Okinawa Institute of Science and Technology
Graduate University

Thesis submitted for the degree

Doctor of Philosophy

Spatial and Modular Regularization in Effective Connectivity Inference from Neural Activity Data

by

Jessica Verena Schulze

Supervisor: **Prof. Dr. Kenji Doya**

December, 2018

Declaration of Original and Sole Authorship

I, Jessica Verena Schulze, declare that this thesis entitled *Spatial and Modular Regularization in Effective Connectivity Inference from Neural Activity Data* and the data presented in it are original and my own work.

I confirm that:

- No part of this work has previously been submitted for a degree at this or any other university.
- References to the work of others have been clearly acknowledged. Quotations from the work of others have been clearly indicated, and attributed to them.
- In cases where others have contributed to part of this work, such contribution has been clearly acknowledged and distinguished from my own work.
- None of this work has been previously published elsewhere.

Date: December, 2018
Signature: [illegible]

Abstract

Spatial and Modular Regularization in Effective Connectivity Inference from Neural Activity Data

Previous studies of effective connectivity inference from neural activity data benefited from simple regularization approaches such as L1 regularization, which promotes sparseness of the connection matrix. In this thesis we investigate the incorporation of two novel physiologically plausible priors based on spatial and modular organization of the neural circuit in the framework of Bayesian inference.

First we formulate a spatial prior which incorporates distance-dependent connectivity in the linear non-linear Poisson (LNP) model. We consider distance-dependent L1 and L2 regularization of connection weights as well as a hierarchical prior with distance-dependent connection probability. We derive maximum a posteriori (MAP) estimation algorithms by gradient descent, Newton method, and Metropolis-Hastings sampling. We test the effectiveness of these algorithms using synthetic data based on physiologically realistic distance-dependent connection weights and clarify the effects of the regularization parameter and data size, as well as the problems with highly synchronous firing and self-connections. The methods are also tested with calcium imaging data from the mouse posterior parietal cortex (PPC).

Next we formulate a modularity prior which assumes multiple modules in a network and different within-module and between-module weight distributions. We formulate a MAP inference by combining Gibbs sampling for module membership and Newton's method for connection weights. The method is validated by synthetic data with various modular structures, including spatially localized modules with distance-dependent connections.

Acknowledgment

A PhD is a journey on, many a times, troubled waters. Many people helped me along the way of this journey. I am particularly thankful to the following individuals:

My supervisor, Prof. Kenji Doya, for years of support in formulation, execution and interpretation of of this research project.

All members of the Neural Computation Unit, in particular, Prof. Junichiro Yoshimoto, former group leader of the unit, who has provided invaluable support around mathematical formulation and practical implementation questions.

Dr. Jean Lineard, Dr. Carlos Gutierrez, and Dr. Hiromichi Tsukada for discussions and feedback over many years.

Dr. Aki Funamizu and Prof. Bernd Kuhn for the experimental data set which inspired this work and is used in Chapter 3. Prof. Kuhn, as my third thesis committee member, also provided insightful discussions about bio-physiological matters.

Prof. Gail Tripp, my academic mentor, who installed in me more knowledge about academia and the human condition than I can ever thank her for. She supported me during the hardest times of my PhD and proof read this thesis to counterbalance my Kantian writing style.

Mariseth Ferring, my life-time mentor, who inspired me to reach high in my work but stay grounded as a person. Her wise words at the right time in my life built the foundation for many later decisions including choosing to study first computer science and later machine learning.

Former dean Prof. Jeff Wickens, who fought for us students as Dean of the Graduate School and continues to do so now.

Former president of OIST Johnathan Dorfan and Renee Dorfan for providing a vision far beyond this PhD program.

All members of the graduate school and student support staff, in particular Ms. Kozue Higashionna, who probably saved my life on multiple occasions.

Miss Mi and Miss Pipi for providing countless pancakes, coffees and much needed doses of realism during write up.

My sister and the rest of my family for believing in my goals despite never being quite sure that STEM is a viable career path.

And finally, Sandrine Anne Laure Burriel for always being right behind me, pushing me and sometimes catching my fall, in equal measures during all these years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Understanding neural circuits and their underlying principles has been the goal of large international projects such as the Brain/MINDS program in Japan [42], the Human Brain Project in the EU [34] and the BRAIN initiative in the US [25]. Neural connectivity is a core component in understanding neural circuits and how they contribute to information processing and computation in the brain. Networks in the brain can be analyzed at different scales. The following work focuses on the mesoscopic scale, that is connectivity of neural populations in their local circuit.

Advancement of neural recording techniques on the meso-scale, such as calcium fluorescence imaging enables neuroscientists to study neural populations on the scale of several hundred to a thousand neurons [41]. The technique of two-photon microscopy, that allows recording in specific cortical layers and location in-vivo, is a promising driving force [60, 61].

Such advancements in recording capability, and the resulting amounts of neural data, drive a need for the development of new computational and statistical methodologies for processing and analysis to efficiently utilize this wealth of data to the further our understanding of the brain [58].

Therefore, the overall goal of the research presented here is to contribute to this understanding by advancing methodologies for the analysis of neural circuitry and by studying their computational and information processing properties.

