NLL lends itself better to GBO than the surrogate over the LIF-model types when using the firing rate distance. For all model classes, some parameters have a more prominent effect on the the loss than others, particularly when considering the rate-based loss metric.

5.3 Results

Results are reported for 20 experiments for each model class, fitted to models of size $N = 4$ for LIF and GLIF neurons in this chapter, and synthetically for SGIF models to models of size $N = 4$ as well as $N = 21$. These are abbreviated as "meGIF" and "miGIF" respectively - representing meso-GIF and micro-GIF, adopted from [92]. The main results are the rates per configuration (i.e. model class and loss function) - implicitly also representing the loss, the parameter distances from the fitted to the ground-truth models, and the geodesic similarities as obtained via NMF.

5.3.1 GBO results

Included here are figures from two experiments, one for a GLIF SNN, and one for a LIF SNN, illustrated in figures 5.7, 5.8, 5.9, 5.10. As may be glimpsed from these, particularly for the LIF model, we may end up in local minima further away from the ground truth. On average, we end up further from the ground truth for LIF models, and around the same distance for GLIF models, whereas for SGIF models, we end up

somewhat closer. As for the rates, these also reflect these results, with the closes fits being attained for the GLIF and SGIF models.

Overall, these results, particularly illustrated by the average parameter distances in figure 5.11, show that ground-truth retrieval is highly unlikely with the specified model definitions and loss metrics, as illuminated by the error landscape plots in 5.2.1.

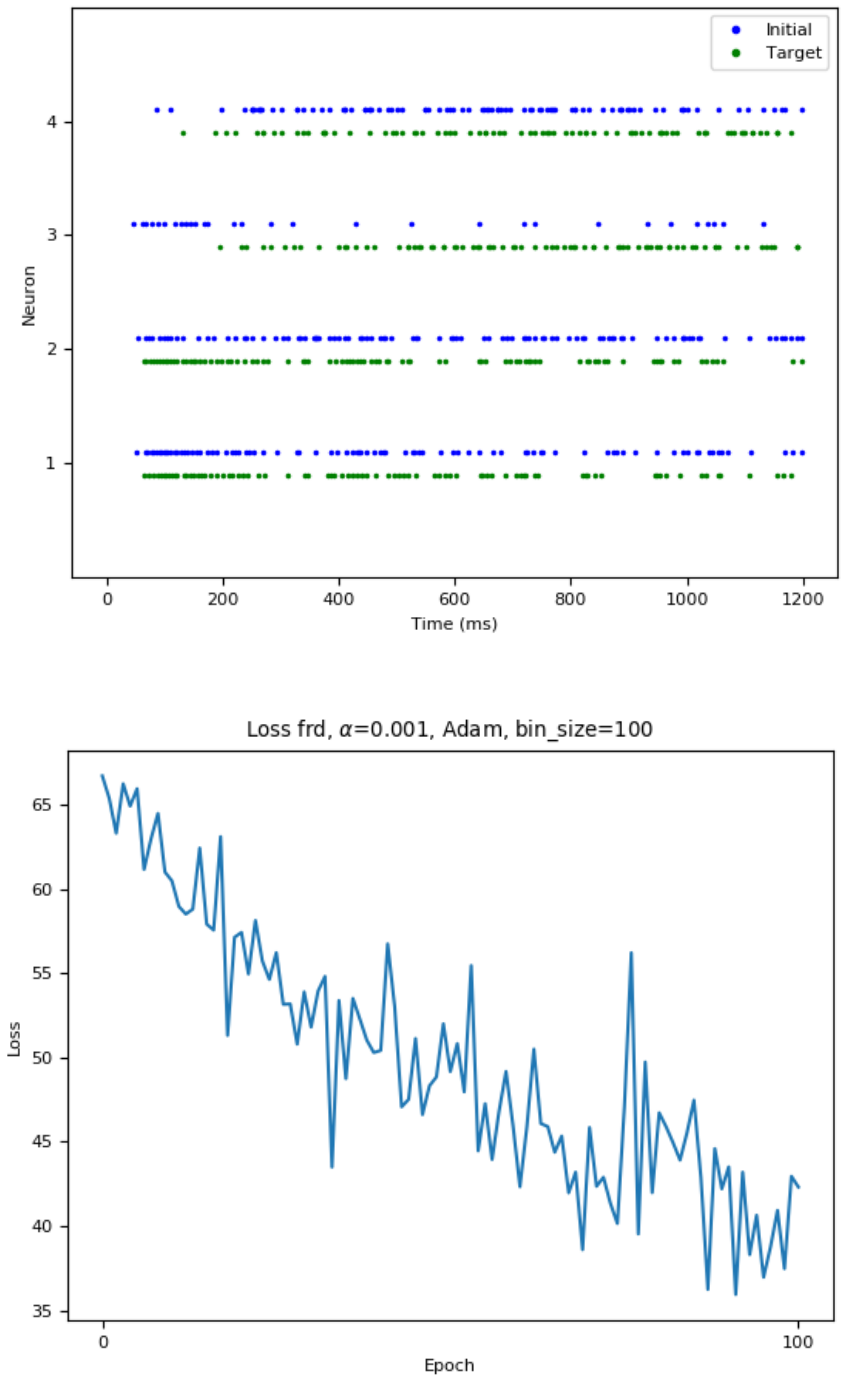

Figure 5.7: Sample GLIF SNN, spike trains after training (top) for a particular experiment, along with the loss per epoch (bottom), corresponding membrane potentials and parameter inference trajectories in 5.8.

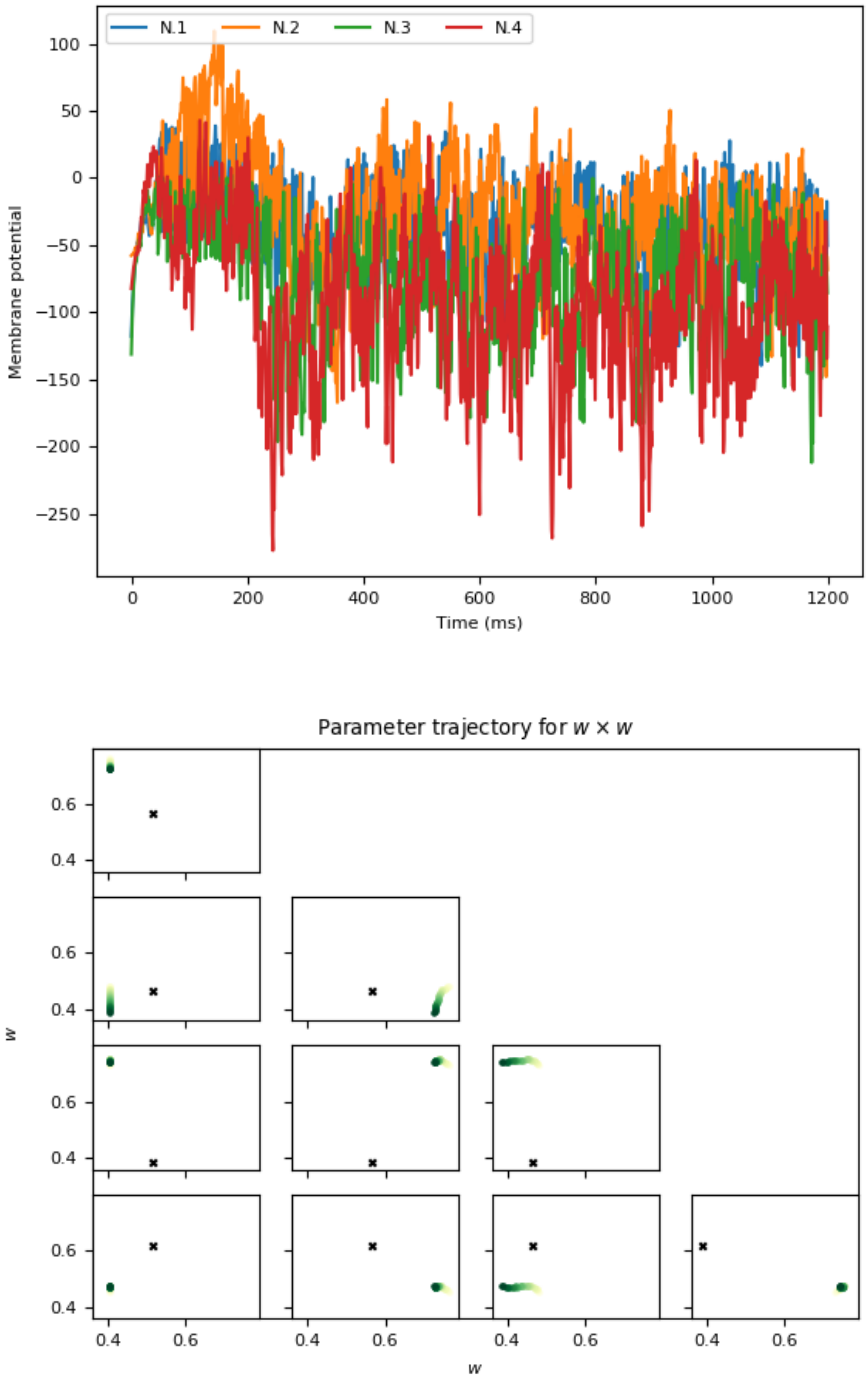

Figure 5.8: Membrane potentials (top) for the spike train plotted in 5.7, and the average trajectory (bottom) of the weights throughout inference.

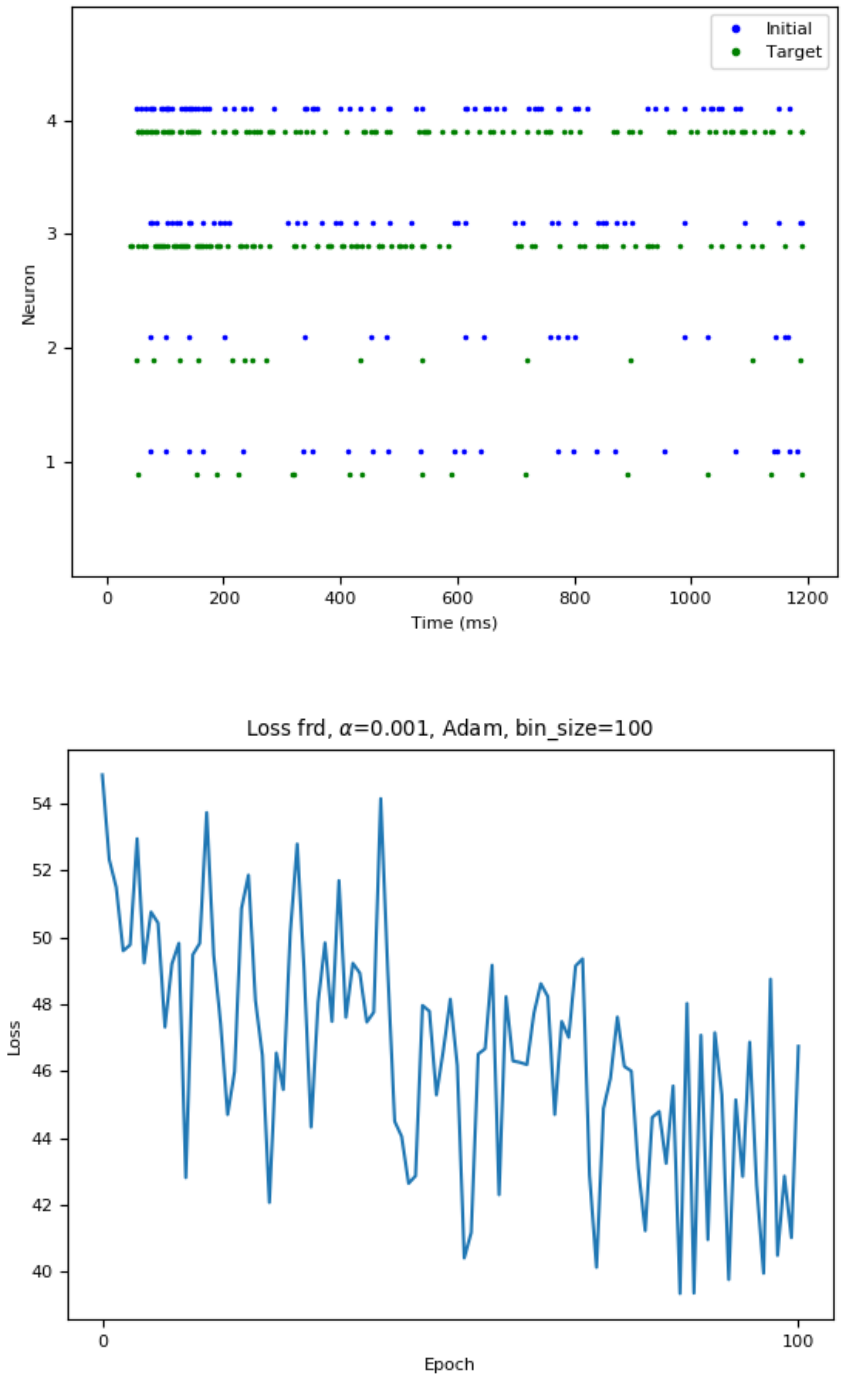

Figure 5.9: Sample LIF SNN, spike trains after training for a particular experiment, along with the loss per epoch, and corresponding membrane potentials for the spike train plotted above and the average trajectory of the weights throughout inference in 5.8.

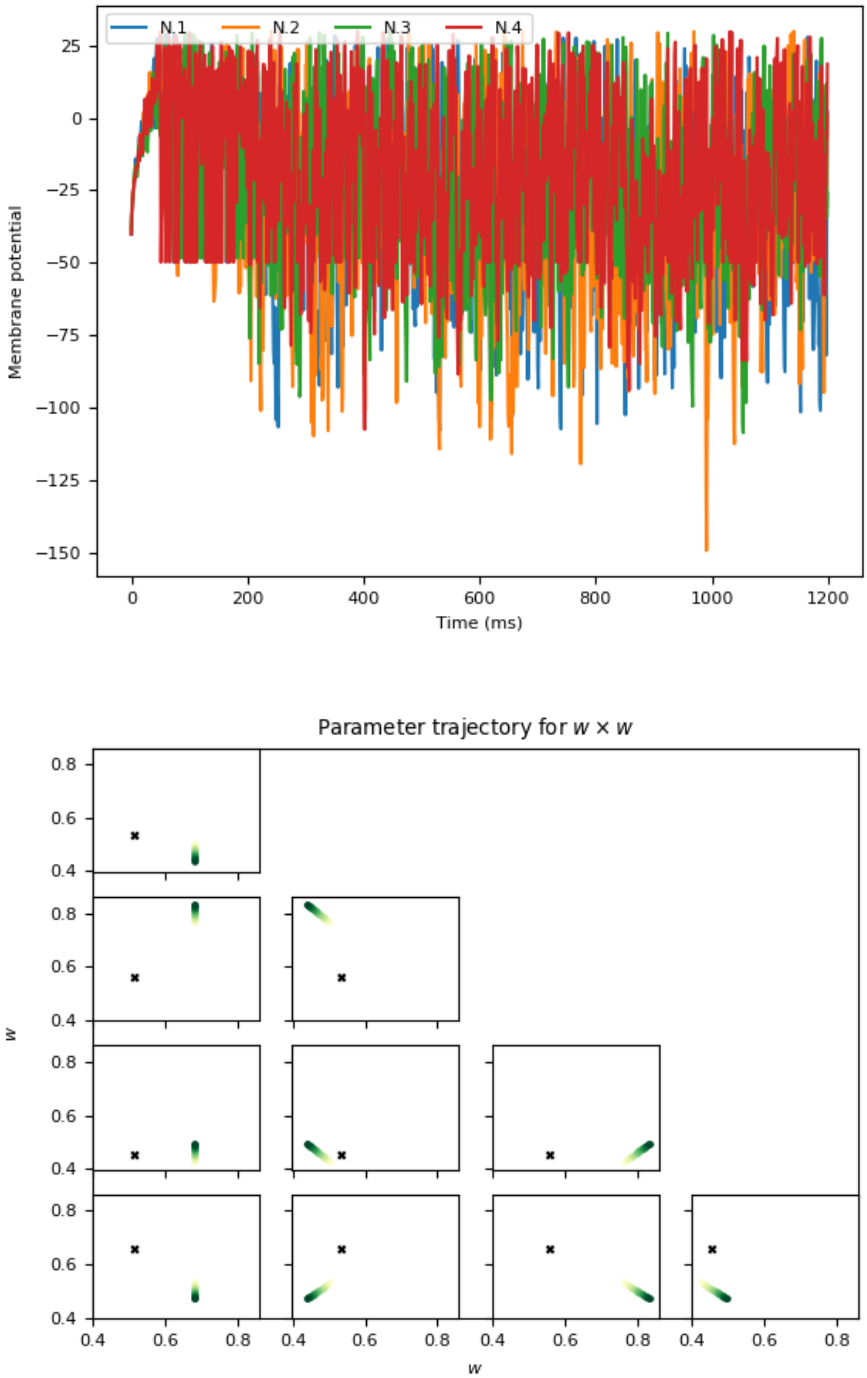

Figure 5.10: Sample LIF SNN membrane potentials for the spike train plotted in 5.9, and the average trajectory of the weights throughout inference.

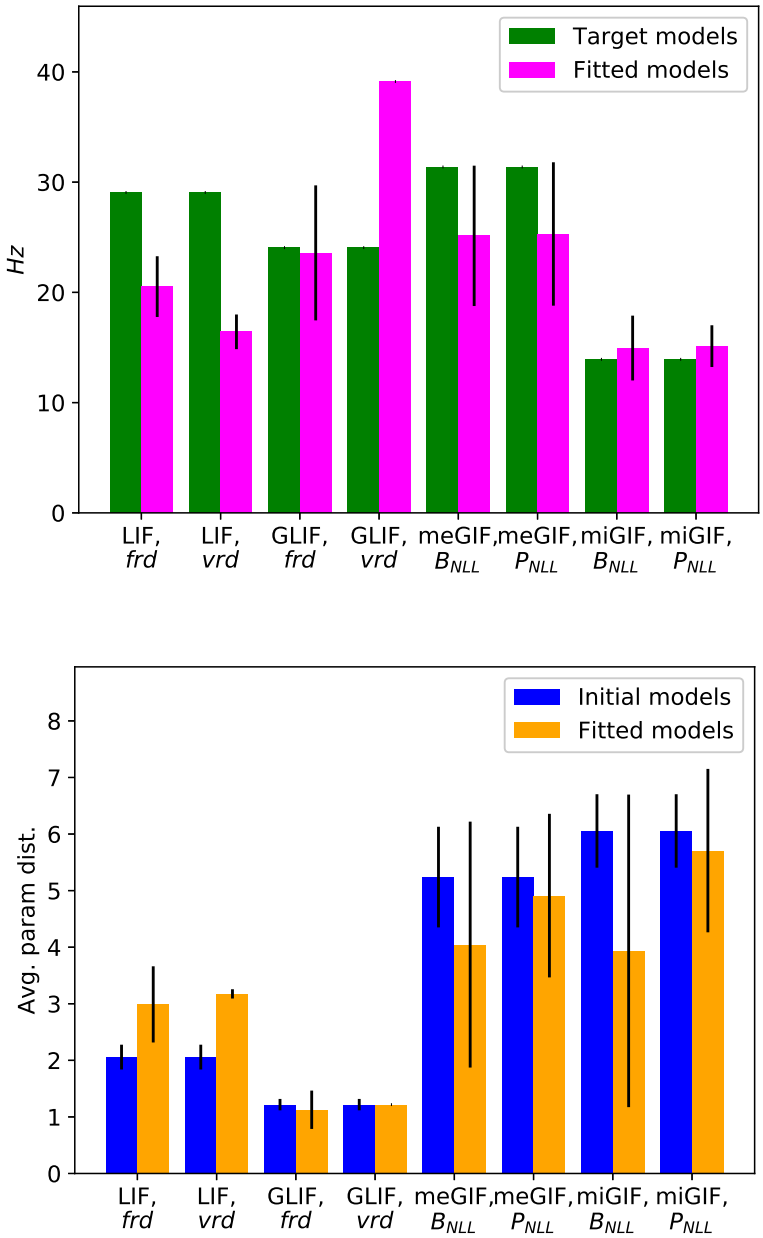

Figure 5.11: Fitted model rates across model types and loss metrics, and average parameter distance between the ground-truth model and both the initial and converged inferred models when using **GBO** for model inference. See figure 5.13 for comparison with SBI.

5.3.2 Stochastic general integrate-and-fire neurons

Figure 5.12 illustrates that GBO is possible for direct, scalable inference of neuron-level models when using a stochastic formulation of the LIF model and the Bernoulli or Poisson NLL by assuming spikes distributed in either of the corresponding probability densities, both being exponential family distributions with shared properties for the case of spike trains. However, as is found and described in table 5.1, the models are less correlated than when using MCMC sampling over the full posterior. It can be argued that this is expected, since no parameter dependence is assumed in the GBO inference methodology, as in the ABC-procedure of [92]. Convergence rates are nonetheless good (table 5.2), and parameter distances are somewhat closer on average when using GBO (figure 5.11), and in contrast significantly further away from the ground-truth than the initial, pseudorandomly distributed model parameters in the case of SBI (see figure 5.13).

5.3.2.1 Comparison with published ABC results

In addition to implementation of GBO for LIF and GLIF SNNs, we test our approach on SGIF neurons, which may lend themselves better to GBO with NLL minimisation over spike probabilities. We here report our comparative results for this model class, as when compared to the work which we adopted the model from [92].

5.3.2.2 Model input perturbation and formulation

While we want to model the site from which spikes have been recorded and decoded, we do not have the input to the site recorded from in biological spike train data, as is often the case. However, we may make some assumptions about the statistical nature of the input data, and incorporate this into model perturbation. Optionally, we may even design a particular perturbation scheme, such as sinusoidal stimulation, as this may also be performed in vitro. In this work we perturb the models with white noise for the rate-based experiments, sine-modulated white noise during training, and also with an Ornstein-Uhlenbeck process during testing of the stochastic general integrate-and-fire (SGIF) models (also optimising over the NLL as previously mentioned) as done in [92] in order to keep our methods as consistent as possible and therefore results more comparable by adopting and extending their work, and lastly with sine-modulated linear transformations using different random seeds for our implementation and extension of [53] where we similarly adopt their procedure.

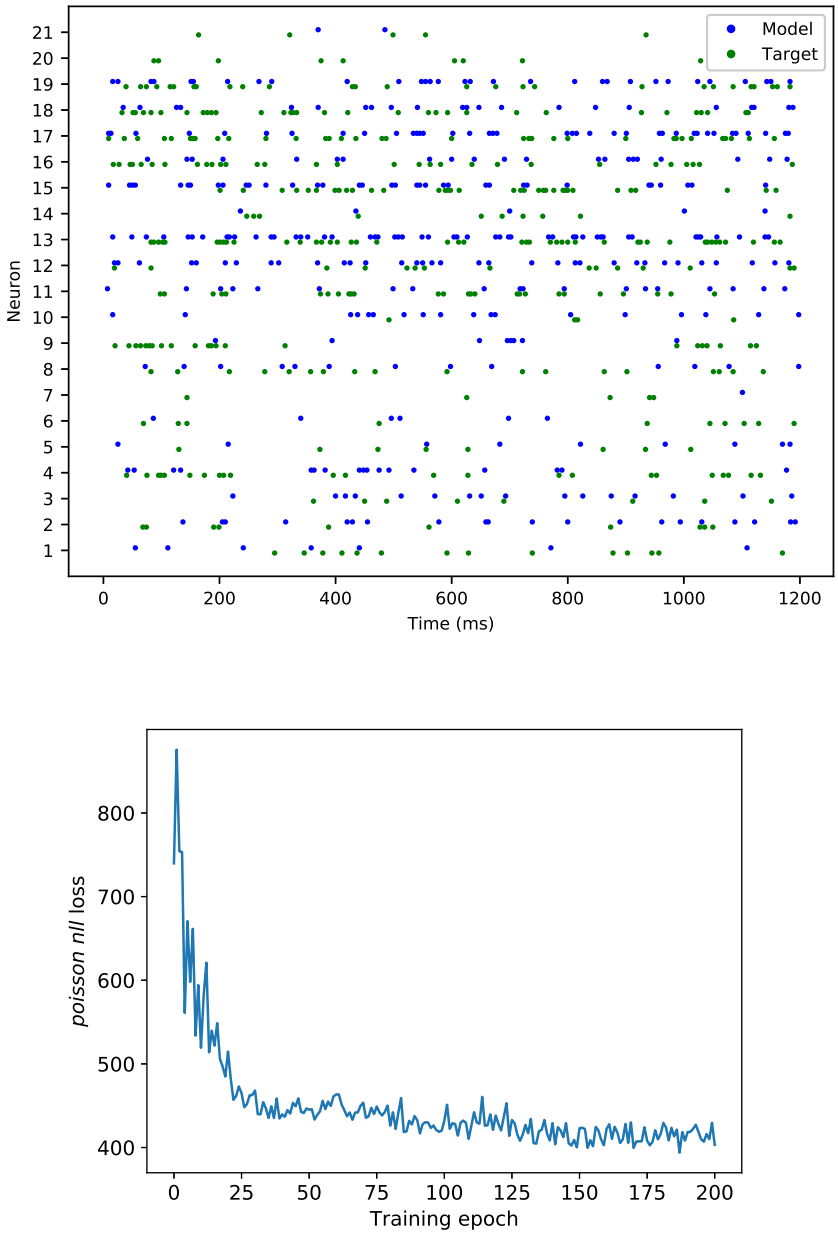

Figure 5.12: Target and fitted SGIF model spike trains, and loss per training epoch, with the Poisson negative log-likelihood as the loss metric (bottom).

Table 5.1: Neuronal correlations ρ for GBO, converged runs, network size N=4, for the SGIF SNN.

		ρ per neuron			
		$e_{L2/3}$	$i_{L2/3}$	e_{L4}	i_{L4}
WN	**Bernoulli**	0.09	0.14	0.21	0.07
WN	**Poisson**	0.27	-0.02	-0.02	0.04
OU	**Bernoulli**	0.23	0.35	0.14	0.14
OU	**Poisson**	0.22	-0.03	0.04	0.03

Table 5.2: RMSE for GBO, converged runs, network size N=4, for the SGIF SNN.

		RMSE per neuron				*Convergence*
		$e_{L2/3}$	$i_{L2/3}$	e_{L4}	i_{L4}	
WN	**Bernoulli**	8.9	11.5	19.9	29.0	50 %
WN	**Poisson**	11.5	14.6	19.8	25.0	80 %
OU	**Bernoulli**	8.7	10.8	16.8	28.6	70 %
OU	**Poisson**	11.2	13.3	17.3	25.5	85 %

For further results on this model class, including the geodesic NMF module similarity, this is included outwith this subsection, alongside the results for the other model classes.

5.3.3 Simulation-based inference

Using the best performing loss-metric as found in the GBO experiments, and also revealed by the parameter landscape plots, i.e. the firing rate loss metric, we performed SBI by using the Python-framework of the Macke-lab SBI.

In sum, SBI over a rate-based metric performed slightly worse for SGIF, and for the leaky integrate-and-fire models was on par for the less high-dimensional LIF model, but performed worse for the more complex GLIF model. As for retrieving the ground truth parameters, this is where SBI might supplement and/or outperform a GBO approach, as we calculate the full posterior over the model parameters, given the observations. Interestingly, however, even for the relatively low-dimensional LIF-model, the posteriors are somewhat skewed to the side of the ground-truth values. They do indeed generally give a reasonable estimate for the centers of the pdfs for the parameters for the LIF model, and to some extent for the SGIF model, but fails to do so for the GLIF model. The declining performance matches well with the increasing number of model parameters, which might be expected with an ABC approach, as posterior approximation requires increasingly more patterns with an increasing dimensionality - and likely even exponentially so. This shows that particularly for more complex models, which SNNs quite often are, the advantages and applicability of ABC may diminish - rendering GBO if not the only tractable approach, even a competitive wrt parameter inference.

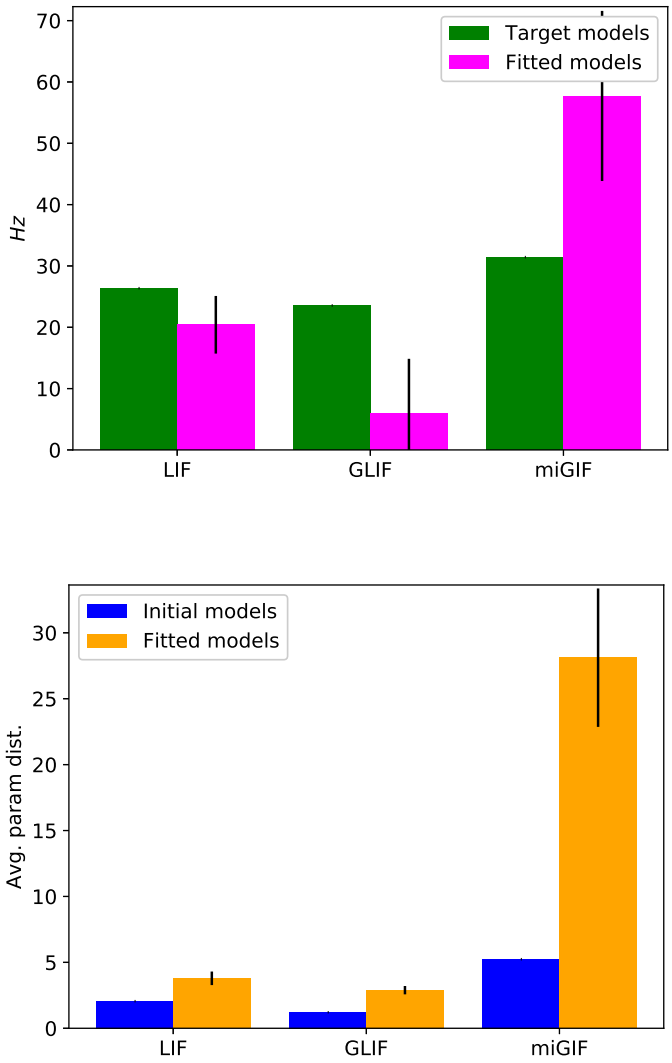

Figure 5.13: Fitted model rates across model types and loss metrics, and average parameter distance between the ground-truth model and both the initial and converged inferred models when using **SBI** for model inference. See figure 5.11 for comparison with GBO.

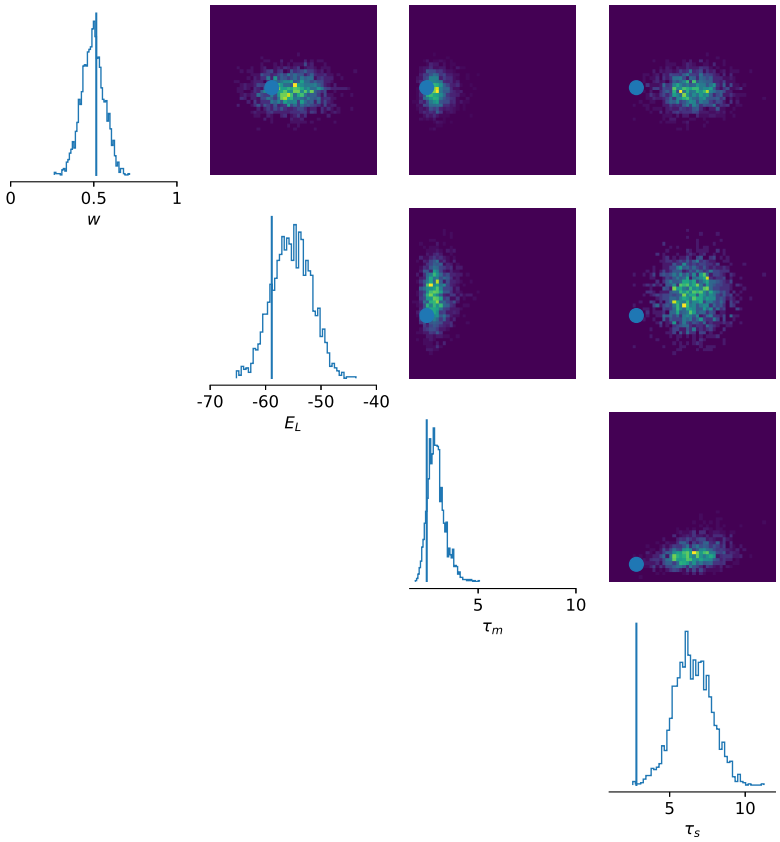

Figure 5.14: Mean posterior marginals between parameters for the LIF model class, fitted using the same ground-truth model and synthetic data as in the LIF GBO experiments.

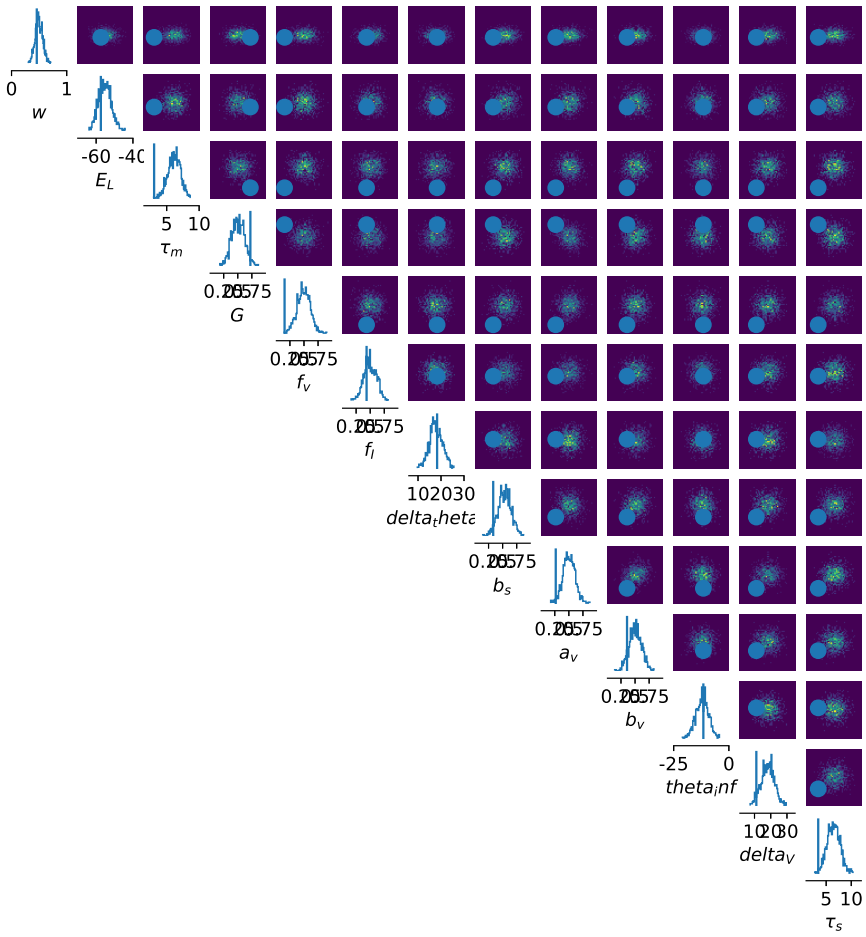

Figure 5.15: Mean posterior marginals between parameters for the GLIF model class.

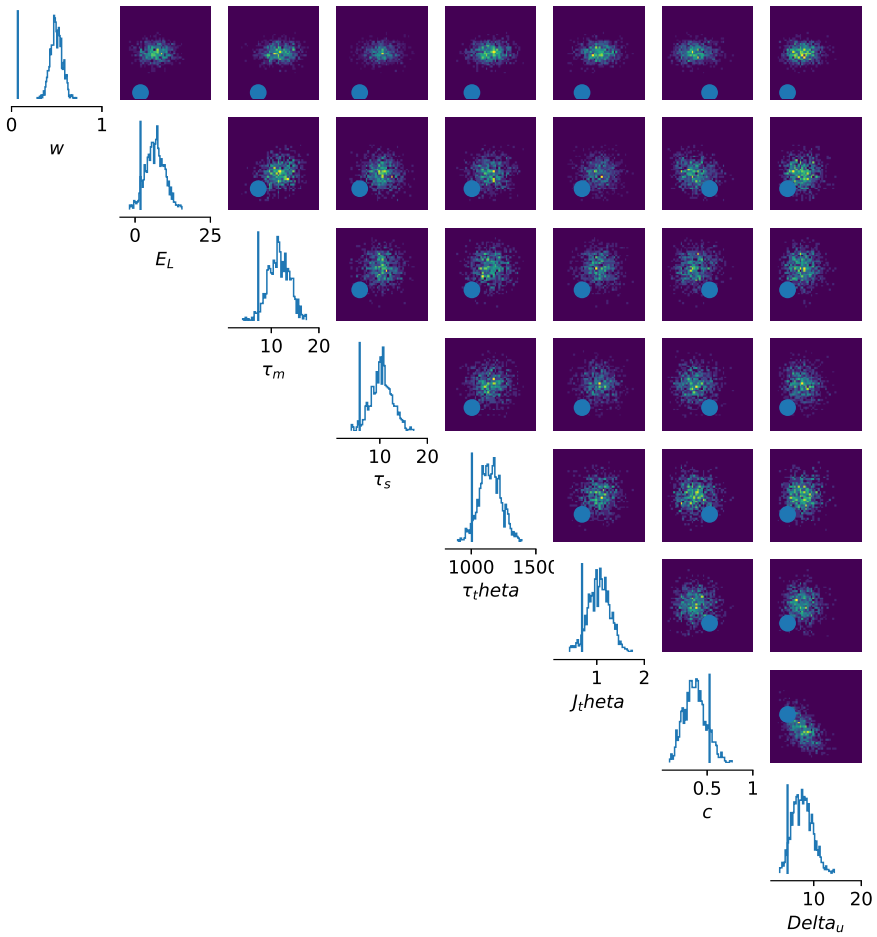

Figure 5.16: Mean posterior marginals between parameters for the SGIF model class.

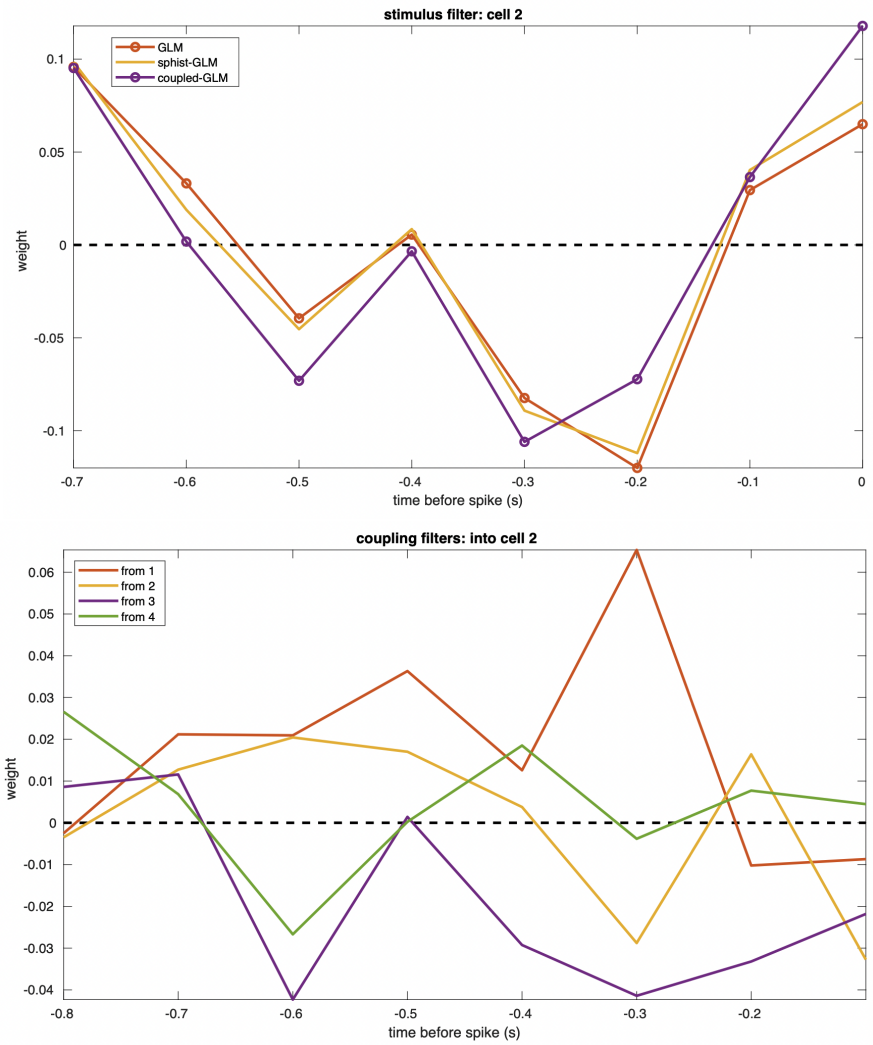

Figure 5.17: A stimulus response filter and coupling filters for a GLM fitted to a population level (N=4) SGIF model.

5.3.4 NMF analysis

We analyse the fits attained with the different approaches in order to assess to what extent the inferred models capture functional organisation of the networks that generated the target data. A high similarity of the factorised representations of neuronal co-activity indicates that the inferred models capture the spatiotemporal behaviour of the target model, which may suggest that the model can be used for further evaluation, such as ing probing to assess functional aspects or in testing or generating functionally related hypotheses. These are more relevant for biological data sets, of course.

As a baseline model for assessing how well we capture higher-order spike statistics, we implement and fit GLMs as previously described (see section 2.5.2), and also perform NMF (see figures 5.19, 5.20, 5.21) on the predicted spike trains of these models

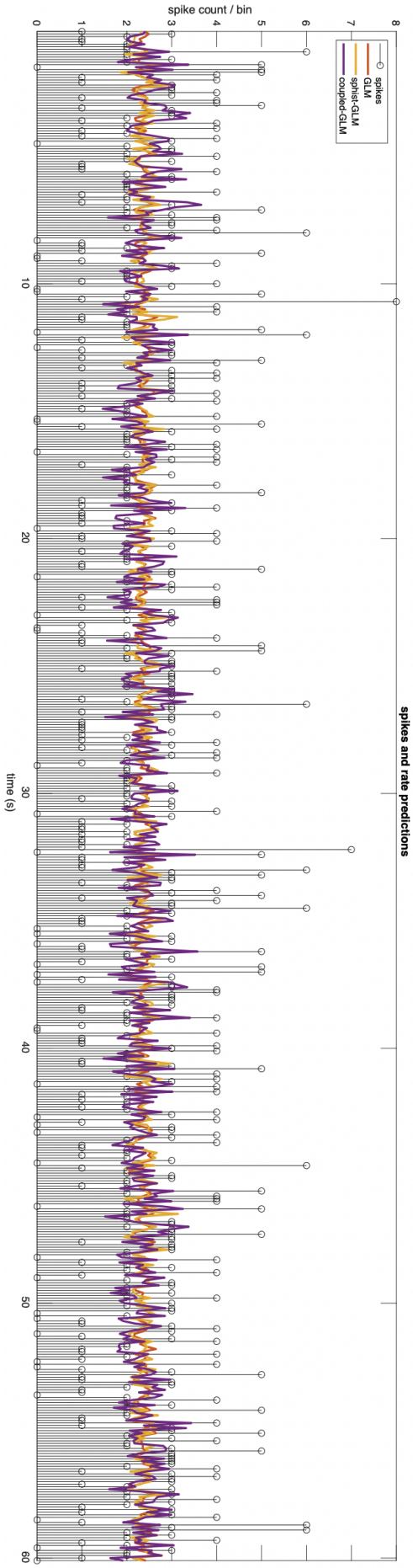

Figure 5.18: Spike train rate predictions for the fitted GLM where two of the response filters are illustrated in figure 5.17.

when perturbing them just as with the other models when generating their spike trains with white noise. A sample figure of one of the neuron response filters of the GLM is shown in figure 5.17, along with spike train rate predictions in figure 5.18. Due to the fairly low-dimensional nature of the LIF model, MLE for a coupled Poisson GLM fits almost perfectly to the generative target LIF model. Both SBI and GBO also perform fairly well for this model class in terms of the geodesic similarity (see figure 5.22). However, when considering the more complex GLIF model class, which may exert a wider range of spike patterns and behaviours, the GLM performs slightly worse than GBO using a rate-based metric. Interestingly, it performs just as well for both small and large SGIF networks, as the limit there seems to be the stochasticity introduced in this model class. It is however outperformed for smaller SGIF networks by both GBO and SBI using NLL minimisation, but performs slightly better than GBO for larger networks.

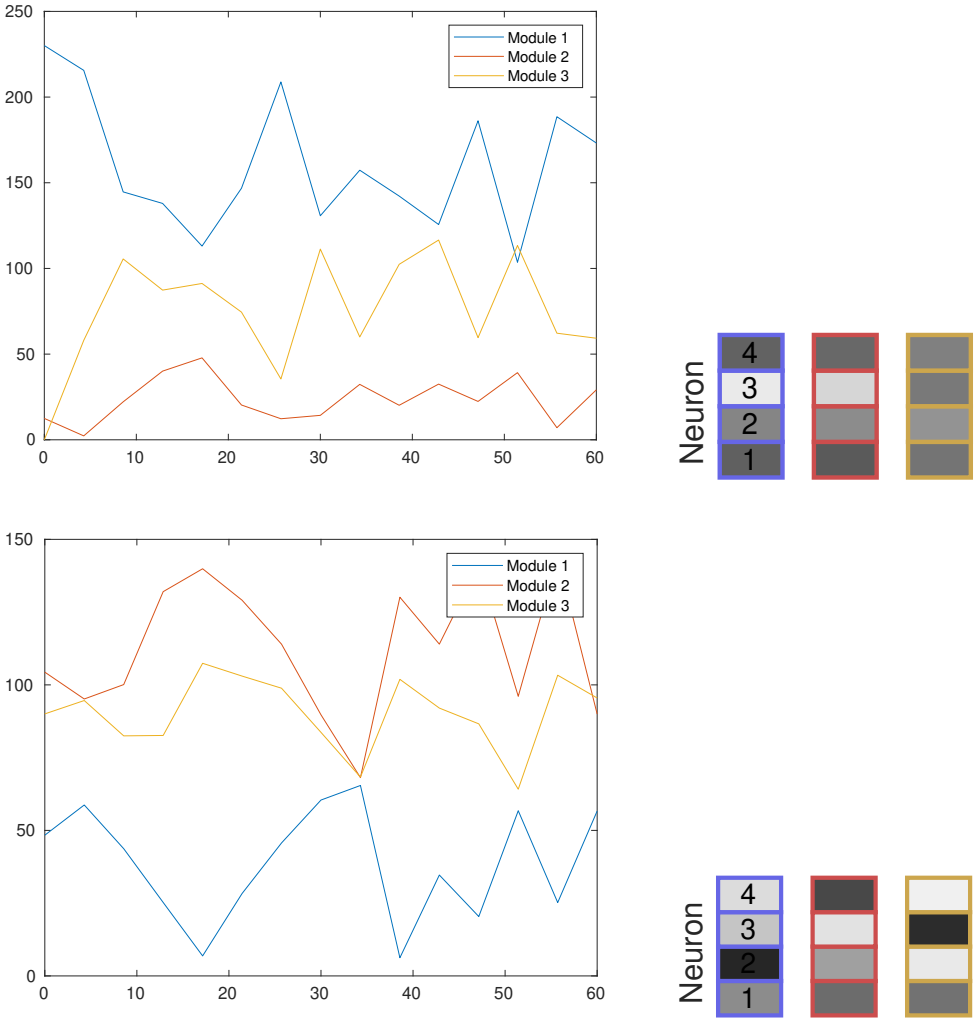

Figure 5.19: Activation coefficients (ACs) (top, seconds along the x-axis, and activation coefficient along the y-axis) for SGIF models, N=4, for target (top) and fitted models (bottom), using the Bernoulli negative log-likelihood (NLL) as the loss metric, along with the factorised NMF modules (right).