More specifically, we develop two new approaches for the inference of effective connectivity from neural recording data on the scale of several hundred neurons.

1.1 The Two Directions: Spatial and Modular

The two methods we investigate impose spatial and modular regularization respectively. Given the distinctive nature of these two directions we divide the main body of this work into two main chapters. Chapter 3 focuses on spatial regularized effective neural connectivity inference (SECI) and Chapter 4 focuses on modular regularized effective connectivity inference (MECI).

To give an initial idea of these two directions we paraphrase the two major assumptions underlying them:

1. Neurons that are spatially located closer to each have a higher probability of

being connected and might show stronger connections.

2. Neurons are organized in spatial or functional modules with stronger connectivity withing each module than those between different modules.

The development of the spatial approach in particular, was driven by the aim to analyze the data from two-photon microscopy recording of the parietal cortex of the mouse brain from a previous study in our research group [16].

To the best of our knowledge, no rigorous study of these two priors which go beyond previous methods that work with structural priors like the sparseness of neural connections [38, 59] has been conducted.

1.2 Structure

The remainder of this thesis is structured in the following way:

In Chapter 2 reviews the theoretical foundation which are needed to appreciate the main body of work, namely, probabilistic models of neural spike generation and Bayesian methods for the inference of their connection weight parameteres. Furthermore, related works and studies supporting the above mentioned two major assumptions are discussed.

Chapter 3 and 4 comprise the main body of this work and both follow a similar subdivision of theoretical formulation, algorithm derivation, evaluation, application and finally discussion of results.

Chapter 3 focuses on spatial effective connectivity inference (SECI). We formulate three variants of spike generation models (SECI-L1, SECI-L2 and Hi-SECI) and derive connectivity inference algorithms for them. We then evaluate performance of the respective algorithms on simulated data and apply them to experimental recordings.

Chapter 4 focuses on modular effective connectivity inference (MECI). We formulate a model with two different priors for connections within each module and between different modules. We then derive connectivity inference method using Gibbs sampling and Newton's method. The performance of the algorithms is tested on simulated data according to the assumption and a realistic neural network model.

Chapter 5 provides an overarching discussion of both approaches including a critically discussion of limitations and possible future directions.

Chapter 6 wraps up the whole work with a conclusion and summary of the key points.

Chapter 2

Theoretical Foundation of Connectivity Inference and Related Works

In this chapter we first introduce the theoretical foundation needed to appreciate later chapters. Then we review related works and discuss literature which supports the assumptions on which basis we construct the rest of this research.

The focus of this work is on the mesoscopic scale of connectivity inference which is concerned with connectivity between neurons in a local region of up to several hundred neurons.

2.1 Definitions of Neural Connectivity

In the community studying neural connectivity a very specific terminology has emerged that we intend to use in this text. We follow the use of connectivity terminology as first coined in the context of electrophysiological recordings [1] and generally being used in the fMRI community as seen in [14, 15] and well defined in [57].

On the anatomical level, connection between neurons to each other through synapses is referred to as *structural connectivity*. In contrast, the concept of *functional connectivity* describes the interaction between parts based on a deviation from the statistical independence. Functional connectivity is measured by correlation of neuron's activity, covariance or concepts like mutual information, but does not account for the direction of causation. If causal effect is taken into account through methods like time series analysis or model-based methods, we refer to it as *effective connectivity*.

In this thesis we are primarily investigating effective connectivity and will generally refer to it as such. It should be noted though, that in the connectivity research community a trend towards using the term functional connectivity for methods which are clearly effective rather than functional can be observed which sometimes can lead to initial confusion when discussing a method or putting it in the larger context research context.

2.2 Model-Free Connectivity Inference

Model-free non-parametric, descriptive statistics can be used for the inference of functional connectivity. There exists a broader array of methods than stated here. For a review see [10].

Cross-correlation. Classically, analysis of neural action potentials and spike train data was typically using correlation [2, 3] and related concepts like coherence. Such methods are descriptive statistics. In cross-correlation analysis a functional connection between two neurons is assumed if the Pearson correlation coefficient exceeds a certain threshold.
Cross-correlation is defined as:

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \tag{2.1}$$

where,

- cov is the *covariance*
- σ_X is the *standard deviation* of X.

Methods based on this and related concepts are still in wide use for functional connectivity analysis, particularly in the study of fMRI data [35].

JPSTH. In comparison to crosscorrelograms, which report correlations based on averages, that only tell you that there was a tendency for a neuron B to fire after a neuron A, *joint peristimulus time histogram* (JPSTH) can take into account the stimulus and report the dynamics of the correlation so you can tell when a neuron reacted in relation to the stimulus.

For a current definition see [19]. The above described methods are simple ways to obtain a description of the simultaneous activation of neurons, that is they all give you measurement of functional connectivity.

Mutual Information. Another popular measure is a concept from information theory called mutual information.

Mutual information a is defined as:

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x)\, p(y)} \right) \tag{2.2}$$

where, $p(x,y)$ is the *joint probability* of X and Y and $p(x)$, $p(y)$ are the marginal probabilities of X and Y, respectively.

Many more methods related to mutual information, such as *transfer entropy, generalized transfer entropy, permutation conditional mutual information* and *incremental mutual information*, exist and are applied to the analysis of neural connectivity [31, 45, 56].