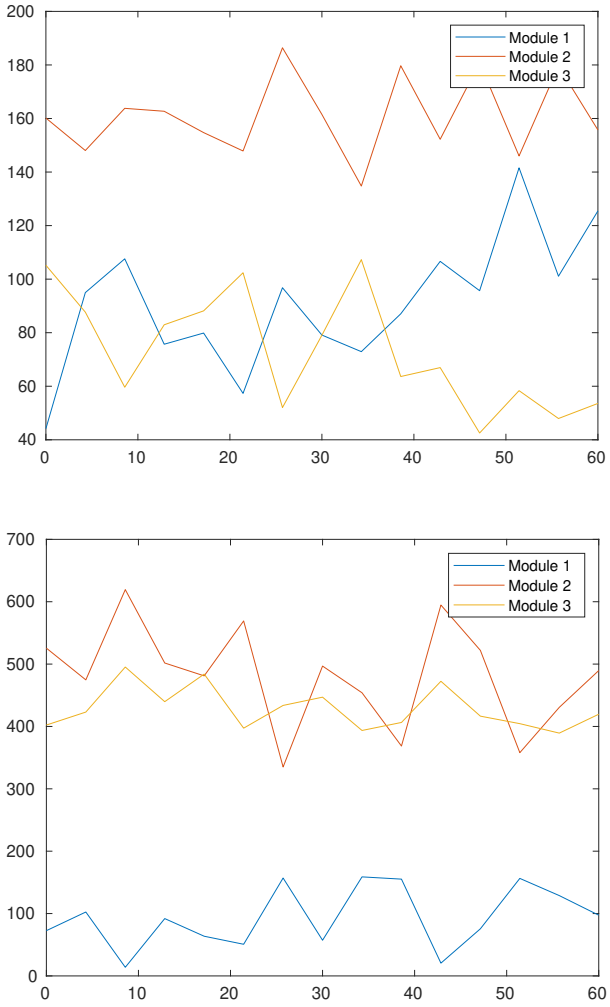

Figure 5.20: Activation coefficients (ACs) (top, seconds along the x-axis, and activation coefficient along the y-axis) for SGIF models, N=21, for target (left) and fitted models (right), using the Bernoulli negative log-likelihood (NLL) as the loss metric, with the factorised NMF modules plotted in figure 5.21.

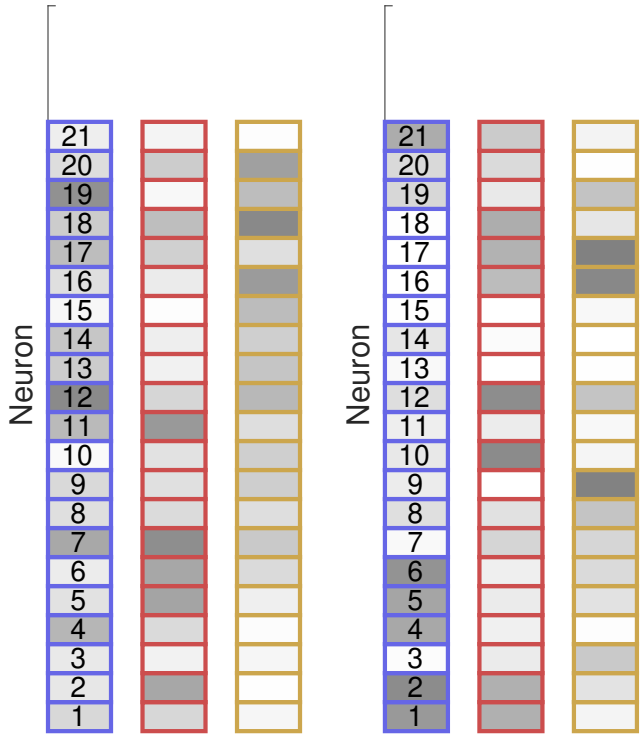

Figure 5.21: The NMF modules for the SGIF models, N=21, target (left) and fitted (right), using the Bernoulli NLL as the loss, as plotted (for the ACs) in figure 5.20.

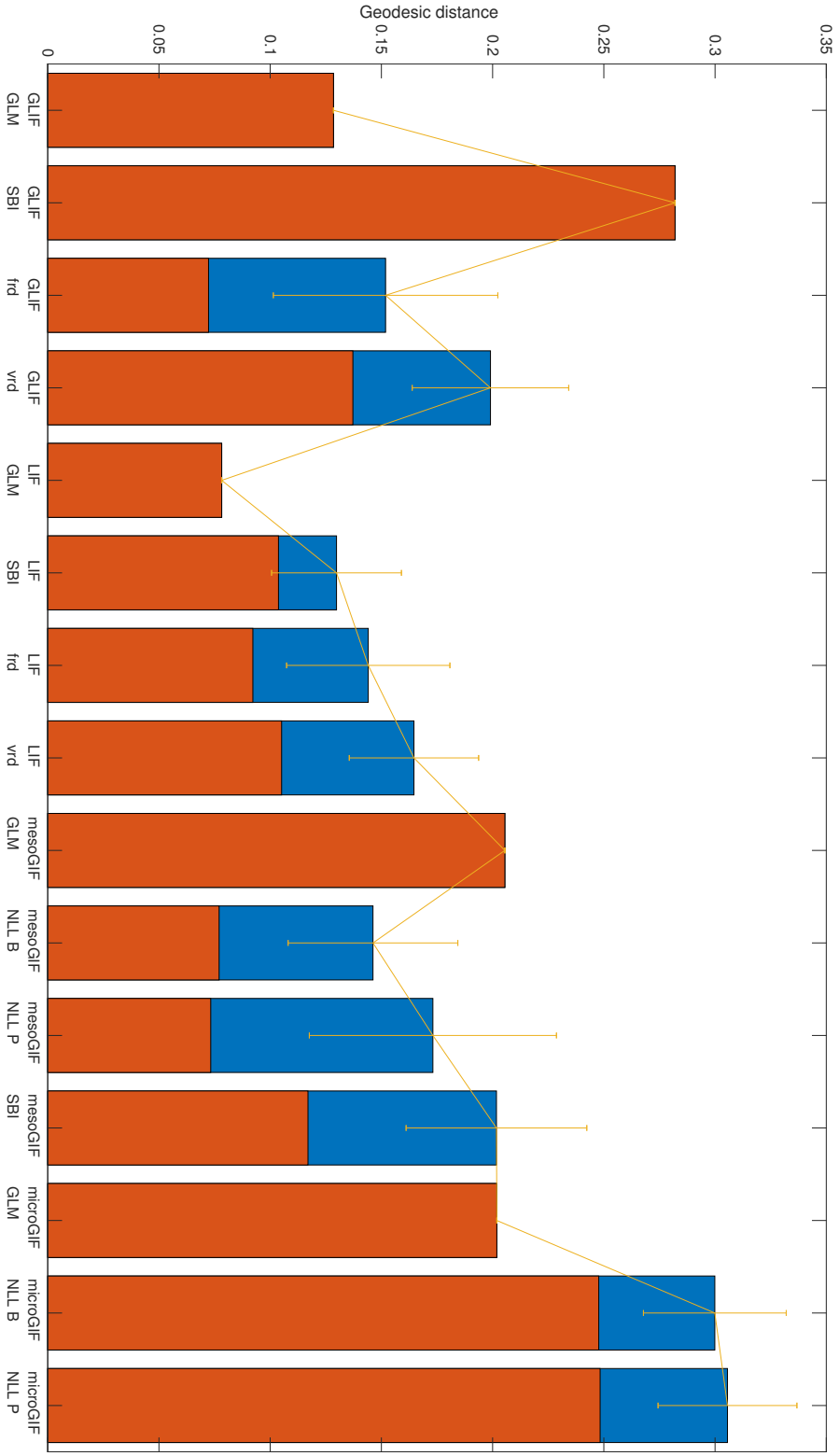

Figure 5.22: The geodesic distances between the NMF modules for all synthetic experiments.

5.4 Discussion

The results showing that for GLIF and SGIF SNNs, the fits are better than for LIF SNNs may be somewhat counter-intuitive, but may be explained by considering that although we introduce greater complexity via the higher number of parameters for the GLIF and SGIF models, their formulations are also more robust in terms of the behaviours they may exhibit. This may be further emphasised by considering our empirical data from trying to apply GBO to the Izhikevich model, discussed in section 3.1, in which we observed that fitting this model was highly challenging due to parameter regions for which model behaviour would be completely chaotic. This is due to that the model is a lower-dimensional, collapsed projection of a system designed to be able to exhibit a myriad of behaviours (namely the Hodgkin-Huxley [47] model), but it comes at the cost of introducing regions in this projected system for which impossible and chaotic behaviours emerge.

The mean posterior marginals between the parameters in figures 5.14, 5.15, 5.16 show that similarly as for GBO, and as illuminated by the parameter error landscape plots, we fail to retrieve the ground-truth parameter means. We are able to do so to some extent for the lower-dimensional (both in terms of parameters and number of neurons) LIF model, but cannot hope to meaningfully, i.e. with a strong link, interpret the values in a biological context for the inferred model.

The sample GLM plot shows illustrates that we may infer sensible stimulus filters and couplings using the MLE procedure (see section 2.5.2), and illustrates a predicted spike train using a few different GLM variants. This forms as previously discussed a baseline model that may be used when assessing whether our inferred SNN models may be more useful with regards to capturing higher-order statistics.

For evaluating whether we have the captured spatiotemporal structure of the spike train data, we perform NMF on the predicted as well as target spike trains for the different models. As illustrated in figures 6.6, 6.9, we may factorise three ensembles of co-active neurons into modules of varying activity. When evaluating their spatial similarity, we find a higher degree of similarity between the modules factorised from the spike trains predicted by the SNN models inferred using GBO. Further, we also note that the activation coefficients of these modules bears similarity for the corresponding (most similar) module to some extent, which further suggests that we may have captured some of the functional organisation of the neurons in the original spike train data. However, there is still a significant dissimilarity between the spatiotemporal

factorised ensembles, and interpretations of the model such be constrained correspondingly. Further, taking into account the aspects pertaining to the training signal, as well as the synapse model as discovered in the work extending [53] as presented in chapter 7 should be incorporated in order to try and better the similarity further.

Stochasticity and SNNs

There usually is a larger space for which performance given by the loss metric is fairly equal (as depicted by the error landscape plots), just requiring a different combination of parameters - i.e. there is no single fixed point or trajectory leading to it in the parameter landscape leading to a global minimum. One interpretation of this is that what we identify with learning algorithms for SNNs is not a specific configuration which captures the data set at hand, but a configuration which reaches a mode of behaviour that would allow it do produce the data. Interestingly, this is in line with the observed phenomena of representational drift whilst maintaining structural stability in the (functional) population patterns of activity [26] in that in this work neurons aren't constrained to one configuration (or mode of behaviour) in order to participate in an ensemble, capturing patterns stably on a population-level. If we want to take this one step further, it may even indicate that specific "fixed" neuronal representations when it comes to single-neuron activity (and synthetically its parametrisation) is not meaningful, or at least would not allow for the rich behaviour that we see in biology. One may imagine that if each neuron would only allow a fixed representation, this could quickly result in disastrous effects on the network-level, rendering the network incapable of capturing patterns of activity. For further reading on the robustness of networks of spiking neurons, the reader might be interested in the principle of tensegrity, which is eloquently described by [15].

One of the motivations for investigating SNN model inference with GBO was the high performance in parameter inference noted by [111], allowing for classification of cell type based on the inferred parameters. However, the aforementioned work pertains to a setting of single-neuron inference (and detailed neurophysiological data). In the literature on sloppiness, single neuron properties and connectivity may change significantly whilst the population-level activity remains stable [87]. Interestingly, in this view it might be expected to observe and attain sloppy regions in the error landscape on a network level - in contrast to on the single-neuron level. There are multiple ways one might interpret that networks are capable of learning tasks irrespective of

single-neuron changes: In one view, the network-level signal is not sufficient for calibrating the single-neuron behaviour and parameters - for this, some more fine-grained signal is required. Note however that this view posits that the 'goal' of a neuron is to learn a specific representation, or similarly specific responses or behaviours, which is in contrast with the observation of representational drift. Another way of viewing this that is more in line with the observation of representational drift is that neurons may change their responses, whilst yet successfully participating in e.g. performing a task on a higher (network and ensemble) level. One hypothesis that comes to mind in this regard, is regarding the driving force behind learning representations and tasks. Since neurons may change their responses naturally and more quickly than the network as a whole, which is naturally a lot stabler due to being the combination of all neuronal activities, resulting from all neurons' parametrisation and wiring, perhaps it is the representational drift in itself that allows for exploration, and the observed sloppiness in the error landscape on the network level that allows the relative functional stability (i.e. exploitation) throughout exploration and learning.

Multi-layer versus single-layer SNNs

While multi-layer nets may be crucial for non-linear function approximation this work only considers fully connected "single-layer" SNNs. Thus, testing the approach for multi-layer SNNs might give new interesting insights, and may also enable more easily capturing more complex data. Perhaps this is one of the key missing pieces for taking the step from the succesful optimisation of the models when fitting to a lower-dimensional signal, as studied in chapter 7, to fitting directly to target spike trains?

The spatiotemporal nature of SNNs, with the main measured effect being a function over a highly composite variable that has a high temporal dependency and variability, may complicate optimisation. In addition to being highly sensitive to initial conditions and other factors potentially both skewing the times of spiking, as well as the very mode of behaviour and spiking, input-output transformations, and the spatiotemporal signature of spike trains need to be handled in a way that is robust to these perturbations. A large number of our experiments revolved around trying to facilitate for ground-truth paramater retrieval. However, the results show that ground-truth retrieval is highly unlikely with the specified model definitions and loss metrics. By constructing error landscape plots in 5.2.1, we illuminate why this is. Interestingly, although local minima are inferred with regards to the parameter values, these configurations

may yet capture the spike train statistics of the target data, as demonstrated by applying the dimensionality reduction technique of NMF to compare the ensembles of coactive neurons in the spike trains predicted by the inferred models. Thus, the scalable approach of GBO may be leveraged in this regard, and bears potential for future work for automating and accelerating SNN inference work. The results highlight that the procedure is feasible by demonstration a working implementation and integration of various model classes, for both stochastic and leaky integrate-and-fire, and highlight that a stricter definition of the model perturbation scheme, as well as a more informative loss metric design, may be key to improving inference performance.

Chapter 6

Sleep regulation in the rodent brainstem

My initial research proposal outlines a research project where the goal is to meet research needs within the field of sleep regulation through computational modelling. In this chapter, we apply the inference methodology of this thesis pertaining to spike based SNN inference using GBO to biological spike train data, recorded from the brainstem of mice during different vigilance and sleep states, and assess to what extent the higher-order NMF modules, which correlate with the brain state, are captured in the inferred models.

More specifically, the aspiration was to model neurons of the pedunculopontine and laterodorsal tegmental areas within the brainstem during different brain states, based on their neuroanatomy as described, even though scarcely, within the literature, and based on in vivo data from these brain areas [43, 116, 86, 73, 33]. Further, an overarching research goal was to be able to capture the emergence of neural ensembles as identified in vivo in the model by using non-negative matrix factorisation (NMF) [100, 99, 84]. To this end, we have implemented various types of SNN models, a framework for gradient based optimisation using Adam in a modular way, compatible with any differentiable model as well as loss metric, all in PyTorch. This work is presented previously in this dissertation. Further, the hand-engineered target models that were used to test the methodology whilst also maintaining the ground truth generative model parameters in chapter 5 were designed to have neuronal rates and parameter values resembling a mixture of excitatory and inhibitory neurons, keeping both biological data and the particular sleep data in mind. More generally, computational modelling and spiking neural network (SNN) inference covers several research needs within the

field of sleep research and the synthesis of neuroscience and its computational counterpart in that it addresses model scarcity, as well as a methodology for accelerating modelling by inference through gradient-based optimisation [43, 52, 105]. This was the foundation for sparking my interest in a methodological project in the first place, with the goal of automating biologically relevant neural network model inference, and more specifically to research both (1) the current state-of-the-art on SNN inference, and (2) leveraging ML based methods of gradient descent and optimisation for SNN inference [52, 76, 109, 66].

While it is evident that the extent to which GBO may be used to infer an exact SNN model based on spike data from the findings in the previous chapter, the results nevertheless show that the approach may infer models that capture the higher-order statistics and functional organisation fairly well, and even better than a solid baseline model, namely the GLM. As the ABC approach of SNPE quickly becomes intractable, and is in fact intractable for the network size of the biological data studied in this chapter (at least for a reasonable execution time, with limited computational resourced), we have employed our GBO procedure to biological spike train data for LIF, GLIF, and SGIF models, and similarly to in the previous chapter, we assess the goodness of fit via the attained model rates, loss, and geodesic NMF module similarities.

6.1 Background

Sleep is widespread across different animal species, crucial to mental functioning [12, 119]. However, why and how we sleep, remains to be understood, with a rich literature pertaining to different aspects of sleep, its regulation, potential functioning, and different pathways [9, 33, 12, 44, 120, 117, 45, 77, 63, 28, 18, 41, 30, 25, 122, 43, 59, 96, 19, 37, 27, 2, 16, 67, 71, 80, 86, 117, 35, 121, 42, 102, 85, 116, 32, 23, 14, 114, 48, 49, 62, 73]. Some models exist that seek to capture the phasic nature of sleep, but mostly at an abstract level. There is as such a need for detailed models in the field, with no current models encompassing direct biological parallels. Addressing the need for modelling in the field, and seeking to illuminate how we sleep, we propose to use a set of recently combined methodologies that allow us to infer neuron-level spiking models based on recorded spike trains. The methodology allows for incorporating both existing knowledge from the ML domain, and lays the groundwork for using GBO for SNN inference, illuminating the extent to which this may be done with the proposed methodology, as well as highlighting key issues that should be adressed in future mod-

eling work along this strand. The outlined algorithmic approach applied to spike train data has, to the best of our knowledge, not been previously explored. This may be due to the high dimensional parameter search space, which also complicates making inference tractable and sufficient.

6.1.1 Sleep stages

When it comes to the phasic nature of sleep, one seminal yet simple model for describing this is a three-process switch or flip-flop model, as described by [9]. This describes how a combination of a homeostatic drive, the circadian (24-hour) rhythm, and an ultradian rhythm may facilitate and regulate sleep, wakefulness, and the different types of sleep (namely NREM- and REM-sleep), without detailing the pathways involved. This begs the question of whether an inferred SNN be used as a starting point to create such a model, or whether it can be determined to be involved in one or more of the three hypothesised regulatory mechanisms.

6.1.2 PPT/LDT

Many areas have been thoroughly researched with regards to their role in sleep regulation, such as the locus coeruleus (LC) [71, 27, 12, 60], which is fairly well understood. However, other areas which have been shown to be involved in sleep regulation are less or poorly understood. One such area is the pedunculopontine and laterodorsal tegmental areas (PPT/LDT) of the brainstem, with its circuitry containing a myriad of inputs and outputs [85, 12, 117, 44]. Could these be modeled by inferring models that reproduce similar spiking activity by using GBO with data recorded by silicon probe insertion into the area? If so, to what extent can we reverse-engineer the neural activity, classify the neurons into types, and illuminate the pathways involved, such as whether the neurons facilitate REM-sleep or not - i.e. are REM ”ON” or ”OFF”? While it is limited what we can capture in the current model, one thing that is out of scope for this thesis, but would be straightforward to implement, is designing input that more closely resembles that of during REM- and NREM-sleep. This could then be presented to inferred models, and if captured by the inferred model, the ensembles could tell us something about state-regulation.

6.2 Biological target data

The biological data that we were granted access to is from a silicon probe recording (8x4 Buszaki probes) from the PPT/LDT area of the brainstem. It is analysed in published research, in which evaluation and assessment through rigorous analysis and classification and prediction of the future brain state given the neuronal signals, as well as NMF modules, was performed [116].

We here replicate much of the results of [116], confirming that NMF ACs were more indicative of future brain state than HPC signals, which was a novel finding and insight when presented in the original paper. This also means that the data should contain information that if captured in a model may illuminate the functional dynamics associated with sleep regulation in the area (PPT/LDT). Further, we also tested and confirmed that single-neuron signals were slightly better predictors. However, as the gain is only marginal or small it also suggests that the ensembles are indeed good representations of the spike activity, and thereby functional ensembles, as argued in the original paper. Therefore, we perform NMF on the predicted spike trains of our inferred models, and compare the factorised modules attained with those of the biological target data. Note that the quality of the spike data in terms of neuronal activity and non-silent neurons varied across the different experiments. So did classification performance with the aforementioned variability. Therefore, we have focussed on the two experiments in which most neurons displayed activity, and the state classification performance was consistently high (see figures 6.1 and 6.2). These were also the two only data sets for which three NMF modules were needed in order to account for at least 75 % of the variance in the original spike data with the factorised matrices.

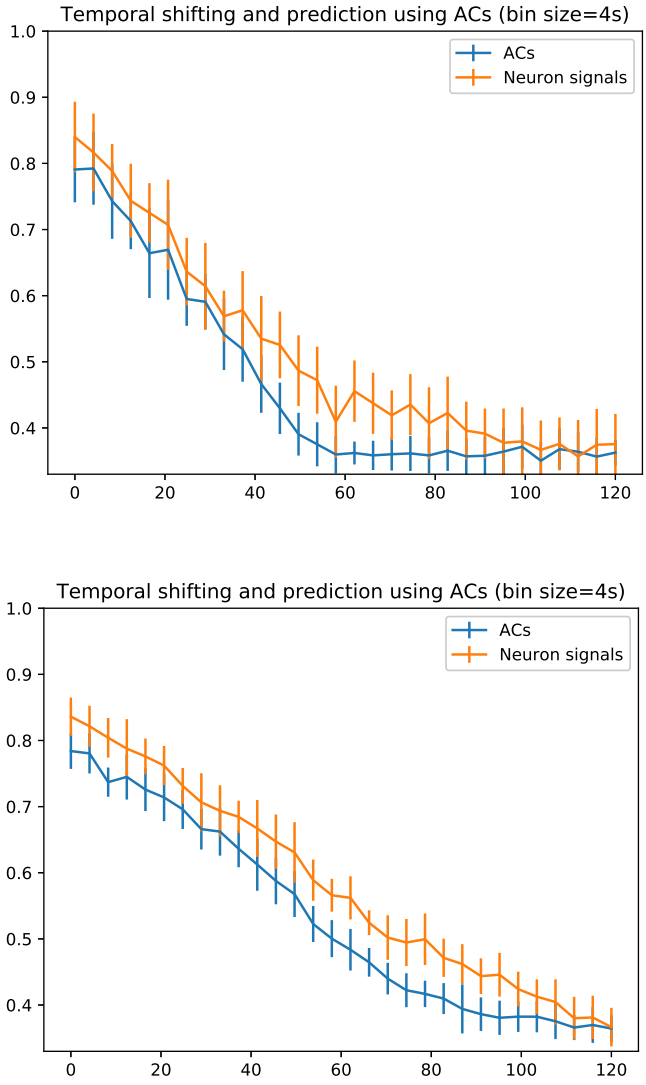

Figure 6.1: Prediction accuracy (second axis) by the time preceding the predicted state in seconds (first axis). The biological data often contained silent neurons, with the most consistently active signals and thus 'well-defined' data being particularly the data sets for experiments 4 and 6.

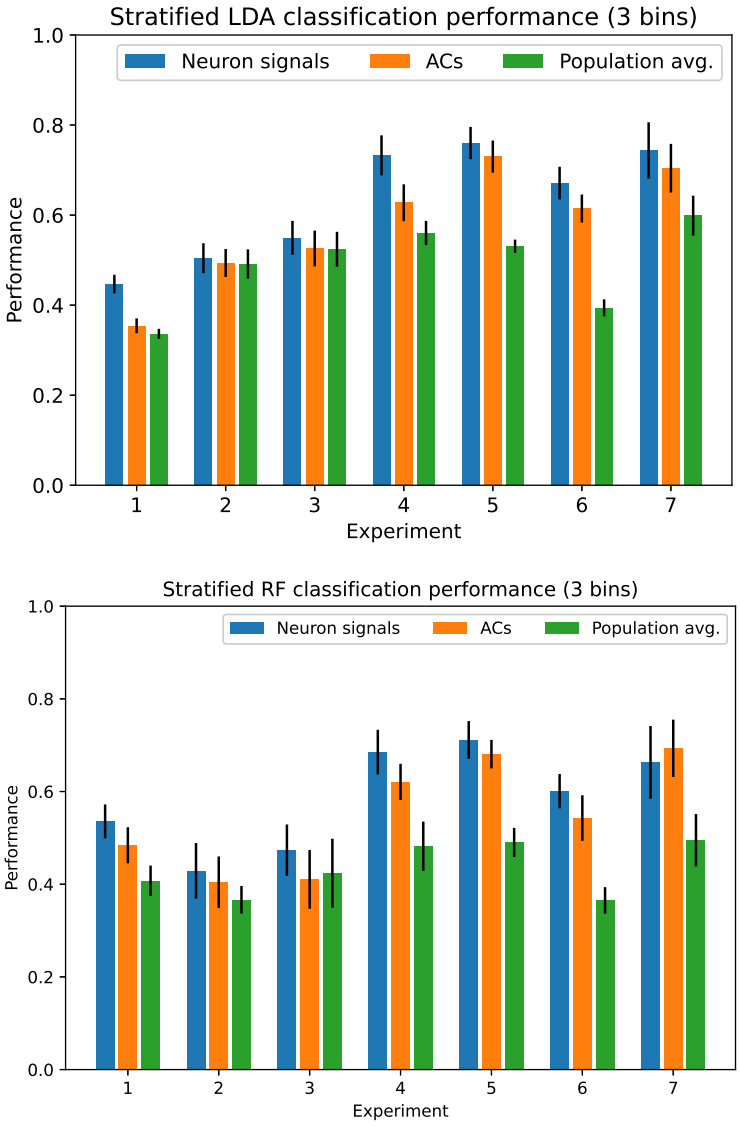

Figure 6.2: LDA and random-forest classification over the different experiments.

6.3 Inference

As mentioned above, we have focussed on the two experiments with consistently high 'quality' in terms of neuronal activity and vigilance state prediction performance. Further, we adopt the differentiable SGIF-model as described in previous chapters [92], using GBO for direct neuron-level SNN inference by negative log-likelihood minimisation, assuming either a Bernoulli or Poisson distribution of the spike train. For NMF performance evaluation between NMF modules (such as illustrated in 6.7, 6.10), we also fit GLMs (section 2.5.2) as a baseline comparison model in NMF module similarity assessment (with the similarity distance plotted in 6.8, 6.11).

For each experiment, we fit $N = 20$ models as in the other setups to each experiment, for each model type and loss function, pseudorandomly and uniformly initialising the model parameters before inference, with a different random seed for each experiment. We here report the results for each configuration for the two data sets, in addition to illustrations of the fitted GLMs (figures 6.3, 6.4).

The results show that we may infer a model where the rates match fairly well for each neuron, 6.5. Interestingly, the results also show that assuming a Poisson distribution and using this to optimise over the negative log-likelihood results in a significantly lower loss than when assuming a Bernoulli distribution: Student-T test (statistic=9.87, $p_{value} = 6.23e^{-05}$). This is in contrast to the work on synthetic target data, in which the best performance was attained by assuming a Bernoulli distribution when optimising over the NLL. One possible explanation for this is that true biological data may be of a more complex nature, which is better described by a Poisson- rather than a Bernoulli-assumption.

As the number of nodes is slightly higher for these data sets, the SBI ABC approach outlined in chapter 5 is intractable for the data at hand, and is thus out of scope for the biological data.

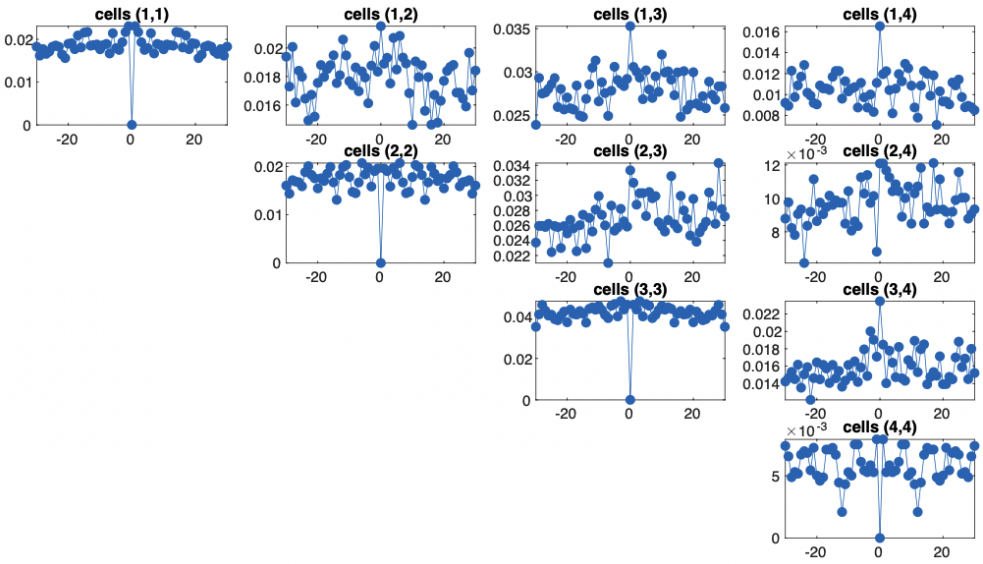

Figure 6.3: Response filters for the GLM, here fitted to experiment 5 data, with corresponding sample node filters shown in figure 6.4.

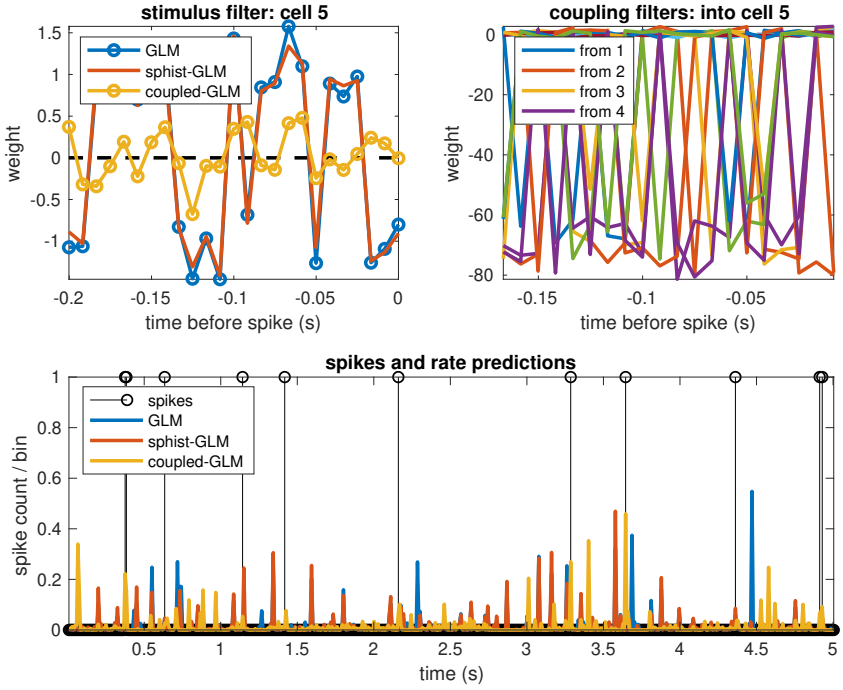

Figure 6.4: Filters for an arbitrary sample GLM cell and predicted spikes and rates for the GLM with different couplings for experiment 5 (the blue - ”GLM” is the full GLM as used for the baseline in the NMF analysis comparison).

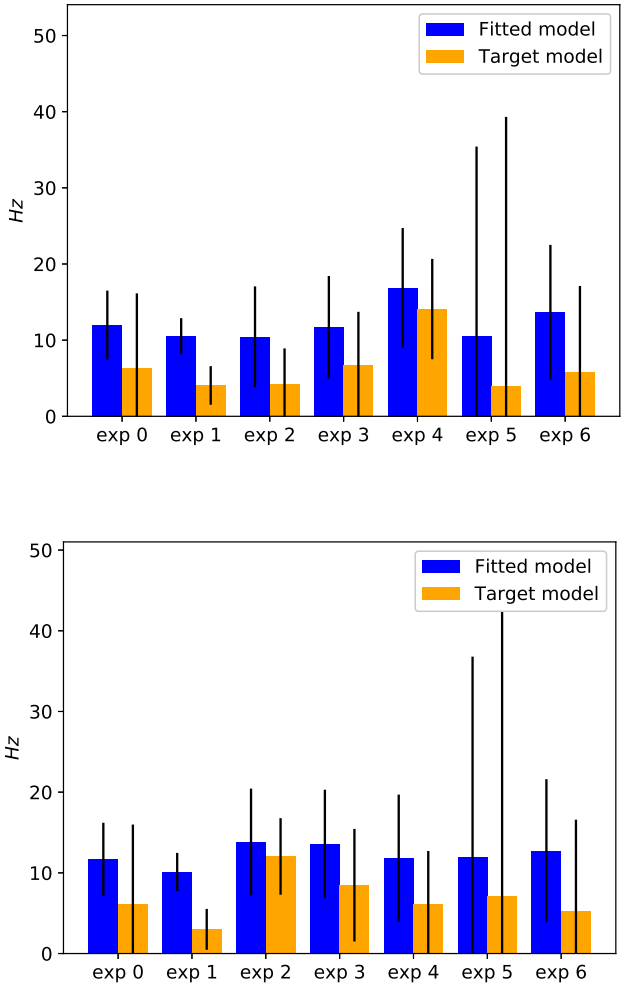

Figure 6.5: Inferred model firing rates per data set for the SGIF model, using white noise with a fixed rate as input perturbation, Bernoulli NLL (top), and Poisson NLL (bottom).

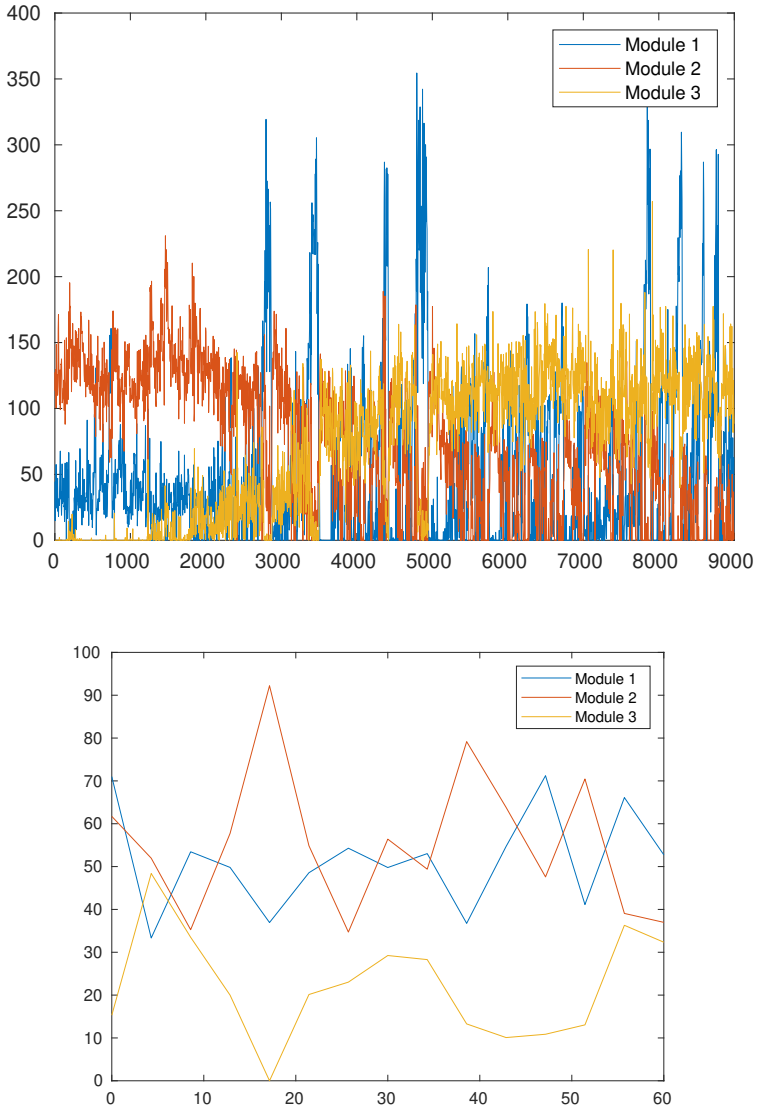

Figure 6.6: Activation coefficients (ACs) (second axis) by time in seconds (first axis) for experiment 5 for the target (left), and SGIF fitted model (right), using the Bernoulli NLL as the loss metric.

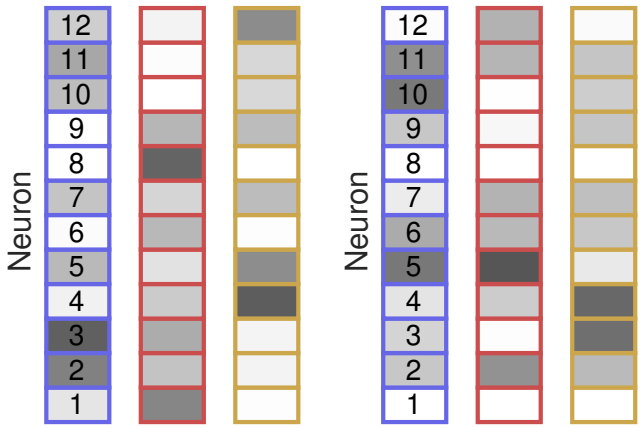

Figure 6.7: ACs for experiment 5 for the target (left), and SGIF model fit (right). These particular modules have a geodesic similarity of approximately 73%.

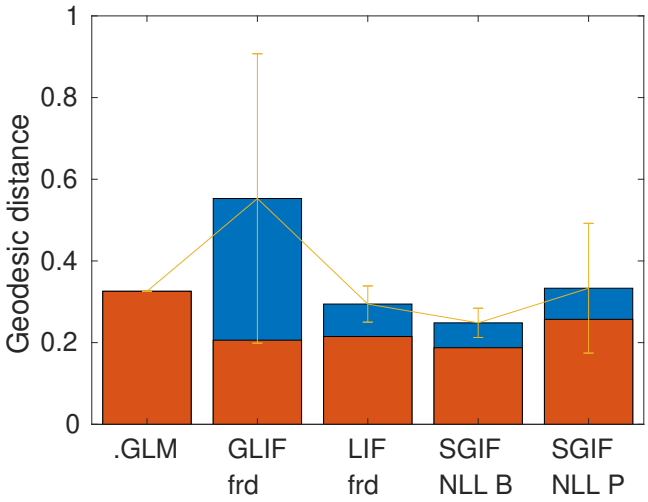

Figure 6.8: Geodesic distances across model types and loss functions for experiment 5.

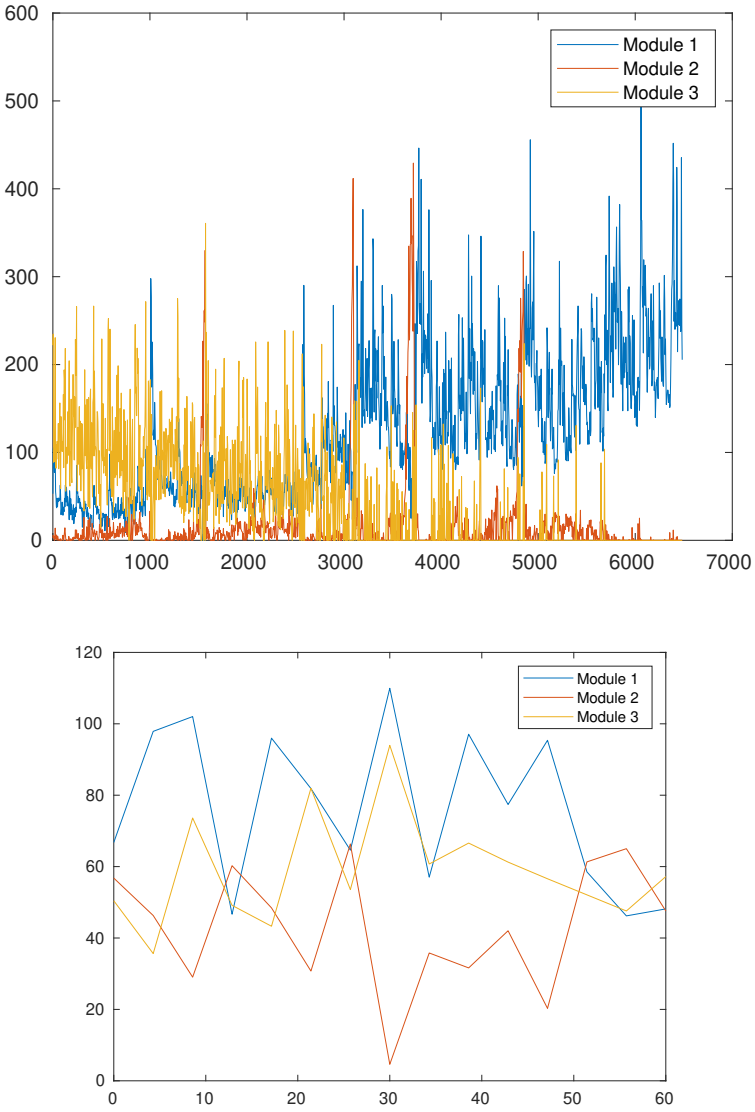

Figure 6.9: ACs (second axis) by time in seconds (first axis) for experiment 7 for the target (left), and SGIF model fit (right), using Poisson NLL minimisation.

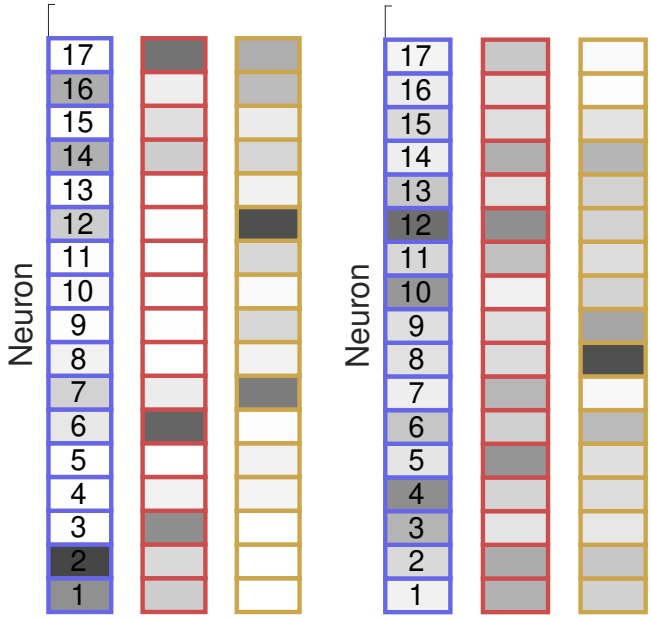

Figure 6.10: NMF modules for experiment 7 the target (left), and SGIF model fit (right). These particular modules have a geodesic similarity of approximately 69%.

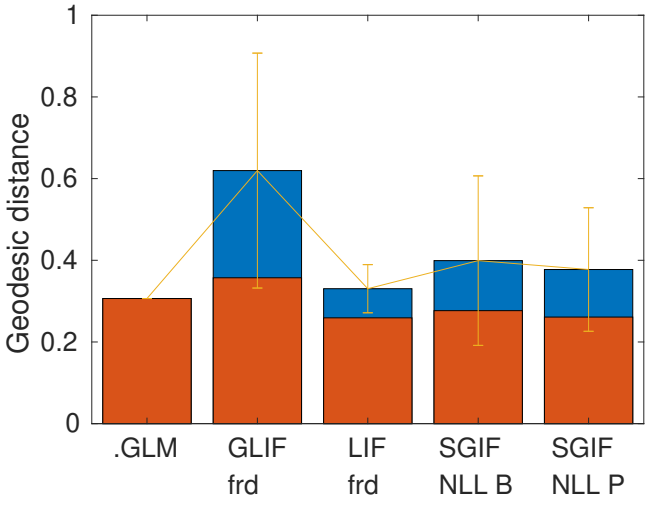

Figure 6.11: Geodesic distances across model types and loss functions for experiment 7.

6.4 Discussion

This chapter demonstrates the scalable and modular GBO approach applied to biological spike train data, for which it is still a tractable approach, as well as MLE for GLMs to form a baseline for the NMF similarity evaluation. By doing so, it provides a starting point for automatic model inference of SNNs using GBO by demonstrating one way of implementing the approach. It is our hope that this may aid in accelerating SNN inference research, and potentially have a positive impact on computational neuroscience modeling research.

As for the results, it is limited what can be hypothesised about biology from the inferred models. Due to the local minima of the parameter values, parallels pertaining to which neuron types and neurotransmitter pathways that we might be looking at would be highly speculative. On the other hand, the higher the geodesic NMF similarity, the better the inferred models capture and represent the spike train statistics. As such, it would be interesting to design different perturbation schemes, which represent and are thought to correspond to different sleep and stimulus states, and assess to what extent the fitted model outputs change accordingly, and comparatively with the biological data.

The fact that the error landscapes for the parameters are ambiguous explains why the spike trains are not as correlated in the models with GBO as when using ABC/SBI (as reported by [92]), as GBO numerically and iteratively here updates parameter values in parallel, thus doing so slightly independently of the other parameters. This may suggest that sequential parameter inference is required for better convergence when applying GBO for SNNs in this setting. However, this takes away the key goal of scalability of the inference algorithm, and would render ABC a better candidate for the job, as we then estimate a full posterior over all parameters dependently - which is also the explanation for a higher correlation and better produced target data by the ABC procedure; as this might be capturing the parameter-dependencies better. Authors have also reported that reproducing realistic data breaks down when drawing from the inferred posterior independently for each parameter, which we also empirically verified in our experimental setup, by using the aforementioned SBI-framework.

Chapter 7

Gated subthreshold synaptic currents and continuous target signals

While we demonstrated the feasibility of gradient descent based SNN inference by application in the previous chapters, we also showed clear limitations to the procedure, particularly relating to being prone to converge to local minima due to the error landscapes formed by the inference signal and configuration. Here, we revisit the synapse model used in our SNN models, and redesign the input-output scheme by adopting the approach of [52]; namely by implementing subthreshold continuous synaptic currents which are used as the model output in conjunction with a readout transformation weight matrix. We first replicate the results with non-leaky integrate-and-fire (NLIF) SNNs, and then extend the research to LIF models, and find that indeed 'exact' gradient calculation (i.e. using non-leaky neurons) is not a hard constraint for optimal auto-encoding and general predictive encoding task performance. Because of this novel insight, we hypothesise that it is the synapse model in combination with the lower-dimensional and continuous target signal that enables optimisation convergence due to greatly constraining the parameter space - and also in that the signal is far less noisy, also resulting in a more well-defined and traversable parameter landscape.

7.1 Tasks

We adopt two of the tasks performed in the original study [53] in order to replicate, extend, test and compare our results for both the NLIF and LIF models. These wwo tasks of input transformation; (1) auto-encoding the input of two sine-modulated white noise inputs, essentially predicting the input as output, and (2) a more general predict-

ing coding task in which a linear transformation of the inputs formed the composite target signal. The equations defining the signals in the two tasks are outlined below.

7.1.1 Auto-encoding

This task is the simplest of the two, and only requires the pseudorandomly initialised model to 'learn' to encode the output as the input itself.

$$\frac{do}{dt} = \frac{-o+i}{\tau_f}, \tag{7.1}$$

where o is the output signal, i the input signal, and τ_f a time-constant for the output adaptation, essentially resulting in a filter/less sensitivity to very rapid input changes in the encoding, i.e. a slightly smoother output signal. However, this constant is kept quite low.

7.1.2 General Predictive Encoding

This task is slightly more complex than the previous, as it requires inferring a composite linear transformation encoding of the input.

$$\frac{do}{dt} = \frac{-o+i+Ao}{\tau_f}, \tag{7.2}$$

where A here denotes a coefficient matrix which transforms the output signal in a less straightforward way than simply encoding the signal itself.

7.1.3 Loss

For the loss function, we used the same function as in the original paper;

$$\mathcal{L}2(M,T) = \frac{||M-T||_2 + \lambda||M||_2}{2} = \frac{\sum\sqrt{(M-T)^2} + \lambda(\sum|M|)}{2} \tag{7.3}$$

where M is the readout of the model, T is the target signal, and λ is a regularisation constant.

7.2 Setup

For each task, we pseudorandomly initialise the model to be fitted, drawing each parameter value uniformly from intervals of non-extreme values. Pseudorandomly here

denotes that we deterministically set a random seed for each experiment and model initialisation, which is then used for the random number generation. Note that this then also affects the input and output signals. We do this for $N = 20$ models for each task and model type, and report the results following in this chapter, including a near 100 % convergence rate. For the fast synapses, we set the time constant τ_{fast} to 1.5ms for numerical stability during optimisation, instead of $\tau_{fast} = 1$ms with double floating point precision as reported in [52], since we found this to be sufficient for numerical stability and convergence.