Transfer entropy Transfer entropy is a non-parametric measure of the amount of transferred information between two random processes [53]. It is conditional mutual information with the history of the influence variable Y taken into consideration:

$$T_{X\to Y} = I(Y_t; X_{t-1:t-L}|Y_{t-1:t-L}) \tag{2.3}$$

It reduces to *Granger Causality* for Gaussian variables [5]. Transfer entropy is an example of a model-free method which is directional. In general though, model-free methods more often fall into the category of functional connectivity analysis and don't take direction of causality into consideration. We now move on to the discussion of model-based methods which make it easier to take direct of influence into account.

2.3 Model-Based Connectivity Inference

Model-based methods for connectivity inference assume certain mathematical models of neuronal dynamics and synaptic interactions and then estimate model parameters from observed experimental data, which makes them more rich than model-free methods and allows us to take direction of causality into account.

In model-based connectivity inference is thus divided in two main components:

1. Construct a neural network model
2. Estimate model parameters from observed data

For each step we have a variety of choices. Examples of popular neural models are the generalized linear model (GLM), stochastic leaky integrate-and-fire model, Hodgkin and Huxley model, and network likelihood model. Major frameworks for model parameter estimation are the maximum likelihood estimate (MLE) and the Bayesian maximum a posterior (MAP) estimate. For an in depth overview of models and methods in the context of neural connectivity inference we refer to the following recent review paper: [10].

In the following section we will focus on giving the theoretical background of the general framework, generalized linear model (GLM), and a specific model, linear-nonlinear-Poisson (LNP) model that we will use in this work. Furthermore, we present the general frameworks for parameter estimation, maximum likelihood and Bayesian inference, as well as specific algorithms we employ in following chapters.

2.3.1 Modeling Neurons

We will first introduce the notion of the *Spike Response Model* (SRM) and then work our way to a general model type *Generalized Linear Model* (GLM), which is used in several recent neural connectivity inference method. These are basic textbook level models that are fundamental as building blocks for more complex models. The text we direct the reader to for more detail is [20], to which we refer all definitions in this section unless stated otherwise.

Then we move on to show the formulation of a more specific model, the *Linear Non-linear Poisson* (LNP) cascade model. We first discuss this model in its general form and then explain how we adapt it for more specific use in later chapters.

Spike Response Model (SRM). In the formulation of the SRM, the membrane potential follows the equation

$$u(t) = \int_0^\infty \kappa(s) I(t-s) ds + \int_0^\infty \eta(s) S(t-s) ds + u_{rest}, \tag{2.4}$$

where $\kappa(s)$ is the membrane filter, a function that describes the impulse response of the neuron, I the input current, $\eta(s)$ a function describing the form of the action potential and the after-hyperpolarization, S the spike train, and u_{rest} the resting membrane potential. t is the time and s is the time back into the history of the spike train S and the input current I.

Spikes are triggered when the membrane potential $u(t)$ reaches a threshold $\vartheta(t)$. This threshold can be dynamically moving, which is one of the key features of the SRM.

Generalized Linear Model (GLM). We now transfer the above defined Spike Response Model into the Generalized Linear Model, which will serve as the model for single neurons in our method.

We discretize the time by Δt and represent the input response η, the spike response κ, and the resting membrane potential u_{rest} using a vector

$$\mathbf{k} = (\kappa(\Delta t), \kappa(2\Delta t), ..., \kappa(K\Delta t), \eta(\Delta t), \eta(2\Delta t), ..., \eta(J\Delta t), u_{rest}). \tag{2.5}$$

The sum of synaptic inputs I and the spike counts in each time bin are represented by a vector

$$\mathbf{x}_t = (I_{t-\Delta t}, I_{t-2\Delta t}, ..., I_{t-K\Delta t}, n_{t-\Delta t}, n_{t-2\Delta t}, ..., n_{t-J\Delta t}, 1). \tag{2.6}$$

Using these vectors we can represent the membrane potential dynamics in discrete-time as

$$u(t) = \sum_{k=1}^{K} \kappa_{k\Delta t} I(t-k\Delta t) + \sum_{j=1}^{J} \eta_{j\Delta t} n_{t-j\Delta t} + u_{rest} = \mathbf{k} \cdot \mathbf{x}_t. \tag{2.7}$$

For a single neuron the firing rate is given by

$$\rho(t) = f(\mathbf{k} \cdot \mathbf{x}_t - \vartheta), \tag{2.8}$$

where $f(u) \geq 0$ is a monotonically increasing firing rate function and ϑ is the firing threshold. In a network of N neurons, the total synaptic input to neuron i is given by

$$I_i(t) = \sum_{j=1}^{N} w_{ij} S_j(t), \tag{2.9}$$

if we assume that the synaptic delay is negligible.

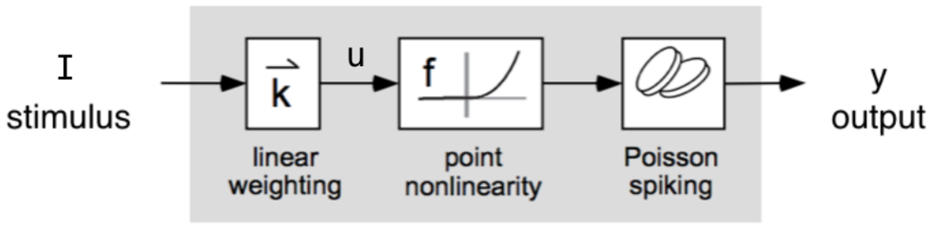

Figure 2.1: Block diagram of the LNP model modified from [55].