As for the optimiser, we used Adam, and set the learning rate $\alpha = 0.01$, and used $\lambda = \frac{1}{N} \approx 0.00333$ for the regularisation constant.

7.3 Results

The tasks were replicated first for the NLIF model, and then tested on a leaky model type, by incorporating the synapse model into a LIF SNN. For both tasks, we observed convergence in every experiment, with some experiments temporarily wandering up to a higher loss during gradient descent - however, this quickly settled into a lower loss than prior to the increase in error - signalling a successful traversal of a 'peak' in the error landscape.

The error landscapes for the different parameters illuminate that they are quite similar for leaky and non-leaky IF SNN models, as illustrated in figures 7.7, 7.8, 7.5, 7.6. In fact, they show that it may be just as well-defined for a LIF SNN as for a NLIF SNN, illuminating the potential for using this synapse model also for leaky SNN models.

We have included further results in B, should the reader wish to study further illustrated results, and for consistency.

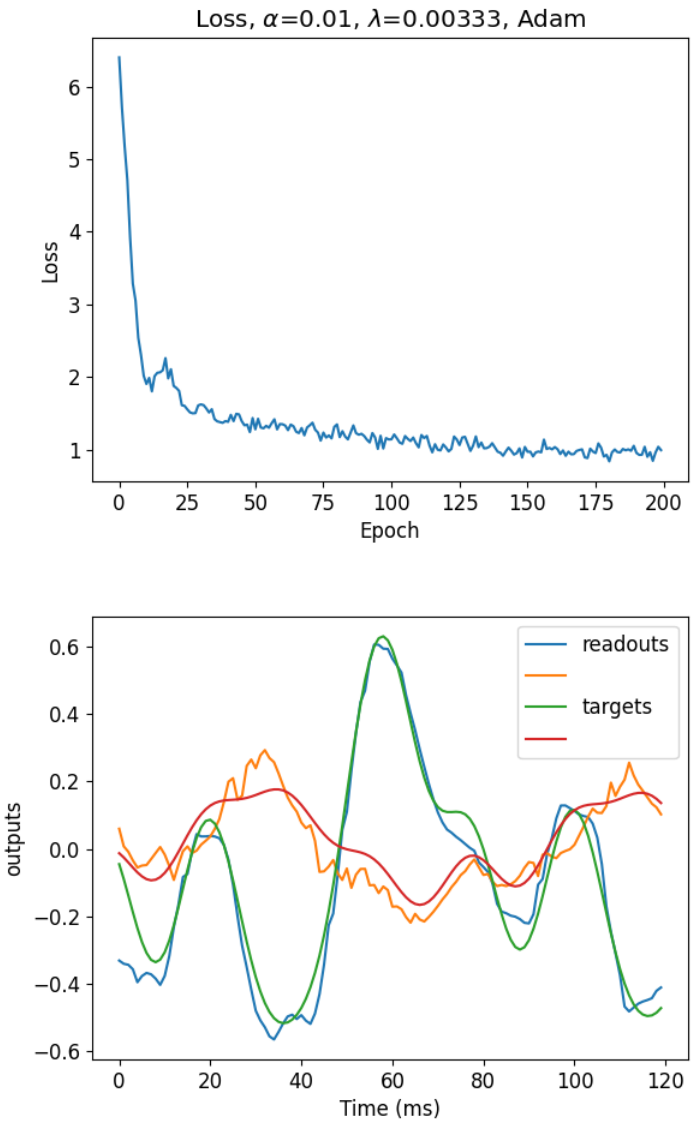

Figure 7.1: Auto-encoding with a NLIF SNN for a sample experiment, with the loss per epoch on the left, target and readout signals on the right.

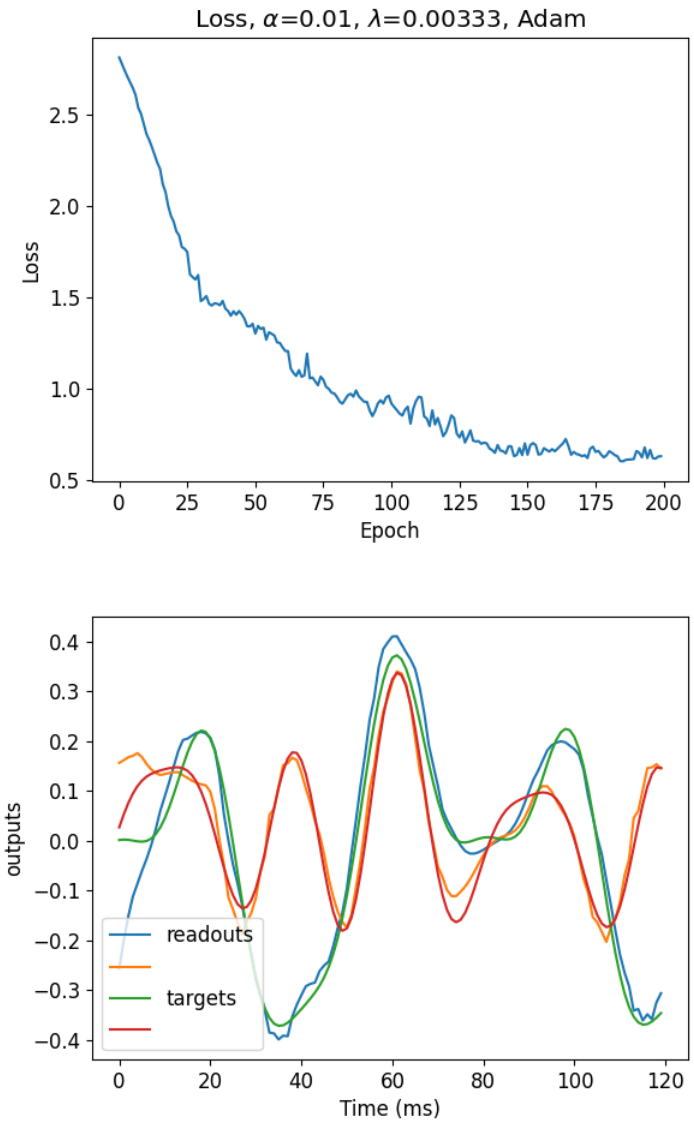

Figure 7.2: Auto-encoding with a LIF SNN for a sample experiment, with the loss per epoch on the left, target and readout signals on the right.

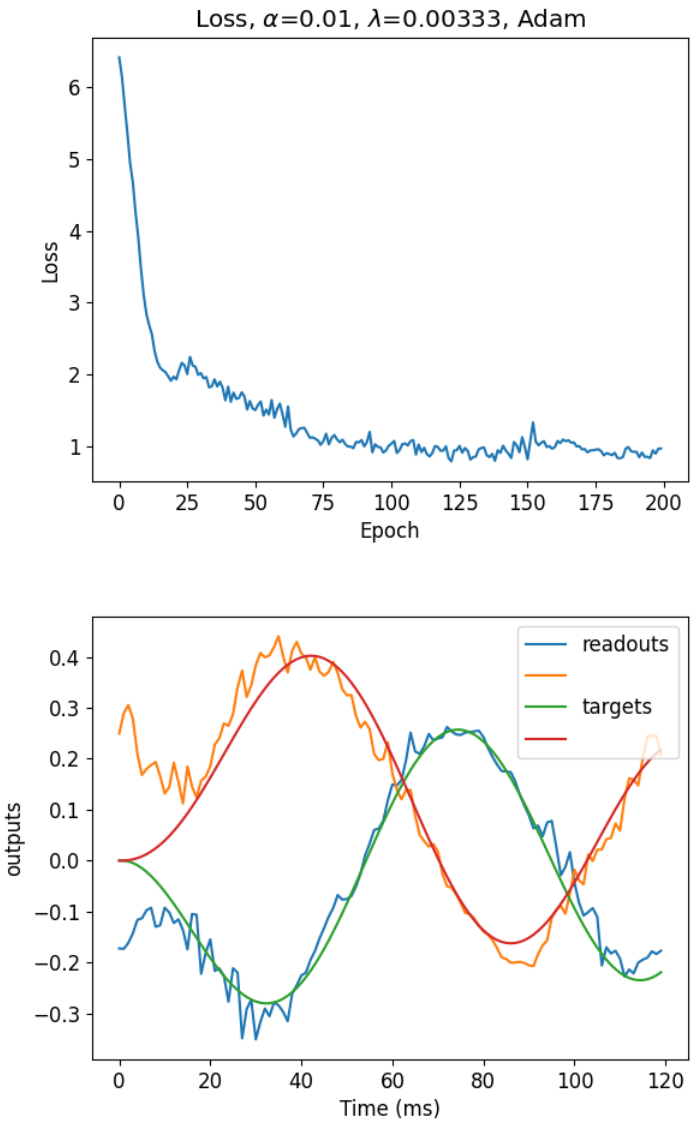

Figure 7.3: General predictive encoding with a NLIF SNN for a sample experiment, with the loss per epoch on the left, target and readout signals on the right.

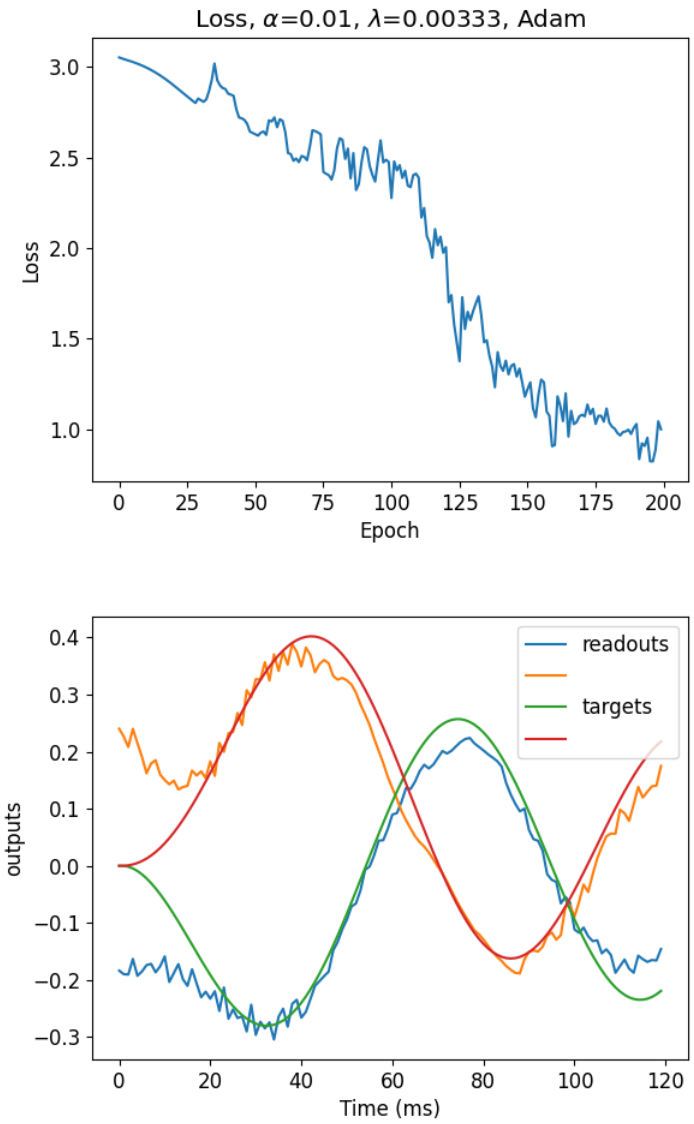

Figure 7.4: General predictive encoding with a LIF SNN for a sample experiment, with the loss per epoch on the left, target and readout signals on the right.

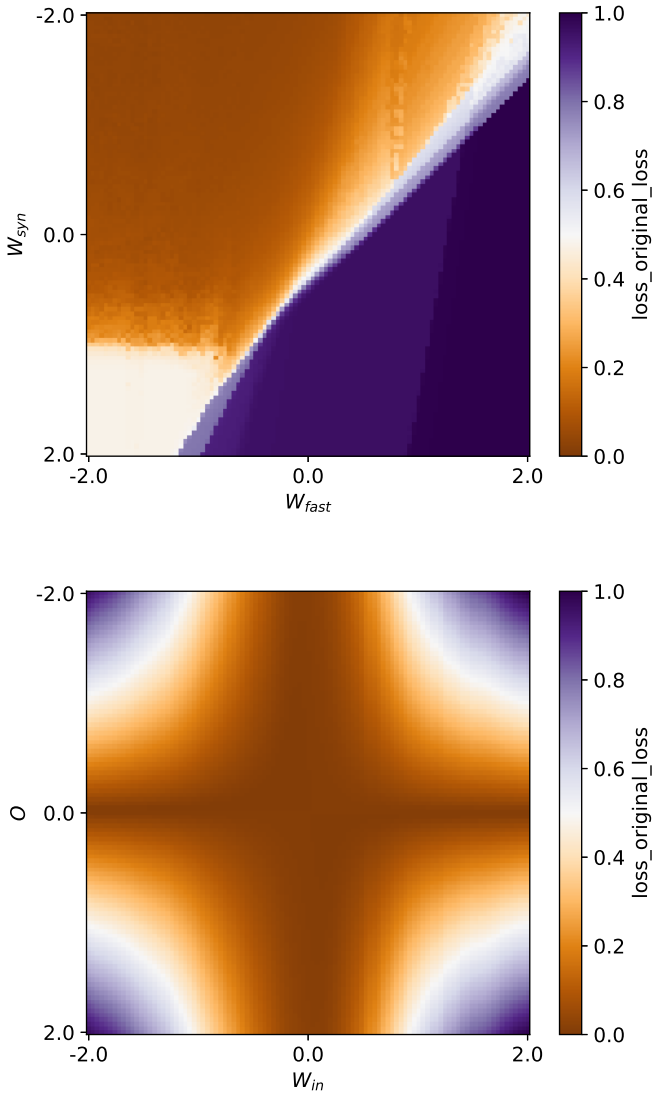

Figure 7.5: The parameter error landscape for NLIF SNNs over the auto-encoding task as the target signal.

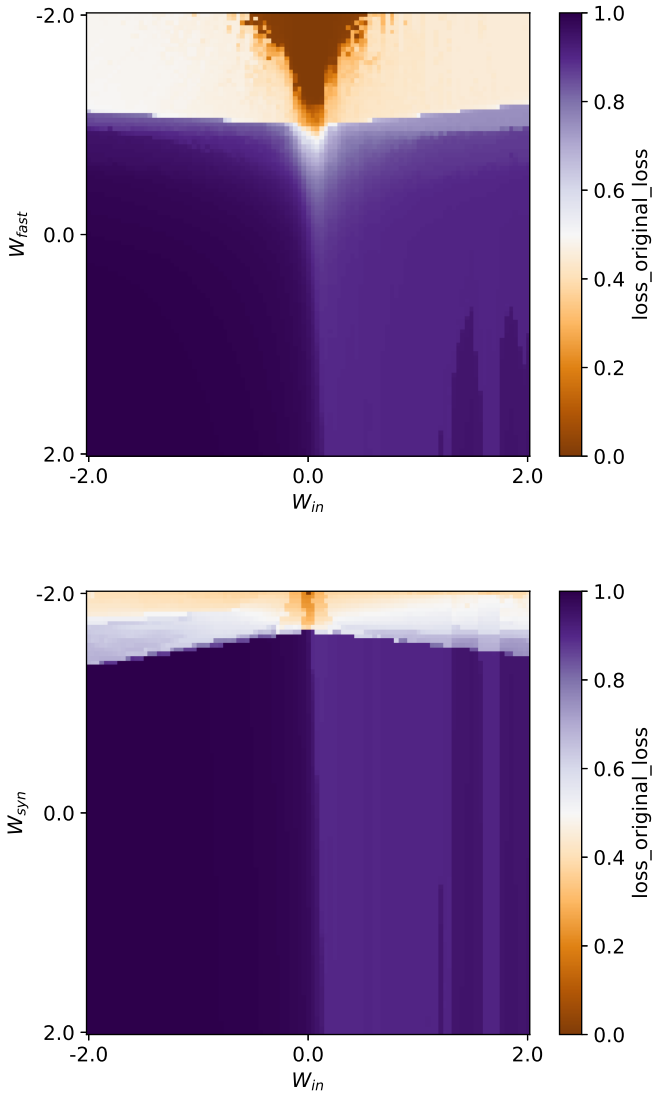

Figure 7.6: The parameter error landscape for NLIF SNNs over the auto-encoding task as the target signal.

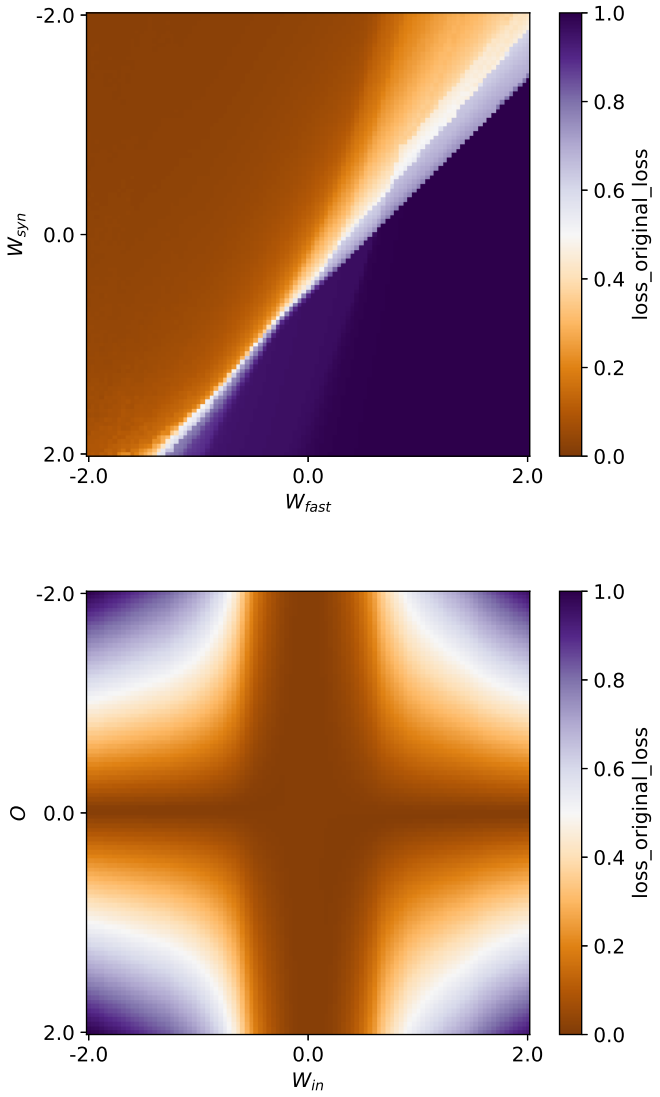

Figure 7.7: The parameter error landscape for LIF SNNs over the auto-encoding task as the target signal.

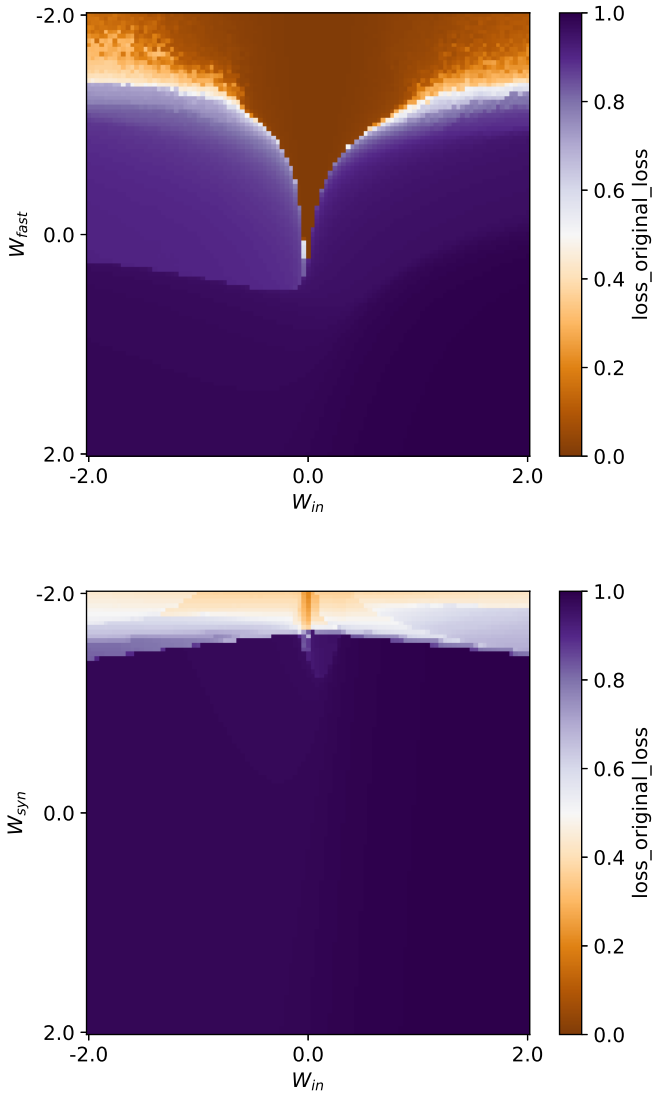

Figure 7.8: The parameter error landscape for LIF SNNs over the auto-encoding task as the target signal.

Table 7.1: Average RMSE per experiment and model type with the subthreshold synapse current model, where ”AE” denotes auto-encoding, and ”GPE” denotes general predictive encoding.

		$RMSE$	std
AE	NLIF	**0.175**	0.063
AE	LIF	0.201	0.098
GPE	NLIF	0.080	0.027
GPE	LIF	**0.058**	0.011

7.4 Discussion

By introducing a synapse model that varies continuously with the membrane potential when inside of an active zone, and defining the signal such that it generates a spike current summing to one in all cases of entering the active zone, we replicate the findings of the original paper [53] for two distinct tasks. The first is an auto-encoding task, in which the output signal is simply an encoding of the input-signal, and the second is a task in which the output signal is a linear combination of the input signal. In both tasks we find that the the setup enables GBO with excellent task performance and convergence rates. However, the model parameters that are inferred vary greatly for the same tasks with the different random seeds and thus training signals and model initialisations. Yet, the different inferred parameter sets perform as well - exemplifying that there are a myriad of possible solutions in the solution space, as previously discussed and also here illuminated in the parameter landscape (defined by the error metric) plots. Interestingly, our results show that the approach works well also for leaky SNN models, demonstrating that aspects relating to the setup enable successful GBO with regards to the task, with the model accurately reproducing the target signal, and also suggesting that further research incorporating the synapse model would be interesting to pursue. Further experiments could be performed to try and illuminate whether these dynamics could in fact be crucial in vivo - but nevertheless, observing such an effect in silico is an interesting finding in itself, which may be exploited in designing algorithms such as for low-power on chip spiking network models.

The fact that we observe sudden jumps in the loss for some epochs during training in this chapter may indicate that we're able to traverse the error landscape better than in the setups of chapter 5, including traversing such peaks which may otherwise hinder inference, as this was not something we observed when using the spike trains as target signals in the previous chapters.

A common argument for using a more fine-grained spiking model than a rate-based model is that in limiting the model to the rate only we do not describe the fast dynamics of spike-based computation. However, it is often problematic to base inference on a precise timing based model and signal, particularly when the signal slightly noisy and/or stochastic, as seen particularly in the synthetic data model inference of this work. Interestingly, however using the lower-dimensional signals in this chapter is entirely compatible with the fast spike-based dynamics, and enables GBO for inference of stable configurations that solve the tasks well. This is an interesting observation that

to the best of our knowledge has not been well described or observed in the literature before for LIF models. As such, it might provide a basis for bridging rate-based and precise timing-based models and approaches. In either case, the approach is a way of enabling optimisation for the usually difficult to optimise discrete, binary all-or-none spike signals of SNNs, by instead considering the subthreshold synaptic currents as the signals - which might be argued to be even more biologically realistic, as the cellular machinery has significantly pronounced dynamics and effects other than upon the precise time of spiking. Further, instead of circumventing the non-differentiability of binary spike-signals by constructing a surrogate signal, we operate on one of the natural signals of the model system. As a last note on the model signal, we would like to include a point made in the original paper; namely that the gradient calculation procedure, involving pre- and post-synaptic multiplication (and by task design), in fact is fairly analogous to spike-time dependent plasticity, due to the active-zone synapse current model.