Linear-nonlinear Poisson (LNP) cascade model The linear non-linear Poisson (LNP) cascade model [9, 54, 55] is a cascade of three stages, as illustrated in Figure 2.1: a linear temporal filter of membrane potential, then a non linear firing rate function, and finally Poisson spiking. It is a simplified functional model of neural spike responses that is more mathematically tractable approximation to more bio-physiologically accurate models. It ties into the broader concept of GLMs. If the non-linearity is a fixed inversible function the model is in fact a GLM with the non-linearity as a link function. The LNP model is a subset of the GLM that omits the post-spike response filter η. Upon each time step, spike count $y(t)$ is independently sampled by a Poisson distribution with the expected spike count $f(u(t))\Delta t$. The general parametric form of the LNP model [47] is given by

$$P(spike|I) = f(\mathbf{k}I), \tag{2.10}$$

where $\mathbf{k}$ is a linear filter. The non linearity is introduced via f. For the input filter $\mathbf{k}$, exponential decay with a time constant τ

$$k_i = \exp(-\frac{i\Delta t}{\tau}) \tag{2.11}$$

is often assumed. For the firing rate function f, an exponential function

$$f(u) = \exp(u) \tag{2.12}$$

is commonly used.

Assuming the synaptic current after spike generation is sharp without transmission delay we can extend this model from single neuron to network model. Thus a LNP model for a network of N neurons is given as follows.
Total synaptic current:

$$I_i(t) = \sum_{j=1}^{N} w_{ij} y_j(t) \tag{2.13}$$

Membrane potential:

$$u_i(t) = \exp(-\frac{\Delta t}{\tau})u_i(t - \Delta t) + I_i(t - \Delta t) \tag{2.14}$$

Firing rate:

$$r_i(t) = \exp(u_i(t)) \tag{2.15}$$

Spike count:

$$P(y_i(t)) = \frac{(r_i(t)\Delta t)^{y_i(t)}}{y_i(t)!} \exp(-r_i(t)\Delta t) \tag{2.16}$$

Specifics of LNP model regrading how we use it in later chapters We now slightly restructure this model and transition in notation for later convinence. In the context of neural connectivity inference, we assume the membrane filter k is an exponential function with same time constant τ for all neurons, which allows us to replace the order of linear summation and linear filtering as seen below.
Filtered spike trains:

$$x_j(t) = \exp(-\frac{\Delta t}{\tau})x_j(t-\Delta t) + y_j(t-\Delta t) \tag{2.17}$$

We include spike history effects by updating $x_j(t)$ using Equation 2.17 which adds a decaying history effect. Thus the membrane potential can now be written as:

$$u_i(t) = \sum_{j=1}^{N} w_{ij}x_j(t) \tag{2.18}$$

Spike trains Y are produced based on the connectivity weights and spike history of all neurons in the network. The firing rate can thus be rewritten as:

$$r_{ti} = \exp\left\{\sum_{j=1}^{N} w_{ij}x_{jt}\right\} = \exp\left\{w_i^\top x_t^\top\right\}. \tag{2.19}$$

The shift in notation seen in Equation 2.19 will be used in later chapters.

The model is beneficial to our later application, because we can prove that the log-likelihood function is concave with respect to the model parameters based on the same discussion as [47]: This implies that any locally maximal solution should be the global maximum.

2.3.2 Estimating Model Parameters

We assume a model that produces output Y, given the input X and the model parameters θ, $P(Y|X,\theta)$.

We want to estimate the model parameters θ, from the observed output Y under input X, such that we are able to compute $P(Y|X,\theta)$, the response probability, which allows us to predict response to new unseen input.

A basic framework to estimate the model parameters is the maximum likelihood (ML) estimate:

$$\theta_{ML} = \arg\max_{\theta} P(Y|X,\theta). \tag{2.20}$$

In the Bayesian approach, the common framework is the maximum a posteriori (MAP) estimate:

$$\theta_{MAP} = \arg\max_{\theta} P(\theta|Y,X) = \arg\max_{\theta} P(Y|X,\theta)P(\theta) \tag{2.21}$$

where $P(\theta)$ is the prior distribution of the parameters.

This maximization can be achieved with different standard optimization algorithms. Example are well known methods such as *Gradient Decent* or *Newton's Method* as discussed below.

Regularization and Bayesian Interpretation

Regularization in machine learning is a technique to prevent over-fitting.

It is though in a Bayesian interpretation related to the prior distribution. Often just called the prior. The prior is a distribution over the model parameters that incorporates our beliefs of how the system behaves without taking any evidence into account.

L1 and L2 regularization Two of the most well known regularization are L1 and L2 norm. L1, often called Lasso regression, adds an absolute magnitude as a penalty term whereby achieving a more sparse solution:

$$\lambda \sum_{i=1}^{N} |w_i| \tag{2.22}$$

The influence of this simple sparseness prior on neural connectivity inference has been investigated by [38, 59] which resulted in the best performance for neural connectivity inference up to date.

L2, which is closely related and also called ridge regression, gives a penalty term of squared magnitude of the coefficients:

$$\lambda \sum_{i=1}^{N} w_i^2 \tag{2.23}$$

Consider here that L1 and L2 regularization reduces to simple vanilla MLE with regularization parameter λ set to zero.

Iterative Optimization Methods

We now discuss two deterministic iterative optimization methods we will be using in subsequent chapters. Gradient decent and Newton's method.

Gradient Decent Gradient Decent is a first-order optimization algorithm. It finds the minimum of a function F by taking steps from a starting point x in the direction of the negative gradient (Opposite for gradient ascent).

$$x_{n+1} = x_n - \eta \frac{\partial F(x_n)}{\partial x} \tag{2.24}$$

Gradient decent can be slow and get stuck in zig-zack type motion over the surface. When the function F is convex Newton's method can remedy this drawback by using the second-order derivative.

Newton's Method This method is generally well known as it is taught in school as a technique to find zeros of a function but the same concept can be used for optimization purposes. In contrast to gradient decent where we take the gradient and follow its direction to find an optima Newton's method is a 2nd order method that takes into account both the 1st order (gradient) and 2nd order (Hessian). Newton's method generally converges faster and doesn't exhibit zig-zag motion and is thus preferable to Gradient Decent if we have access to the Hessian. Non-differentiability can pose a problem.

The general formulation of Newton's method is:

$$x_{n+1} = x_n - \left(\frac{\partial^2 F(x_n)}{\partial x \partial x^T}\right)^{-1} \frac{\partial F(x_n)}{\partial x} \tag{2.25}$$

Sampling Methods

In the Bayesian paradigm it is often not possible to compute the posterior in closed form. In this case approximation by using sampling methods is a popular option. We will be using two sampling methods in later chapters. Metropolis-Hastings sampling and Gibbs sampling. These are both Markov chain Monte Carlo (MCMC) Methods.

Markov chain Monte Carlo Methods Markov chain Monte Carlo (MCMC) methods are a class of sampling algorithms. They depend on the Markov property [33], which says each sate of a chain only depends on its previous state:

$$\Pr(X_{n+1} = x \mid X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \Pr(X_{n+1} = x \mid X_n = x_n) \tag{2.26}$$

MCMC methods are often used in Bayesian statistics for approximation. Many MCMC methods are random walk methods. We introduce two of them below which we will be using in later chapters of this thesis.

Metropolis-Hastings

The Metropolis-Hasting algorithm [21, 36] is a Markov chain Monte Carlo sampling algorithms. It generates a chain of samples such that with increased number of samples it approximates the probability distribution.

1. Init
   (a) Pick initial candidate
2. Iterate
   (a) Randomly generate new candidate
   (b) Calculate probability Q

(c) Accept or reject candidate:
  i. $u \leftarrow$ uniform random number between [0, 1]
  ii. if $u \leq Q$: accept and keep new candidate
  iii. if $u > Q$: reject and keep old candidate

Gibbs sampling

Gibbs sampling [18] is a special case of Metropolis-Hastings [7]. It lets us approximate a joint distribution $P(\theta_1, \theta_2)$ by sampling from $P(\theta_1|\theta_2)$ and $P(\theta_2|\theta_1)$ which on the long run converges to a draw from the joint distribution.

After setting an initial starting values for $\theta_1^{i=0}$ and $\theta_2^{i=0}$ we thus sample in each iteration of the Gibbs sampling algorithm for an updated value of θ_1 and θ_2.

1. $\theta_1^i \sim P(\theta_1|\theta_2^{i-1})$
2. $\theta_2^i \sim P(\theta_2|\theta_1^i)$

This produces a chain in the process.

Convergence in MCMC Methods

Making good decisions about convergence is difficult for MCMC algorithms in general as you can't prove convergence [17].

A variety of convergence diagnostics for MCMC methods exist[8, 17, 51].

A practical way of determining convergence is to plot chains and reason about how stable they have become, which is the approach we chose. We decide likely convergence by plotting and re-running estimations and when it looks like results are stable and similar results are achieved from different starting points we consider convergence to be likely.

2.4 Related Works

We have discussed theoretical foundation so far. We now turn towards reviewing more recent related works and studies which motivated the two projects discusses in later chapters and which support the core assumptions underlying these two projects.

GLMs are often used in maximum likelihood approaches [47]. Another successful application is in the connectivity inference of the retina [49]. Other approaches suggest the use of network likelihood model[43] for the analysis of neural connectivity.

Bayesian approaches allow the incorporation of prior domain knowledge like in the example of [59] who proposed a maximum a priori approach with combined sparseness and smoothness of neural interaction. Another influential Bayesian approach [38] investigated the incorporation of a sparseness prior in network inference from calcium fluorescence data in combining connection inference and spike inference from calcium imaging data. This is a computationally costly MCMC approach. A potentially computationally less costly approach than [38] is [12] which proposes the use of approximal message passing instead of MCMC.

2.4.1 Underlying Assumptions

In the remainder of this work we investigate two different directions in which to extend Bayesian neural connectivity inference methods. These two directions rely on two specific assumptions for which we will discuss support now.