Chapter 8

In sum: SNN GBO requires a well-defined setting

Optimisation enables in-place inference, but convergence requires a well-defined setting, such as a well-defined input-output signal. If dealing with spike train data directly, by correlation and number of mutations of the ordered spike train set, we could potentially get closer to a more well-defined loss metric than when using a rate-based metric as in this work - but such a metric would not be differentiable, nor would even such a metric be able to precisely measure whether one network is functionally equivalent to another. Part of the difficulty stems from that (1) there are multiple configurations which can produce similar spike patterns, and thus we may hypothesise that there is no ”ground-truth” optimal solution per se, and (2) due to the multiple factors of stochasticity we cannot operate with precise spike train comparisons. A better starting point for SNN inference might be to fit the models to more low-dimensional, well-defined target signals, preferably with an informative input, as demonstrated to converge well for both non-leaky as well as leaky integrate-and-fire SNNs in chapter 7. With a composite linear transformation, convergence is significantly improved and near perfect with this lower-dimensional signal, and task performance also consistently good with a low error.

An issue in all contexts defined throughout this SNN inference work might be said to be that the same (ground-truth) parameters are not retrieved when comparing the inferred models with the target ground-truth generative models, as illuminated by the parameter landscape plots in the previous chapter. However, there are multiple trajectories that may give rise to elegant solutions biologically speaking of e.g. motor tasks in the motor cortex [72]. Along this line of thought, it may not matter what specific

configuration a network reaches, as long as it reaches "the bottom of the loss curve", i.e. finds a good/satisfactory solution. Biologically speaking, any configuration solving the task at hand may be satisfactory, and the definition of optimal is plural, and diverse. In terms of optimisation; optimal can be said to be a set of frontiers in a high-dimensional space. In Bayesian inference approaches we may be able to trace this front by sampling from the full posterior, which may form realistic combinations in the output-space [70]. However, the fact that potentially non-overlapping regions may form good solutions also explain why our posteriors as inferred in chapter 5 were not necessarily centred around the true means, even for the less complex experiments, where the complexity should not give rise to divergence due to a too low sample size. Further, when ABC/SBI approaches become intractable, we have demonstrated that GBO may yet successfully retrieve model configurations that model the data or perform the task at hand.

Future work

The question yet remains how we may robustly solve automatic model inference of SNN models in a setting more suited for in vivo recordings and data, such as for spike train data, and in doing so potentially accelerating computational neuroscience and modeling research. This work contributes to this goal by demonstrating a way of combining gradient based optimisation and spiking neural network model inference by considering a range of different models, loss metrics, model types, data types, and synapse models. In doing so, we demonstrate both the potential as well as limitations that this approach bears in its studied form. Below is a list of points summarising some key observations that we made during this research, which may be useful to bear in mind for the future researcher in the field:

- Work with as well-defined input-output transformations as possible
- Consider using lower-dimensional input-output signals to greatly facilitate GBO
- Consider simulating multi-layer networks - although these are then non-linear and less straightforward, they could constrain the state space
- Constrain parameter intervals as much as possible using sensible intervals (such as when looking at probable cell type mixtures using the Allen Brain DB)

- Using a subthreshold synapse model results in more granular continuously differentiable signals, allowing for optimisation for instance in ”silent” models with subthreshold synaptic currents, and improves optimisation

Lastly, another strand of work that we think would would be interesting to pursue in future work is extending work on FORCE-learning both for learning different SNN parameters, and in different task domains, such as for more heterogeneous target spike trains, including spike trains recorded across modalities and brain states. As a promising method applied to Izhikevich SNNs for data of a rather cyclic nature, if successful for more complex data, such as data from across states and modalities, the Izhikevich model could lend itself well to biological interpretation, as is the case for GLIF SNNs.

Bibliography

[1] Allen Institute for Brain Science. Allen Cell Types Database, Technical White Paper: GLIF Models, 2017.

[2] Ron C. Anafi, Matthew S. Kayser, and David M. Raizen. Exploring phylogeny to find the function of sleep. *Nature Reviews Neuroscience*, 20(2):109–116, 2 2019.

[3] Guillaume Bellec, Franz Scherr, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. Biologically inspired alternatives to backpropagation through time for learning in recurrent neural nets. Technical report, Institute for Theoretical Computer Science, Graz University of Technology, Austria, 2019.

[4] Safa Ben Atitallah, Maha Driss, Wadii Boulila, and Henda Ben Ghézala. Randomly initialized convolutional neural network for the recognition of COVID-19 using X-ray images. *International Journal of Imaging Systems and Technology*, 32(1):55–73, 1 2022.

[5] Yoshua Bengio, Nicolas Boulanger-Lewandowski, and Razvan Pascanu. Advances in optimizing recurrent networks. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pages 8624–8628, 2013.

[6] Yoshua Bengio and Olivier Delalleau. On the expressive power of deep architectures. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6925 LNAI:18–36, 2011.

[7] James Bergstra, Olivier Breuleux, Frederic Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley,

and Yoshua Bengio. Theano: a CPU and GPU math compiler in Python. *Proceedings of the Python for Scientific Computing Conference (SciPy)*, (Scipy):1–7, 2010.

[8] S M Bohte, H La Poutré, and J N Kok. Error-Backpropagation in Temporally Encoded Networks of Spiking Neurons. *Neurocomputing*, 48:17–37, 2002.

[9] ALEXANDER A. BORBÉLY and PETER ACHERMANN. Concepts and models of sleep regulation: an overview. *Journal of Sleep Research*, 1(2):63–79, 6 1992.

[10] Romain Brette. Philosophy of the spike: Rate-based vs. Spike-based theories of the brain. *Frontiers in Systems Neuroscience*, 9(November):151, 11 2015.

[11] Romain Brette. Is coding a relevant metaphor for the brain?, 2019.

[12] Ritchie E. Brown, Radhika Basheer, James T. McKenna, Robert E. Strecker, and Robert W. McCarley. Control of Sleep and Wakefulness. *Physiological Reviews*, 92(3):1087–1187, 7 2012.

[13] A. N. Burkitt. A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input. *Biological Cybernetics*, 95(1):1–19, 7 2006.

[14] György Buzsáki. Hippocampal sharp wave-ripple: A cognitive biomarker for episodic memory and planning. *Hippocampus*, 25(10):1073–1188, 2015.

[15] György C N Buzsáki. *Rhythms of the brain*. Oxford University Press, New York, 2006.

[16] Clifton W. Callaway, Ralph Lydic, Helen A. Baghdoyan, and J. Allan Hobson. Pontogeniculooccipital waves: spontaneous visual system activity during rapid eye movement sleep. *Cellular and Molecular Neurobiology*, 7(2):105–149, 1987.

[17] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Gated Feedback Recurrent Neural Networks. 2015.

[18] Michael Schellenberger Costa, Jan Born, Jens Christian Claussen, and Thomas Martinetz. Modeling the effect of sleep regulation on a neural mass model. *Journal of Computational Neuroscience*, 41(1):15–28, 2016.

[19] Julia Cox, Lucas Pinto, and Yang Dan. Calcium imaging of sleep–wake related neuronal activity in the dorsal pons. *Nature Communications*, 7(1):10763, 12 2016.

[20] Benjamin Cramer, Yannik Stradmann, Johannes Schemmel, and Friedemann Zenke. The Heidelberg spiking datasets for the systematic evaluation of spiking neural networks. Technical report, 2019.

[21] Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences of the United States of America*, 117(48):30055–30062, 12 2020.

[22] Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences of the United States of America*, 117(48):30055–30062, 12 2020.

[23] Subimal Datta. Cellular basis of pontine ponto-geniculo occipital wave generation and modulation. *Cellular and Molecular Neurobiology*, 17(3):341–365, 1997.

[24] Peter Dayan and L.F. Abbot. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*, volume 241. The MIT Press, London, England, 2001.

[25] Luisa De Vivo, Michele Bellesi, William Marshall, Eric A. Bushong, Mark H. Ellisman, Giulio Tononi, and Chiara Cirelli. Ultrastructural evidence for synaptic scaling across the wake/sleep cycle. *Science*, 355(6324):507–510, 2 2017.

[26] Daniel Deitch, Alon Rubin, and Yaniv Ziv. Representational drift in the mouse visual cortex. *Current Biology*, 31(19):4327–4339, 10 2021.

[27] C. G. Diniz Behn and V. Booth. Simulating Microinjection Experiments in a Novel Model of the Rat Sleep-Wake Regulatory Network. *Journal of Neurophysiology*, 103(4):1937–1953, 2010.

[28] J R Dunmyre, G A Mashour, and V Booth. Coupled Flip-Flop Model for REM Sleep Regulation in the Rat. *PLoS ONE*, 9(4):94481, 2014.

[29] Conor Durkan, George Papamakarios, and Iain Murray. Sequential Neural Methods for Likelihood-free Inference. 11 2018.

[30] Ada Eban-Rothschild, Lior Appelbaum, and Luis De Lecea. Neuronal Mechanisms for Sleep/Wake Regulation and Modulatory Drive. *Neuropsychopharmacology*, 43:937–952, 2018.

[31] Francisco Fernández. Polynomial Approximations. *Introduction to Perturbation Theory in Quantum Mechanics*, 32(Nips 2018):137–172, 2000.

[32] Michelle Fleshner, Victoria Booth, Daniel B. Forger, and Cecilia G.Diniz Behn. Circadian regulation of sleep-wake behaviour in nocturnal rats requires multiple signals from suprachiasmatic nucleus. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 369(1952):3855–3883, 2011.

[33] Jimmy J. Fraigne, Zoltan A. Torontali, Matthew B. Snow, and John H. Peever. REM sleep at its core - Circuits, neurotransmitters, and pathophysiology. *Frontiers in Neurology*, 6(MAY):123, 2015.

[34] Pedro J. Gonçalves, Jan Matthis Lueckmann, Michael Deistler, Marcel Nonnenmacher, Kaan Öcal, Giacomo Bassetto, Chaitanya Chintaluri, William F. Podlaski, Sara A. Haddad, Tim P. Vogels, David S. Greenberg, and Jakob H. Macke. Training deep neural density estimators to identify mechanistic models of neural dynamics. *bioRxiv*, pages 1–45, 2019.

[35] Oscar C Gonzalez, Yury Sokolov, Giri Krishnan, and Maxim Bazhenov. Can sleep protect memories from catastrophic forgetting? *bioRxiv*, page 569038, 2019.

[36] Dan Goodman. Brian: a simulator for spiking neural networks in Python. *Frontiers in Neuroinformatics*, 2(November):1–10, 2008.

[37] K. P. Grace, L. E. Vanstone, and R. L. Horner. Endogenous Cholinergic Input to the Pontine REM Sleep Generator Is Not Required for REM Sleep to Occur. *Journal of Neuroscience*, 34(43):14198–14209, 10 2014.

[38] David S Greenberg, Marcel Nonnenmacher, and Jakob H Macke. Automatic Posterior Transformation for Likelihood-free Inference. Technical report.

[39] David S Greenberg, Marcel Nonnenmacher, and Jakob H Macke. Automatic Posterior Transformation for Likelihood-free Inference. Technical report, 5 2019.

[40] Stefan Grün, Sonja; Rotter. *Analysis of Parallel Spike Trains*. Springer, 2010.

[41] Milan Halgren and Michael M Halassa. The Many Roads to Sleep. *Neuron*, 103(2):181–183, 7 2019.

[42] Charlotte Héricé, Amisha A Patel, and Shuzo Sakata. Circuit mechanisms and computational models of REM sleep. *Neuroscience Research*, pages 1–16, 2018.

[43] Charlotte Héricé, Amisha A. Patel, and Shuzo Sakata. Circuit mechanisms and computational models of REM sleep, 3 2019.

[44] Charlotte Héricé and Shuzo Sakata. Pathway-dependent regulation of sleep dynamics in a network model of the sleep-wake cycle. *bioRxiv*, page 705822, 7 2019.

[45] J. Allan Hobson and Edward F. Pace-Schott. The cognitive neuroscience of sleep: neuronal systems, consciousness and learning. *Nature Reviews Neuroscience*, 3(9):679–693, 9 2002.

[46] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[47] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500–544, 8 1952.

[48] Erik Hoel. The overfitted brain: Dreams evolved to assist generalization. *Patterns*, 2(5):100244, 2021.

[49] J J; Hopfield, D I; Feinstein, and R G Palmer. 'Unlearning' has a stabilizing effect in collective memories. *Letters to Nature*, 304:158–159, 1983.

[50] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1 1989.

[51] Conor Houghton and Kamal Sen. A new multineuron spike train metric. *Neural Computation*, 20(6):1495–1511, 2008.

[52] Dongsung Huh and Terrence J Sejnowski. Gradient Descent for Spiking Neural Networks. Technical report, Salk Institute, 2017.

[53] Dongsung Huh and Terrence J. Sejnowski. Gradient descent for spiking neural networks. *Advances in Neural Information Processing Systems*, 2018-Decem(NeurIPS):1433–1443, 2018.

[54] Eugene M. Izhikevich. Which Model to Use for Cortical Spiking Neurons. *IEEE Transactions on Neural Networks*, 15(5):1063–1070, 9 2004.

[55] Eugene M. Izhikevich. Polychronization: Computation with Spikes. *Neural Computation*, 18(2):245–282, 2 2006.

[56] Hyeryung Jang, Osvaldo Simeone, Brian Gardner, and André Grüning. An Introduction to Probabilistic Spiking Neural Networks. *arXiv*, 2019.

[57] Yingyezhe Jin, Wenrui Zhang, and Peng Li. Hybrid macro/micro level backpropagation for training deep spiking neural networks. *Advances in Neural Information Processing Systems*, 2018-Decem(1):7005–7015, 2018.

[58] Melissa G. Johnson and Sylvain Chartier. Spike neural models (part I): The Hodgkin-Huxley model. *The Quantitative Methods for Psychology*, 13(2):105–119, 5 2017.

[59] Alex C Keene and Erik R Duboue. The origins and evolution of sleep. *The Company of Biologists Ltd*, 2018.

[60] Mudasir Ahmad Khanday, Bindu I Somarajan, Rachna Mehta, and Birendra Nath Mallick. Noradrenaline from Locus Coeruleus Neurons Acts on Pedunculo-Pontine Neurons to Prevent REM Sleep and Induces Its Loss-Associated Effects in Rats. *eNeuro*, 3(6):0108–16, 11 2016.

[61] Diederik P. Kingma and Jimmy Lei Ba. Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations*, pages 1–13, 2015.

[62] Osame Kinouchi. Dreams , Endocannabinoids and Itinerant Dynamics in Neural Networks : elaborating Crick-Mitchison ' s Unlearning Hypothesis. *arXiv*, 2002.

[63] Jens G. Klinzing, Niels Niethard, and Jan Born. Mechanisms of systems memory consolidation during sleep, 8 2019.

[64] Thomas Kreuz, Daniel Chicharro, Conor Houghton, Ralph G. Andrzejak, and Florian Mormann. Monitoring spike train synchrony. *Journal of Neurophysiology*, 109(5):1457–1472, 3 2013.

[65] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[66] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10(NOV):508, 2016.

[67] Andrew S. Lim, Andres M. Lozano, Elena Moro, Clement Hamani, William D. Hutchison, Jonathan O. Dostrovsky, Anthony E. Lang, Richard A. Wennberg, and Brian J. Murray. Characterization of REM-Sleep Associated Ponto-Geniculo-Occipital Waves in the Human Pons. *Sleep*, 30(7):823–827, 7 2007.

[68] Jan-Matthis Lueckmann, Giacomo Bassetto, Theofanis Karaletsos, and Jakob H Macke. Likelihood-free inference with emulator networks. Technical report, Max Planck Institute for Intelligent Systems, the University of Tübingen, 1 2018.

[69] Jan-Matthis Lueckmann, Jan Boelts, David S Greenberg, Pedro J Gonçalves, and Jakob H Macke. Benchmarking Simulation-Based Inference. Technical report, Max Planck Institute for Intelligent Systems, the University of Tübingen, 2021.

[70] Jan Matthis Lueckmann, Pedro J. Gonçalves, Giacomo Bassetto, Kaan Öcal, Marcel Nonnenmacher, and Jakob H. Mackey. Flexible statistical inference for mechanistic models of neural dynamics. *Advances in Neural Information Processing Systems*, 2017-Decem(Nips):1290–1300, 2017.

[71] B. N. Mallick, S. Kaur, and R. N. Saxena. Interactions between cholinergic and GABAergic neurotransmitters in and around the locus coeruleus for the induction and maintenance of rapid eye movement sleep in rats. *Neuroscience*, 104(2):467–485, 2001.

[72] Adam H. Marblestone, Greg Wayne, and Konrad P. Kording. Toward an integration of deep learning and neuroscience. *Frontiers in Computational Neuroscience*, 10(SEP):1–41, 2016.

[73] Cristina Martinez-Gonzalez, J. Paul Bolam, and Juan Mena-Segovia. Topographical Organization of the Pedunculopontine Nucleus. *Frontiers in Neuroanatomy*, 5:22, 4 2011.

[74] W S McCulloch and W Pitts. A Logical Calculus of the Idea Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[75] Hesham Mostafa. Supervised learning based on temporal coding in spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 29(7):3227–3235, 2018.

[76] Hesham Mostafa, Bodo Rückauer, Eth Zürich, Switzerland Eric Hunsberger, David Kappel, Chankyu Lee, Syed Shakib Sarwar, Priyadarshini Panda, Gopalakrishnan Srinivasan, and Kaushik Roy. Enabling Spike-Based Backpropagation for Training Deep Neural Network Architectures. *Frontiers in Neuroscience — www.frontiersin.org*, 14:119, 2020.

[77] Santosh C. Narwade, Birendra N. Mallick, and Deepti D. Deobagkar. Transcriptome Analysis Reveals Altered Expression of Memory and Neurotransmission Associated Genes in the REM Sleep Deprived Rat Brain. *Frontiers in Molecular Neuroscience*, 10(March):1–13, 2017.

[78] Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate Gradient Learning in Spiking Neural Networks: Bringing the Power of Gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

[79] J.A. Nelder and R.W.M. Wedderburn. Composite Link Functions in Generalized Linear Models Author (s): R . Thompson and R . J . Baker Published by : Wiley for the Royal Statistical Society Stable URL : http://www.jstor.org/stable/2346381 Composite Link Functions in Generalized Linear Model. *Journal of the Royal Statistical Society*, 135(3):370–384, 1972.

[80] Kun-Ming Ni, Xiao-Jun Hou, Ci-Hang Yang, Ping Dong, Yue Li, Ying Zhang, Ping Jiang, Darwin K Berg, Shumin Duan, and Xiao-Ming Li. Selectively driv-

ing cholinergic fibers optically in the thalamic reticular nucleus promotes sleep. *eLife*, 2016.

[81] Wilten Nicola and Claudia Clopath. Supervised Learning in Spiking Neural Networks with FORCE Training. *Nature Communications*, 8(1):2208, 9 2016.

[82] Wilten Nicola and Claudia Clopath. Supervised learning in spiking neural networks with FORCE training. *Nature Communications*, 8(1):1–15, 12 2017.

[83] Lucas D.R. Oliveira, Rogerio M. Gomes, Bruno A. Santos, and Henrique E. Borges. Effects of the parameters on the oscillation frequency of Izhikevich spiking neural networks. *Neurocomputing*, 2 2019.

[84] Arno Onken, Jian K. Liu, P. P.Chamanthi R. Karunasekara, Ioannis Delis, Tim Gollisch, and Stefano Panzeri. Using Matrix and Tensor Factorizations for the Single-Trial Analysis of Population Spike Trains. *PLoS Computational Biology*, 12(11):1–46, 2016.

[85] Dinesh Pal, Vibha Madan, and Birendra Nath Mallick. Neural mechanism of rapid eye movement sleep generation: Cessation of locus coeruleus neurons is a necessity. *Sheng li xue bao : [Acta physiologica Sinica]*, 57(June):401–413, 2005.

[86] Dinesh Pal and Birendra Nath Mallick. Neural mechanism of rapid eye movement sleep generation with reference to REM-OFF neurons in locus coeruleus. *The Indian journal of medical research*, 125(6):721–739, 6 2007.

[87] Dagmara Panas, Hayder Amin, Alessandro Maccione, Oliver Muthmann, Mark van Rossum, Luca Berdondini, and Matthias H. Hennig. Sloppiness in Spontaneously Active Neuronal Networks. *Journal of Neuroscience*, 35(22):8480–8492, 6 2015.

[88] Liam Paninski, Jonathan W. Pillow, and Eero P. Simoncelli. Maximum likelihood estimation of a stochastic integrate-and-fire neural encoding model. *Neural Computation*, 16(12):2533–2561, 2004.

[89] Hugh Pastoll, Lukas Solanka, Mark C.W. van Rossum, and Matthew F. Nolan. Feedback Inhibition Enables Theta-Nested Gamma Oscillations and Grid Firing Fields. *Neuron*, 77(1):141–154, 1 2013.

[90] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary Devito Facebook, A I Research, Zeming Lin, Alban Desmaison, Luca Antiga, Orobix Srl, and Adam Lerer. Automatic differentiation in PyTorch. In *Advances in Neural Information Processing Systems*, pages 8024–8035. University of Warsaw, 10 2019.

[91] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary Devito Facebook, A I Research, Zeming Lin, Alban Desmaison, Luca Antiga, Orobix Srl, and Adam Lerer. Automatic differentiation in PyTorch. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035, 2019.

[92] Alexandre René, André Longtin, and Jakob H. Macke. Inference of a Mesoscopic Population Model from Population Spike Trains. *Neural Computation*, pages 1–51, 6 2020.

[93] Edmund T Rolls and Alessandro Treves. Introduction. In *Neural Networks and Brain Function*, chapter 1, page 418. Oxford University Press, Oxford, UK, 1998.

[94] Murray Rosenblatt. Remarks on Some Nonparametric Estimates of a Density Function. *The Annals of Mathematical Statistics*, 27(3):832–837, 9 1956.

[95] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, pages 318–362. The MIT Press, US, Cambridge MA 02142-1209, 1986.

[96] Thomas E. Scammell, Elda Arrigoni, and Jonathan O. Lipton. Neural Circuitry of Wakefulness and Sleep. *Neuron*, 93(4):747–765, 2 2017.

[97] J Schmidhuber. Deep Learning in Neural Networks: An Overview. *arXiv preprint arXiv:1404.7828*, 61:1–66, 2014.

[98] Tilo Schwalger, Moritz Deger, and Wulfram Gerstner. Towards a theory of cortical columns: From spiking neurons to interacting neural populations of finite size. *PLoS Computational Biology*, 13(4):e1005507, 4 2017.

[99] D Seung and L Lee. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems 13*, (1):556–562, 2001.

[100] H. Seung, Sebastian and Daniel D. Lee. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[101] Martino Sorbaro Sindaci. *Statistical modelling of neuronal population activity: from data analysis to network function*. PhD thesis, University of Edinburgh, 2018.

[102] Chen Song, Melanie Boly, Enzo Tagliazucchi, Helmut Laufs, and Giulio Tononi. BOLD signatures of sleep. *bioRxiv*, 1 2019.

[103] Carsen Stringer, Marius Pachitariu, Nicholas A. Steinmetz, Michael Okun, Peter Bartho, Kenneth D. Harris, Maneesh Sahani, and Nicholas A. Lesica. Inhibitory control of correlated intrinsic variability in cortical networks. *eLife*, 5(DECEMBER2016), 12 2016.

[104] Yanan Sun, Bing Xue, Mengjie Zhang, and Gary G. Yen. Evolving Deep Convolutional Neural Networks for Image Classification. *IEEE Transactions on Evolutionary Computation*, 24(2):394–407, 4 2020.

[105] Aboozar Taherkhani, Ammar Belatreche, Yuhua Li, Georgina Cosma, Liam P. Maguire, and T. M. McGinnity. A review of learning in biologically plausible spiking neural networks. *Neural Networks*, 122:253–272, 2 2020.

[106] Aboozar Taherkhani, Ammar Belatreche, Yuhua Li, and Liam P. Maguire. A Supervised Learning Algorithm for Learning Precise Timing of Multiple Spikes in Multilayer Spiking Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 29(11):5394–5407, 11 2018.

[107] Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony Maida. Deep learning in spiking neural networks. *Neural Networks*, 111:47–63, 3 2019.

[108] Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony Maida. Deep learning in spiking neural networks, 3 2019.

[109] Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony Maida. Deep learning in spiking neural networks, 3 2019.