Support for Project 1: Spatial Prior

The hypothesis that neurons located spatially closer to each exhibit higher probability and stronger connections is well established. [46] provide connection probability measurements of up to a distance of 150 μm for layer 2/3 pyramidal neurons in the mouse auditory cortex. This idea is further supported by bio-physiological measurements of up to 300 μm of distance for layer 5 pyramidal neurons in the somato-sensory cortex [48] and for several cell types in the mouse auditory cortex [30]. While measurements differ for cell types and area the peak probability is generally around 0.2 with a bell curved reduction of probability with distance towards around 0.05 for neurons 300 μm apart.

Support for Project 2: Modular Prior

Modular organization in the brain can be found on different levels. On the macro scale anatomical regions that contribute to specific functions have been mapped in detail for many species [22, 29].

A recent method for connectivity inference from fMRI data on the macro-scale [50] proposed the incorporation of a modularity prior in the estimation of functional brain networks using a matrix rank that encodes how connected or disconnected a graph is.

On the meso-scale cortical columns [23, 40] are another well known modular structure organized by functionality.

To express modular structure in form of a prior we assume different connectivity statistics of the distribution of density or connectivity strength in different modules.

Chapter 3

Spatial Effective Connectivity Inference (SECI)

3.1 Introduction

Many bio-physiological studies have shown that connection probability and weight strength in biological neural networks is influenced by distance [30, 46, 48]. To the best of our knowledge no attempt has been made to incorporate this well known information into neural connectivity inference algorithms on the meso-scale. Prior studies have investigated the impact of priors like smoothness [58, 59] and sparseness [38]. In this chapter we investigate the merit of incorporating neural spacial information, specifically euclidean distance, into a Bayesian approach to neural connectivity inference for the regularization of both the strength of connections between neurons as well as the probability of connections.

The structure of this chapter is as follows. In the methods section we first give a theoretical formulation of a maximum likelihood estimation approach that imposes a combination of either L1-norm or L2-norm in combination with a distance-based regularization term over the weights of a connection matrix. As a model of neural behavior we employ the linear non-linear Poisson (LNP) cascade model. We then derive two algorithms, one with gradient decent for parameter optimization and distance regularization combined with L1-norm (SECI-L1) and one with Newton's method for parameter optimization and distance regularization combined with L2-norm (SECI-L2). We then extend these approaches to a hierarchical maximum a posteriori approach (HI-SECI) that imposes a prior over the existence of connections by utilizing Metropolis-Hastings sampling of connections. We then continue on to an in depth evaluation of these algorithms on a simulated data set including comparison of all three of them to simple L1/L2-norm regularization as well as inference without a prior. Furthermore, we apply these algorithms to a data set of in-vivo 2-photon microscopy recordings of the Parietal and Secondary Visual Cortex of the mouse brain. Finally, we conclude this chapter with a discussion of the main findings, critical considerations and future directions.

3.2 Methods

We now turn towards the presentation of the methods used in this part of the research project. We start with a direct adaptation for the L1 and L2-norm penalty term to incorporate knowledge about the distance between neurons and construct a maximum likelihood estimation approach with it. Then we move on to the extension of this formulation to a hierarchical maximum apriori approach.

3.2.1 Theoretical Formulation

We begin by presenting the mathematical formulation of a method that combines a distance-based regularization term with L2-norm. The reason for first introducing the L2 case is the ease of obtaining first and second order derivatives which allows us to use Newton's method for parameter optimization. As we have mentioned in previous sections Newton's Method converges faster than Gradient Decent and is thus preferable. Furthermore, the case of combining a distance-based regularization term with L1-norm and further derivation can easily be expressed by adapting the L2 case in two specific locations and dropping the derivation of the Hessian which poses a problem in the L1 case.

Derivation of a Spatial Regularized L2-Norm Inference Method

Notations

- N: the number of neurons of interest
- Δt: bin size for time descritization
- T: the number of time bins
- $Y \equiv [y_{ti}] \in \{0, 1, ...\}^{T \times N}$: Training data matrix representing spike train of all neurons of interest, from which the functional connectivity is estimated.
  - y_{ti}: spike count data in the tth time bin.
  - $y_t = [y_{t1}, \cdots, y_{tN}]$: spike counts of all neurons at the tth time bin.
  - Y can be written as $Y = [y_1^\top, \cdots, y_T^\top]^\top$.
- $X \equiv [x_{ti}] \in \mathbb{R}^{T \times N}$: Feature matrix extracted from the history of neural activity before the tth time bin.
  - x_{ti}: filtered spike train with the membrane time constant tau:

$$x_j(t) = \exp(-\frac{\Delta t}{\tau})x_j(t - \Delta t) + y_j(t - \Delta t)$$

  - X can be written as $X = [x_1^\top, \cdots, x_T^\top]^\top$.
- $W \equiv [w_{ij}] \in \mathbb{R}^{N \times N}$: Connectivity matrix

  - w_{ij}: connectivity weight from the jth to the ith neurons
  - $w_i = [w_{i1}, \dots, w_{iN}]^\top$: connectivity wegith vector projecting to the ith neurons.
  - W can be written as $W = [w_1, \dots, w_N]$

- $W_{-i} \equiv [w_1, \dots, w_{i-1}, w_{i+1}, \dots, w_N]$: matrix constructed by removing the ith column vector from connection matrix W.