[110] Tatjana Tchumatchenko, Theo Geisel, Maxim Volgushev, and Fred Wolf. Spike correlations - What can they tell about synchrony?, 2011.

[111] Corinne Teeter, Ramakrishnan Iyer, Vilas Menon, Nathan Gouwens, David Feng, Jim Berg, Aaron Szafer, Nicholas Cain, Hongkui Zeng, Michael Hawrylycz, Christof Koch, and Stefan Mihalas. Generalized leaky integrate-and-fire models classify multiple neuron types. *Nature Communications*, 9(1):1–15, 12 2018.

[112] Alvaro Tejero-Cantero, Jan Boelts, Michael Deistler, Jan-Matthis Lueckmann, Conor Durkan, Pedro Gonçalves, David Greenberg, and Jakob Macke. sbi: A toolkit for simulation-based inference. *Journal of Open Source Software*, 5(52):2505, 2020.

[113] Dominik Thalmeier, Marvin Uhlmann, Hilbert J. Kappen, and Raoul Martin Memmesheimer. Learning Universal Computations with Spikes. *PLoS Computational Biology*, 12(6):1004895, 6 2016.

[114] Panagiota Theodoni, Bernat Rovira, Yingxue Wang, and Alex Roxin. Theta-modulation drives the emergence of connectivity patterns underlying replay in a network model of place cells. *eLife*, 7, 10 2018.

[115] G. Thimm and E. Fiesler. Neural network initialization. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 930:535–542, 1995.

[116] Tomomi Tsunematsu, Amisha A Patel, Arno Onken, and Shuzo Sakata. State-dependent pontine ensemble dynamics and interactions with cortex across sleep states. *bioRxiv*, page 752683, 9 2019.

[117] Christa J. Van Dort, Daniel P. Zachs, Jonathan D. Kenny, Shu Zheng, Rebecca R. Goldblum, Noah A. Gelwan, Daniel M. Ramos, Michael A. Nolan, Karen Wang, Feng-Ju Weng, Yingxi Lin, Matthew A. Wilson, and Emery N. Brown. Optogenetic activation of cholinergic neurons in the PPT or LDT induces REM sleep. *Proceedings of the National Academy of Sciences*, 112(2):584–589, 1 2015.

[118] M. C.W. Van Rossum. A novel spike distance. *Neural Computation*, 13(4):751–763, 4 2001.

[119] M Walker. *Why we sleep: the new science of sleep and dreams*. Penguin Books, 2018.

[120] Christopher J. Watson, Ralph Lydic, and Helen A. Baghdoyan. Sleep duration varies as a function of glutamate and GABA in rat pontine reticular formation. *Journal of Neurochemistry*, 118(4):571–580, 2011.

[121] Franz Weber and Yang Dan. Circuit-based interrogation of sleep control. *Nature*, 538:51, 10 2016.

[122] Franz Weber, Johnny Phong Hoang Do, Shinjae Chung, Kevin T. Beier, Mike Bikov, Mohammad Saffari Doost, and Yang Dan. Regulation of REM and Non-REM Sleep by Periaqueductal GABAergic Neurons. *Nature Communications*, 9(1), 2018.

[123] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12(MAY):1–12, 2018.

[124] Friedemann; Zenke and Surya Ganguli. SuperSpike: Supervised Learning in Multilayer Spiking Neural Networks Friedemann. *Neural Computation*, 30:1514–1541, 2018.

Appendix A

Code

- For the subthreshold synapse current models, the code used may be found at github:gated_synapse_model.
- For a more general SNN inference demo framework, please see github:snn_inference_demo.
- For the NMF analysis, we extended custom Matlab code. The Python-code that we used in order to export model spike train data to a sparse, Matlab-compatible format may be found in the snn_inference_demo-repository, in the file data_util.py
- For the GLM MLE, we extended the custom Matlab code found at github:GLMspiketools.

Appendix B

Supplementary figures

Gated subthreshold synapse currents

Auto-encoding

NLIF

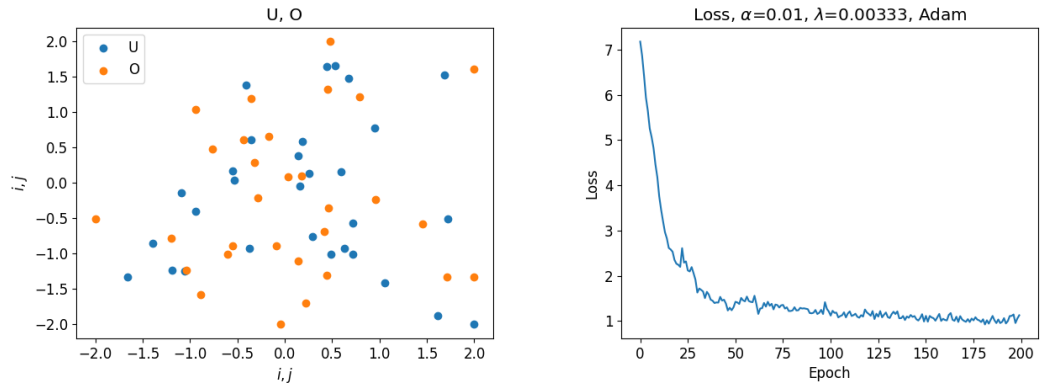

Figure B.1: Sample NLIF auto-encoding experiment; final weights and loss across epochs.

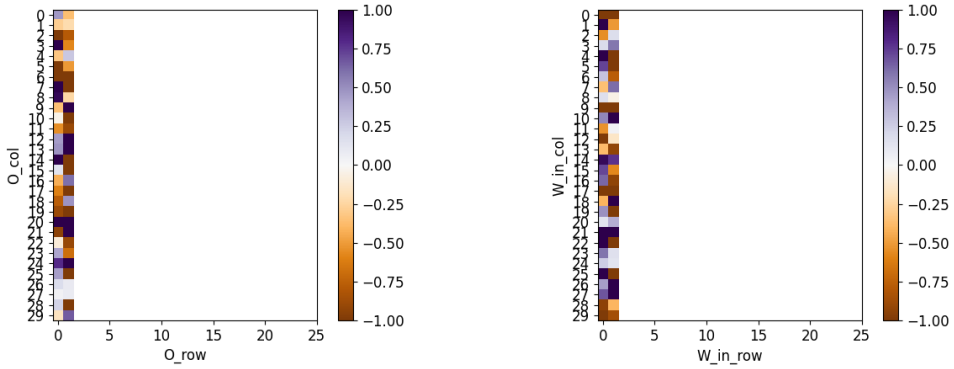

Figure B.2: Sample NLIF auto-encoding experiment; input and readout weights after training.

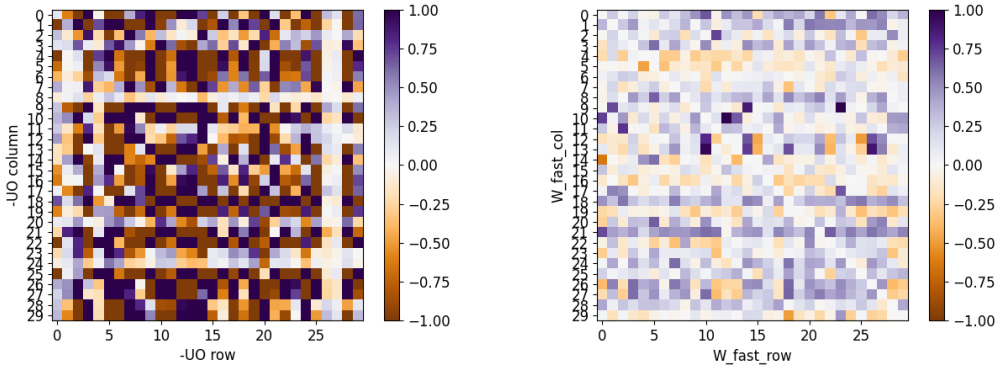

Figure B.3: Sample NLIF auto-encoding experiment; negative product of input and output weights, and the fast synaptic weights.

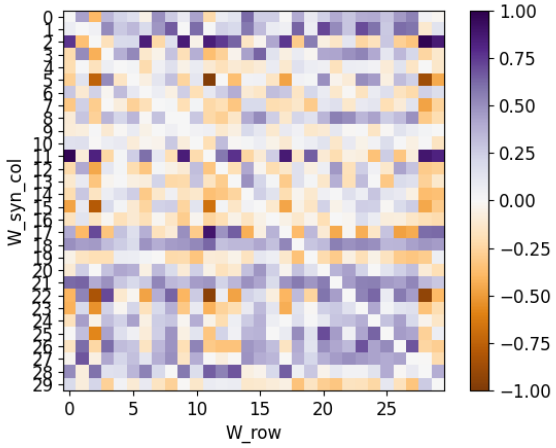

Figure B.4: Sample NLIF auto-encoding experiment; the synaptic weights.

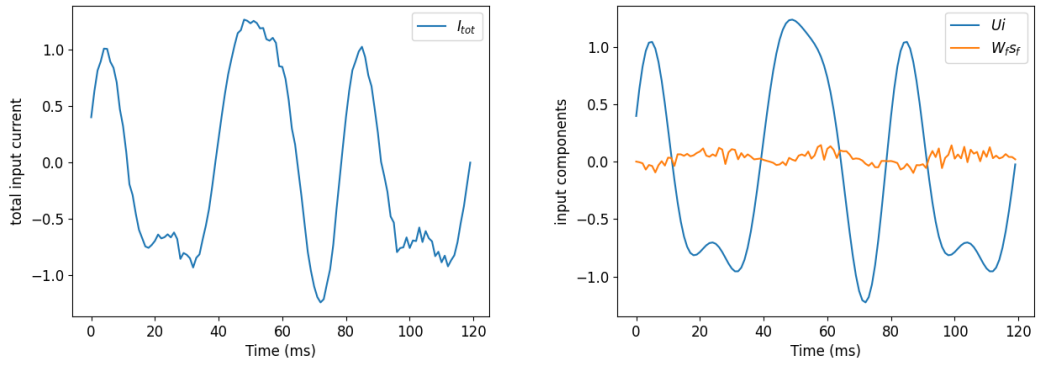

Figure B.5: Sample NLIF auto-encoding experiment; the input current and input components.

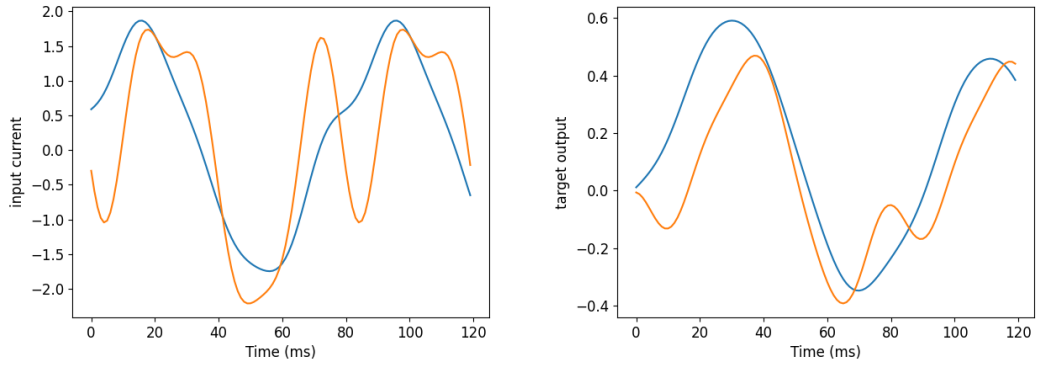

Figure B.6: Sample NLIF auto-encoding experiment; the inputs and targets.

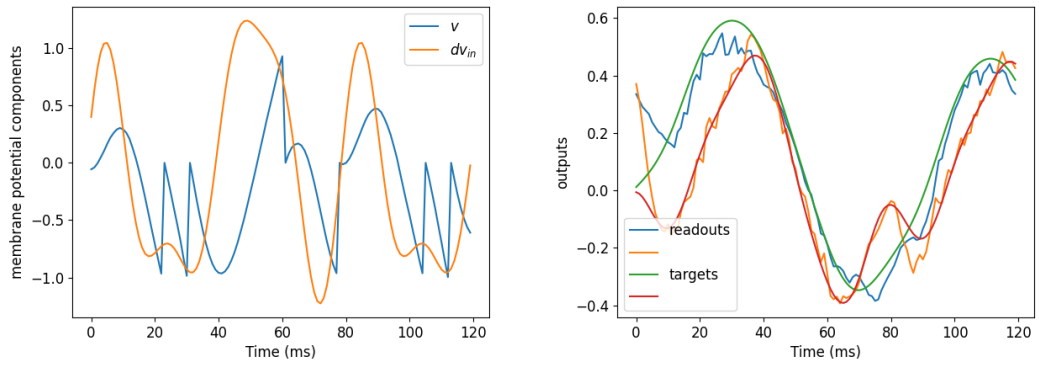

Figure B.7: Sample NLIF auto-encoding experiment; the membrane potentials and readouts.

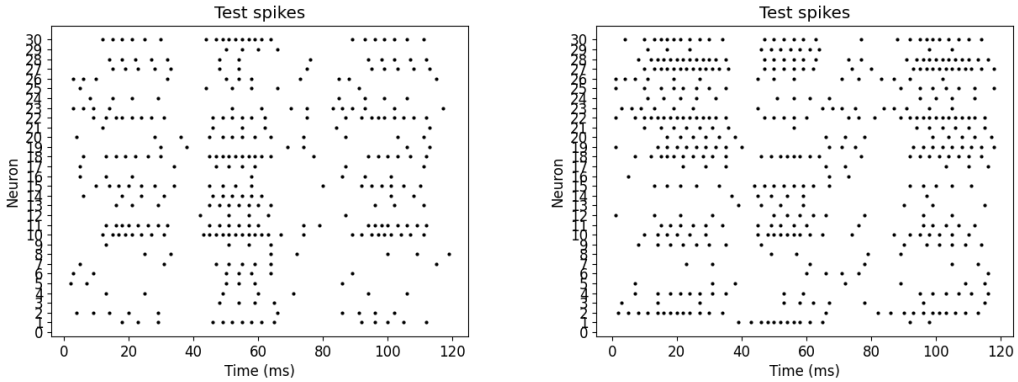

Figure B.8: Sample NLIF auto-encoding experiment; the model spike train after epoch $t_{iter} = 30$ (left), and after training, $t_{iter} = 200$ (right).

LIF

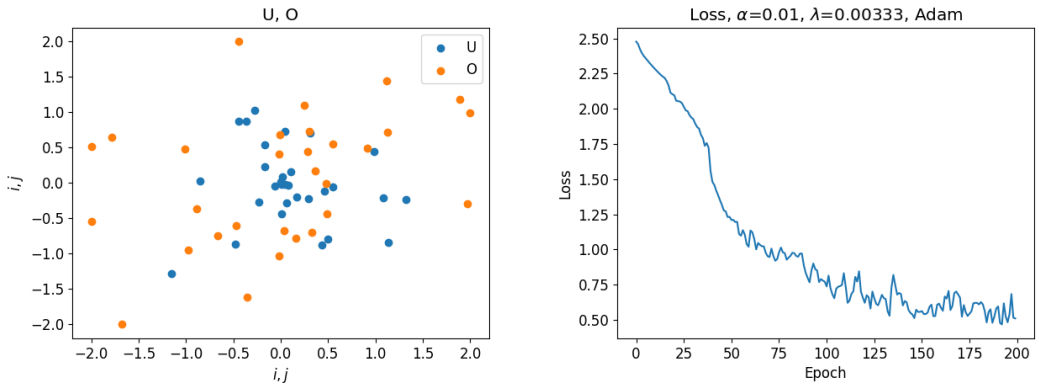

Figure B.9: Sample LIF auto-encoding experiment; final weights and loss across epochs.

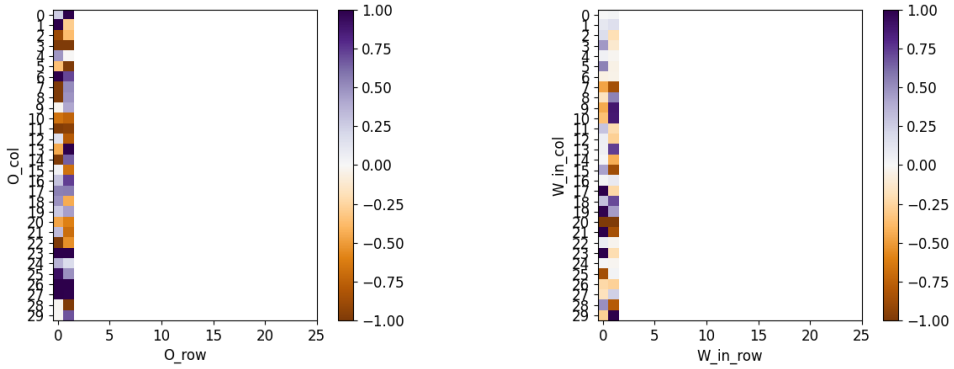

Figure B.10: Sample LIF auto-encoding experiment; input and readout weights after training.

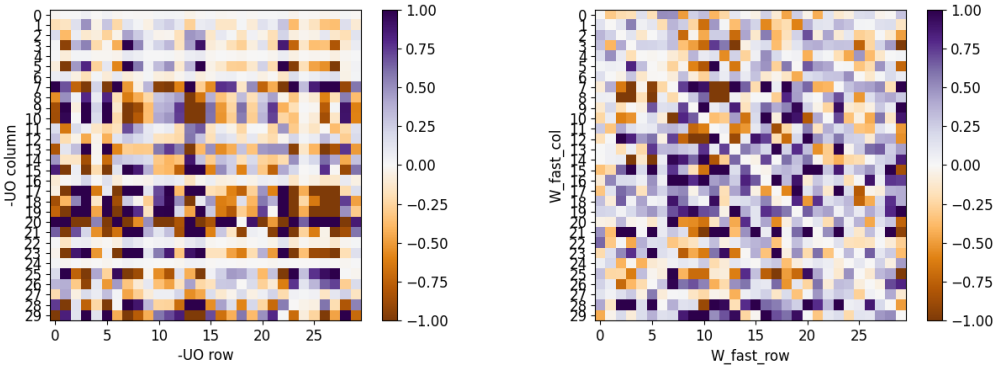

Figure B.11: Sample LIF auto-encoding experiment; negative product of input and output weights, and the fast synaptic weights.

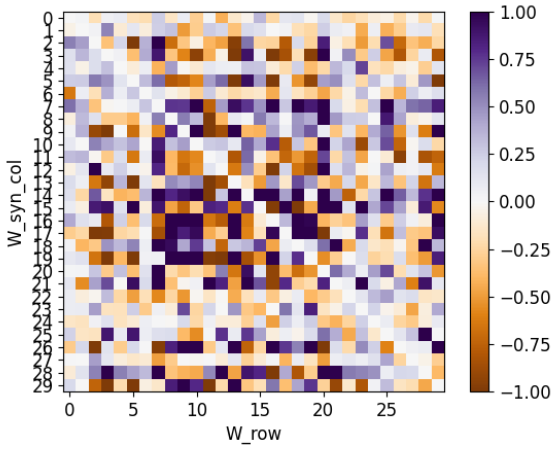

Figure B.12: Sample LIF auto-encoding experiment; the synaptic weights.

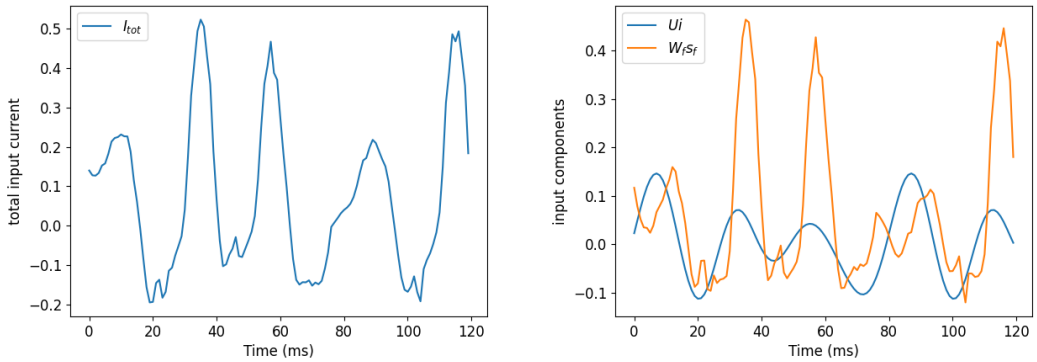

Figure B.13: Sample LIF auto-encoding experiment; the input current and input components.

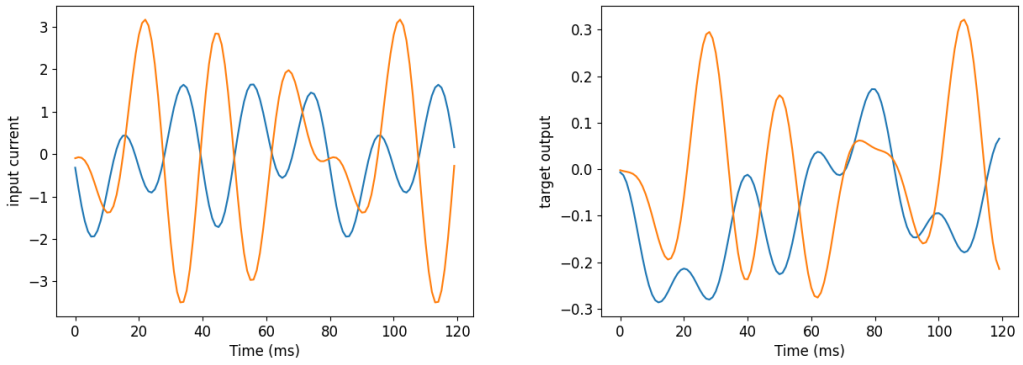

Figure B.14: Sample LIF auto-encoding experiment; the inputs and targets.

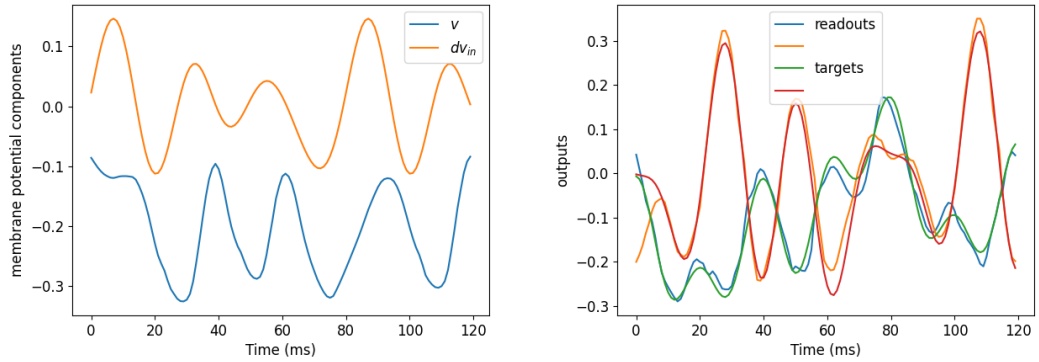

Figure B.15: Sample LIF auto-encoding experiment; the membrane potentials and readouts.

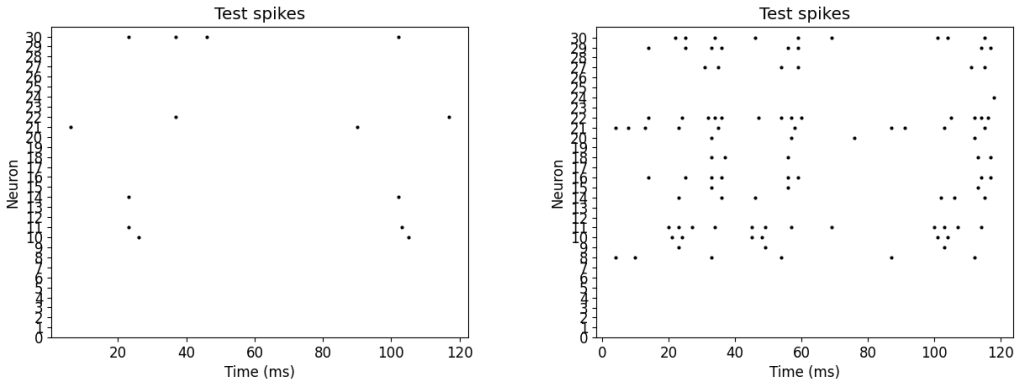

Figure B.16: Sample LIF auto-encoding experiment; the model spike train after epoch $t_{iter} = 30$ (left), and $t_{iter} = 80$ (right).

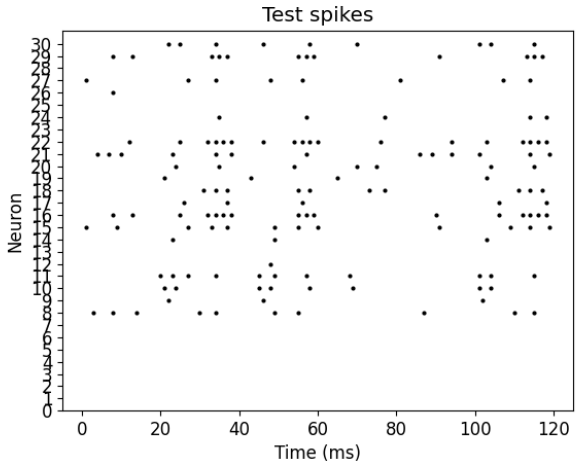

Figure B.17: Sample LIF auto-encoding experiment; the model readouts after training, epoch $t_{iter} = 200$.

General predictive encoding

NLIF

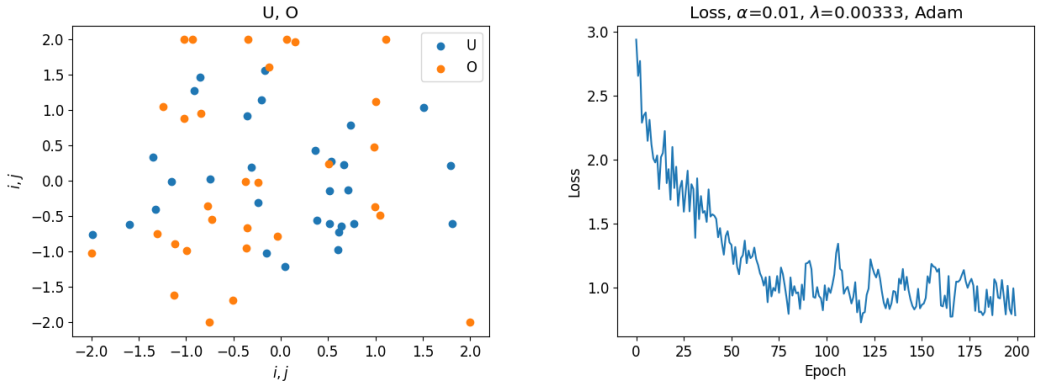

Figure B.18: Sample NLIF general predictive encoding experiment; final weights and loss across epochs.

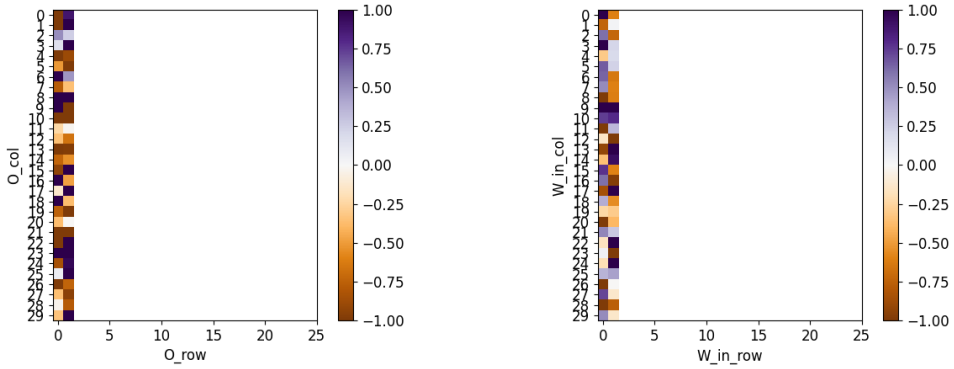

Figure B.19: Sample NLIF general predictive encoding experiment; input and readout weights after training.

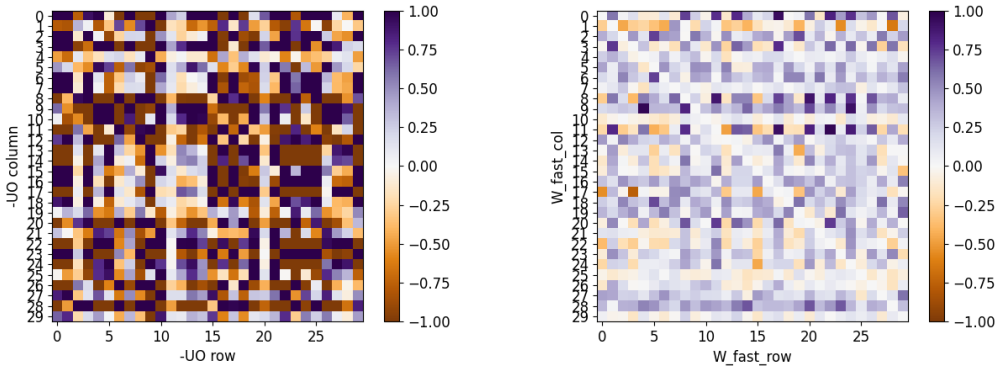

Figure B.20: Sample NLIF general predictive encoding experiment; negative product of input and output weights, and the fast synaptic weights.

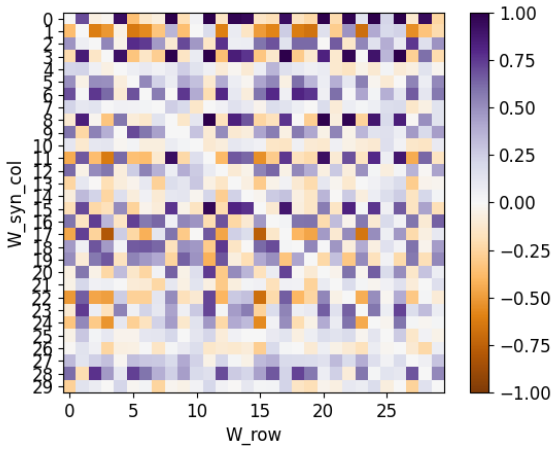

Figure B.21: Sample NLIF general predictive encoding experiment; the synaptic weights.

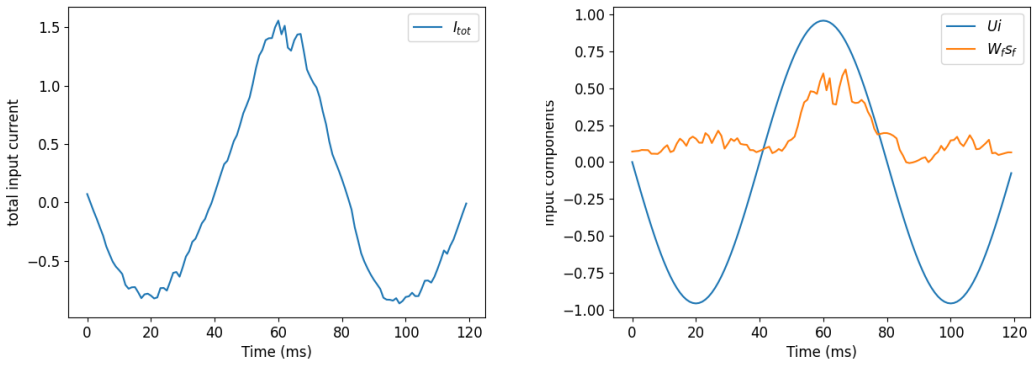

Figure B.22: Sample NLIF general predictive encoding experiment; the input current and input components.

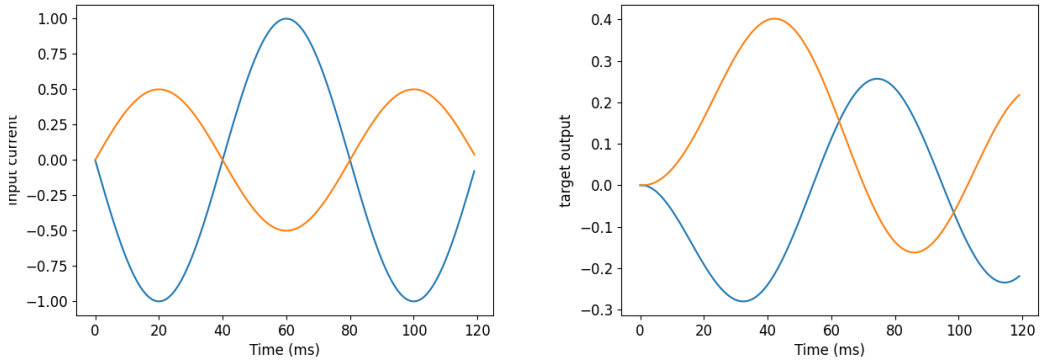

Figure B.23: Sample NLIF general predictive encoding experiment; the inputs and targets.

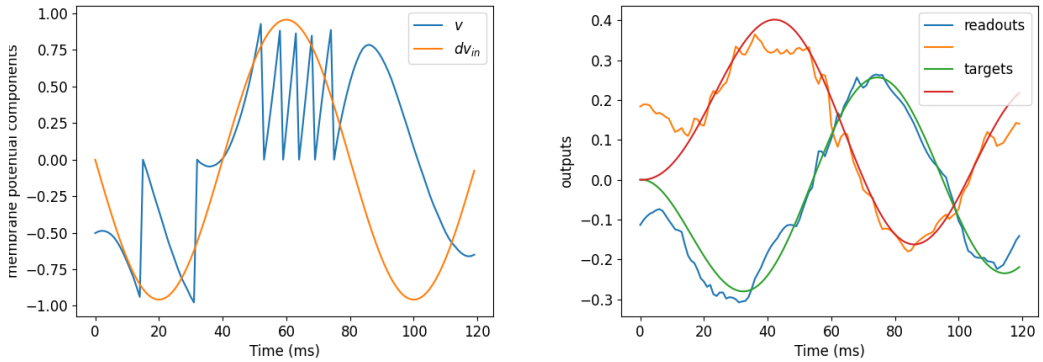

Figure B.24: Sample NLIF general predictive encoding experiment; the membrane potentials and readouts.

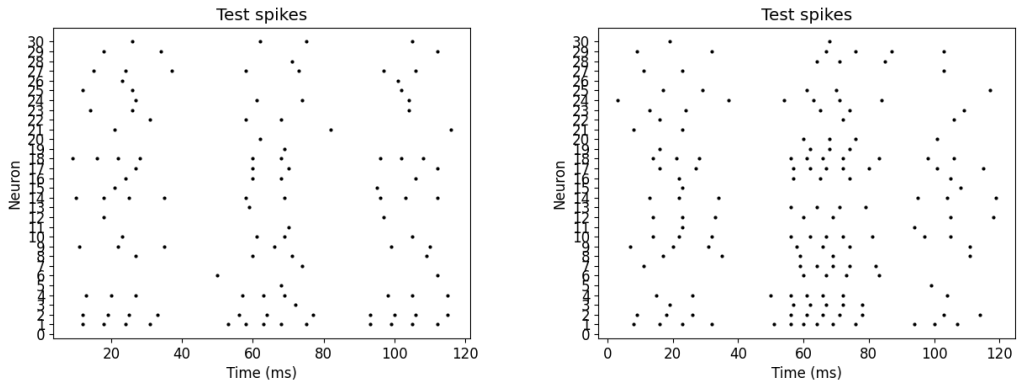

Figure B.25: Sample NLIF general predictive encoding experiment; the model spike train after epoch $t_{iter} = 0$ (left), and after training, $t_{iter} = 50$ (right).

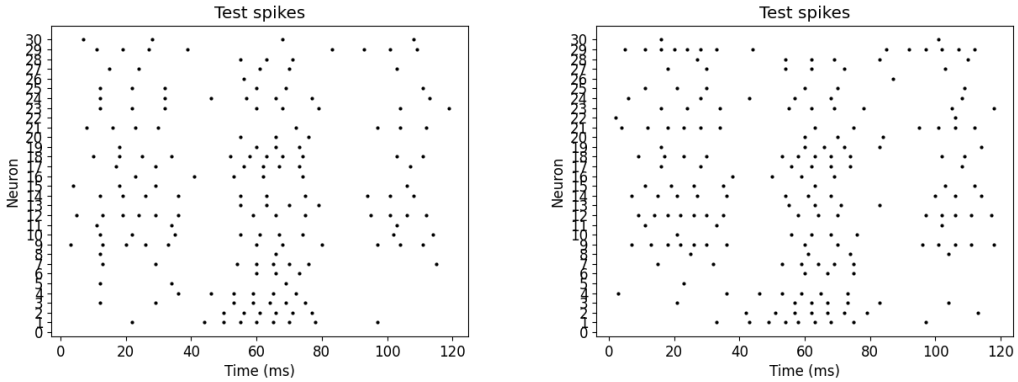

Figure B.26: Sample NLIF general predictive encoding experiment; the model spike train after epoch $t_{iter} = 100$ (left), and after training, $t_{iter} = 200$ (right).

LIF

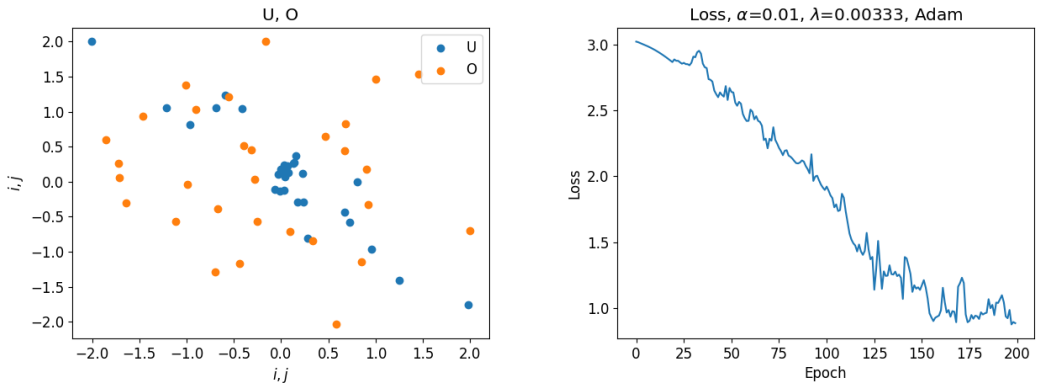

Figure B.27: Sample LIF general predictive encoding experiment; final weights and loss across epochs.

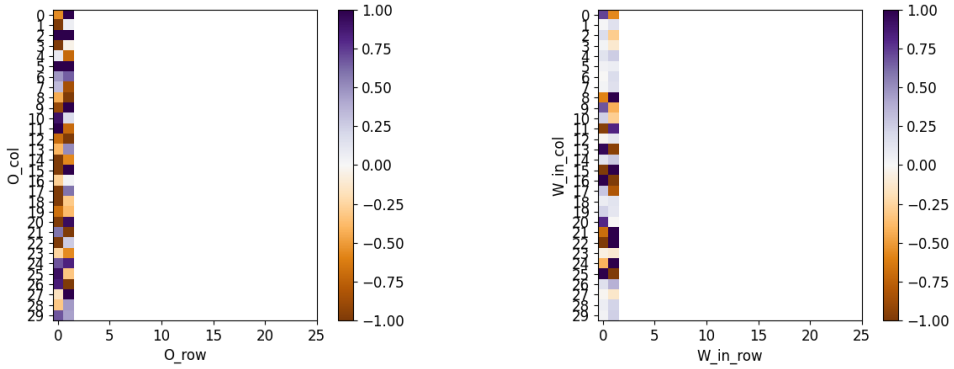

Figure B.28: Sample LIF general predictive encoding experiment; input and readout weights after training.

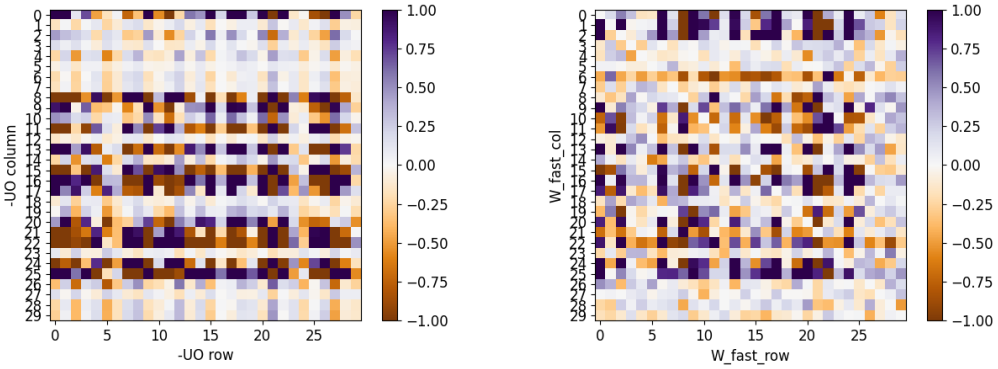

Figure B.29: Sample LIF general predictive encoding experiment; negative product of input and output weights, and the fast synaptic weights.

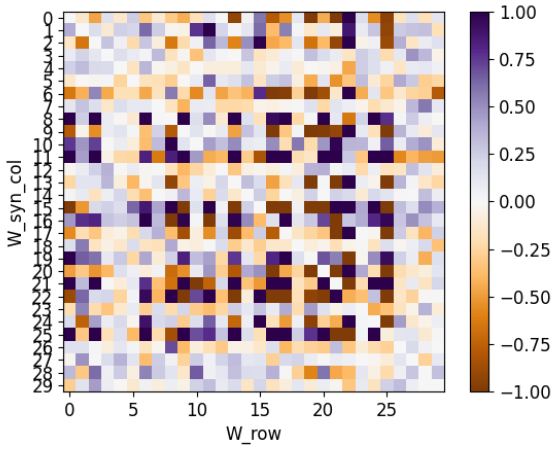

Figure B.30: Sample LIF general predictive encoding experiment; the synaptic weights.

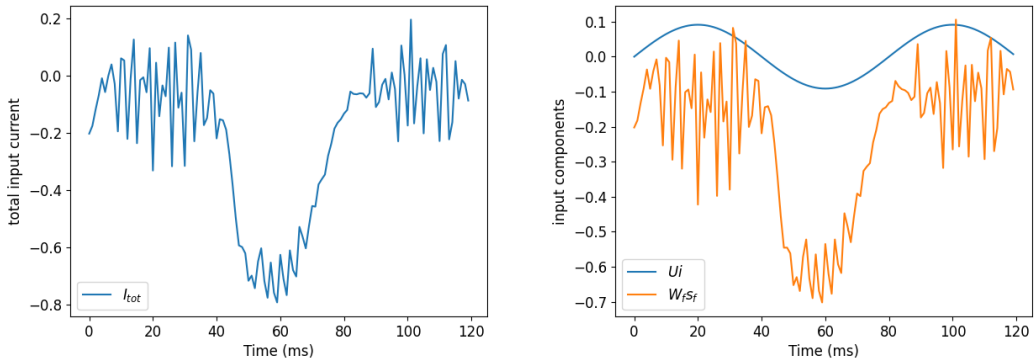

Figure B.31: Sample LIF general predictive encoding experiment; the input current and input components.

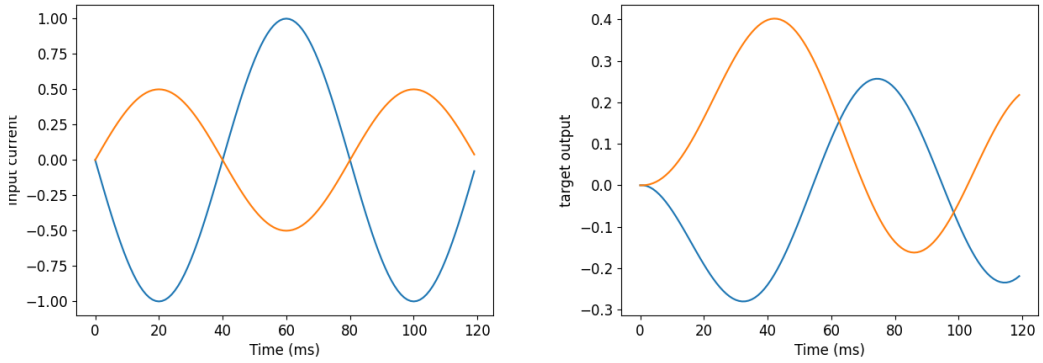

Figure B.32: Sample LIF general predictive encoding experiment; the inputs and targets.

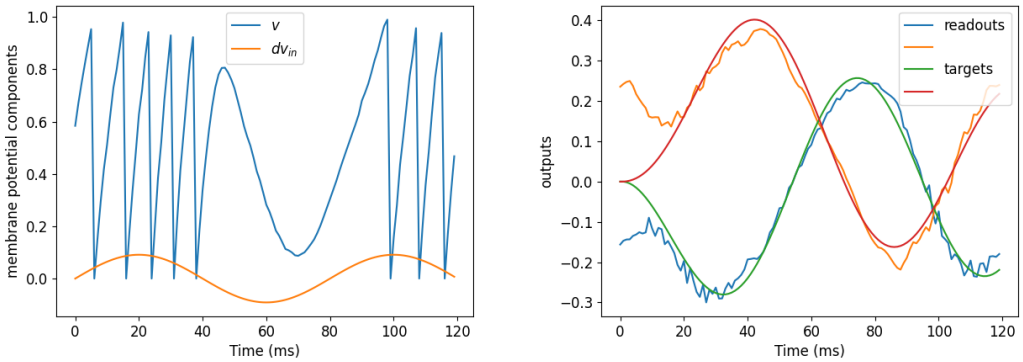

Figure B.33: Sample LIF general predictive encoding experiment; the membrane potentials and readouts.

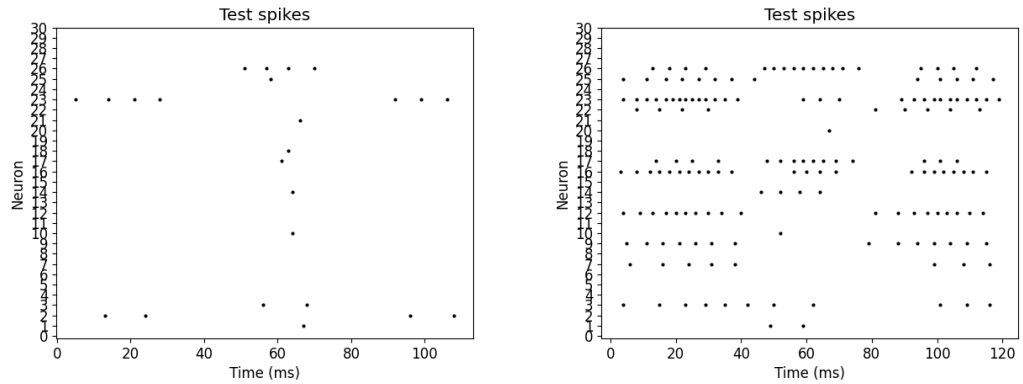

Figure B.34: Sample LIF general predictive encoding experiment; the model spike train after epoch $t_{iter} = 360$ (left), and after training $t_{iter} = 200$ (right).

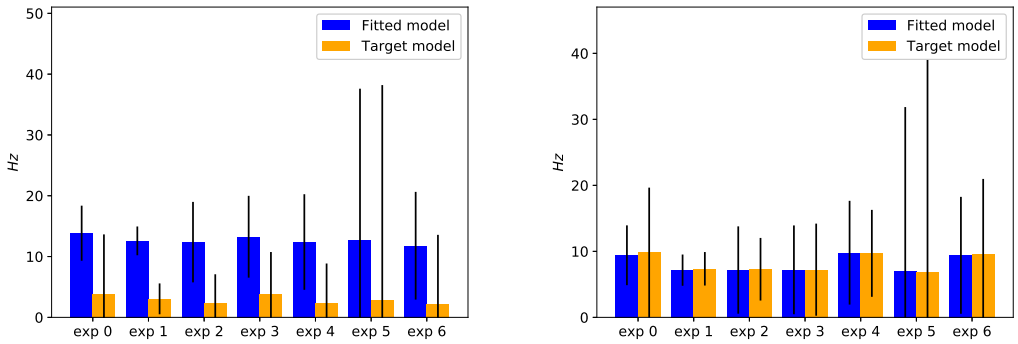

Figure B.35: Inferred model firing rates per data set for the (non-Dales compliant) LIF model (left), and GLIF model (right), using white noise with a fixed rate as input perturbation.