- $D \equiv [d_{ij}] \in \mathbb{R}^{N \times N}$: distance matrix

  - d_{ij}: distance[1] between the center of the soma of the jth to the ith neurons.
  - $d_i = [d_{i1}, \dots, d_{iN}]^\top$: distance vector to the ith neurons.
  - D can be written as $D = [d_1, \dots, d_N]$

Generative model of the weights and spikes The prior of each component of the connection weight, w_{ij}, follows a Gaussian distribution with the mean $\mu_{ij} = 0$ and the standard deviation $\sigma_{ij} = 1/d_{ij}$. This implies stronger weight as the distance gets smaller. Assuming that all connection weights are independent of each other, the total distance-dependent prior can be written as

$$P(W) = \prod_{i,j=1}^{N} P(w_{ij}), \tag{3.1}$$

With

$$P(w_{ij}) = \frac{1}{\sqrt{2\pi\sigma_w^2}} \exp(-\frac{w_{ij}^2}{2\sigma_w^2}) \tag{3.2}$$

$$= \frac{1}{\sqrt{2\pi\frac{1}{d_{ij}^2}}} \exp(-\frac{w_{ij}^2}{2\frac{1}{d_{ij}^2}}) \tag{3.3}$$

$$= \frac{d_{ij}}{\sqrt{2\pi}} \exp(-\frac{d_{ij}^2 w_{ij}^2}{2}) \tag{3.4}$$

By taking the logarithm it follwos that

$$\ln P(W) = \sum_{i=1}^{N}\sum_{j=1}^{N} \left[-\frac{d_{ij}^2}{2} w_{ij}^2 + \ln d_{ij} - \frac{1}{2}\ln(2\pi) \right]. \tag{3.5}$$

The ***likelihood*** of the weight matrix W given the input feature matrix X and the spike output matrix Y based on definition of the linear-nonlinear Poisson cascade

[1]It should be noted that we do not discuss the scaling of d with respect to w in the theoretical formulation here. It is of practical importance but can easily be addressed by the introduction of a scaling parameter λ as seen in Section 3.2.2 and by keeping units in the simulation implementation consistent. For example we consistently work with μm for the distance in later sections and all other values depending on distance take this unit into consideration.

model is given as follows. Here we omit X that appears as the condition of all the distributions for simplicity.

$$P(Y|W) = \prod_{t=1}^{T}\prod_{i=1}^{N} P(y_{ti}; w_i) \tag{3.6}$$

where

$$P(y_{ti}; w_i) = \frac{(r_{ti}\cdot\Delta t)^{y_{ti}}}{y_{ti}!}\exp(-r_{ti}\cdot\Delta t), \tag{3.7}$$

with the firing rate defined as:

$$r_{ti} = \exp\left\{\sum_{j=1}^{N} w_{ij}x_{tj}\right\} = \exp\left\{w_i^\top x_t^\top\right\}. \tag{3.8}$$

Accordingly the ***log-likelihood*** becomes,

$$\begin{aligned}\ln P(Y|W) &= \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti}\left\{\ln r_{ti} + \ln\Delta t\right\} - \Delta t\cdot r_{ti} - \ln y_{ti}!\right] && (3.9)\\ &= \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti}\left\{w_i^\top x_t^\top + \ln\Delta t\right\} - \Delta t\exp\left(w_i^\top x_t^\top\right) - \ln y_{ti}!\right] && (3.10)\end{aligned}$$

The ***joint distribution of all stochastic variables*** takes the form

$$P(Y, W) = P(W)P(Y|W). \tag{3.11}$$

Taking the logarithm we obtain,

$$\begin{aligned}\ln P(Y,W) &= \ln P(W) + \ln P(Y|W) && (3.12)\\ &= \sum_{i=1}^{N}\sum_{j=1}^{N}\left[-\frac{d_{ij}^2}{2}w_{ij}^2 + \ln d_{ij} - \frac{1}{2}\ln(2\pi)\right] && (3.13)\\ &\quad + \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti}\left\{w_i^\top x_t^\top + \ln\Delta t\right\} - \Delta t\exp\left(w_i^\top x_t^\top\right) - \ln y_{ti}!\right] && (3.14)\end{aligned}$$

Conditional distribution **of w_i for each $i = 1, \cdots, N$**

$$P(w_i|Y, W_{-i}) = \frac{P(Y,W)}{\int P(Y,W)dw_i} \tag{3.15}$$

Thus the ***log conditional probability*** of w_i is,

$$\begin{aligned}
\ln P(w_i|Y, W_{-i}) &= \ln P(Y, W) + (w_i\text{-independent}) && (3.16)\\
&= -\sum_{i=1}^{N}\sum_{j=1}^{N}\frac{d_{ij}^2}{2}w_{ij}^2 && (3.17)\\
&\quad + \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti}w_i^\top x_t^\top - \Delta t\exp\left(w_i^\top x_t^\top\right)\right] + (w_i\text{-independent}) && (3.18)\\
&= -\frac{1}{2}\sum_{j=1}^{N}w_{ij}^2 d_{ij}^2 && (3.19)\\
&\quad + \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti}w_i^\top x_t^\top - \Delta t\exp\left(w_i^\top x_t^\top\right)\right] && (3.20)
\end{aligned}$$

Equation (3.19) without the negative can also we referred to as the *penalty term* γ as it encodes the penalizing effect of the weights and distance.

Now we can derive the ***gradient*** (first order derivative) with respect to w_i:

$$\begin{aligned}
\mathbf{g}_i &\equiv \nabla_{w_i}\ln P(w_i|Y, W_{-i}) && (3.21)\\
&= -d_i^2 w_i && (3.22)\\
&\quad + \sum_{t=1}^{T}x_t^\top y_{ti} - \Delta t\sum_{t=1}^{T}x_t^\top\exp\left(w_i^\top x_t^\top\right) && (3.23)\\
&= -d_i^2 w_i + \sum_{t=1}^{T}\left(y_{ti} - r_{ti}\Delta t\right)x_t^\top. && (3.24)
\end{aligned}$$

The ***Hessian*** (second order derivative) is given by

$$\begin{aligned}
\mathbf{H}_i &\equiv \nabla_{w_i}^2\ln P(w_i|Y, W_{-i}) && (3.25)\\
&= -\text{diag}\left[d_i^2\right] + \sum_{t=1}^{T}-\text{diag}(r_{ti}\Delta t)x_t^\top x_t. && (3.26)
\end{aligned}$$

Having access to the Hessian allows us to use Newton's method for optimization when constructing our algorithm in the next section.

Furthermore, if we set the maximum point of $\ln P(w_i|Y, W_{-i})$ as w_i^*, the following approximation becomes available:

$$\ln P(w_i|Y, W_{-i}) \approx -\frac{1}{2}\left(w_i - w_i^*\right)^\top\left(-H_i^*\right)\left(w_i - w_i^*\right) \tag{3.27}$$

Thus,

$$P(w_i|Y, W_{-i}) \approx \mathcal{N}(w_i; w_i^*, -[H_i^*]^{-1}), \tag{3.28}$$

where

$$w_i^* = \underset{w_i}{\operatorname{argmax}} \left[\ln P(w_i|Y, W_{-i})\right]. \tag{3.29}$$

$$H_i^* = -\operatorname{diag}\left[d_i^2\right] - \sum_{t=1}^{T} \operatorname{diag}(r_{ti}^* \Delta t) x_t^\top x_t. \tag{3.30}$$

$$r_{ti}^* = \exp\left\{(w_i^*)^\top x_t^\top\right\} \tag{3.31}$$

Computing the weight matrix W is an optimization problem of N^2 variables and can be separated into N terms to be optimized separately by optimization methods like Gradient Decent or Newton's method. Since we have derived the Hessian for SECI-L2 Newton's method is available to us.

Formulation for the L1 Based Approach

We choose to first derive a spatial regularized method that relies on L2-norm as it is straight forward to derive the Hessian for it, but the formulation of an L1-Norm based method is very similar. It is, however, difficult to derive the Hessian due to the existence of a non-differential point. We chose instead to work with the first order derivative alone and use Gradient Decent for parameter optimization. In this case, for the construction of an algorithm, we only need to adapt two terms: The penalty term γ and the gradient of the penalty term.

$$\gamma = \frac{1}{2}|w_i|d_i^2 \tag{3.32}$$

$$\nabla_{w_i}\gamma = sign(w_i)d_i^2 \tag{3.33}$$

These two equations directly compare to Equation (3.19) and (3.22) and allow us to skip reiterating the whole formulation for the L1 case.

Extension to a Hierarchical MAP Version

In this section we expand upon the idea of simple regularized Maximum Likelihood (ML) estimation and build a full Maximum A Posteriori (MAP) estimation approach, which sets the ground to derive an algorithm utilizes Metropolis-Hastings sampling to find good connections.

As we have mentioned before, while the above methods have a Bayesian interpretation they aren't explicitly computing a posterior in the pure sense. We show here how to add a prior over the existence of connections. This prior is imposed in a hierarchical manner. First the existence or absence of a connection is decided by this prior then in the second hierarchical step weight strength for the existing connections is estimated with any of the previously derived methods or just L1/L2 regularization.

In addition to the weight matrix W we now introduce a connection matrix C that populated with either 1s for active or 0s for inactive connections:

- $C \equiv [C_{ij}] \in \{0,1\}^{N\times N}$: Connectivity matrix
- C_{ij}: connectivity from the jth to the ith neurons
- $c_i = [c_{i1}, \ldots, c_{iN}]^\top$: connectivity vector projecting to the ith neurons.
- C can be written as $C = [c_1, \ldots, c_N]$

Recall the probability density of a normal distribution is given by:

$$\frac{1}{\sigma\sqrt{2\pi}} e^{-(\frac{x-\mu}{\sigma})^2} \tag{3.34}$$

We construct the connection probability for our method using a normal distribution without the normalization term, which is in our case can be dropped. Thus the connection probability can be calculated as follows:

$$P(C) = \exp\left[\ln(p_{max}) - \frac{1}{2}\left(\frac{d}{\sigma_{dist}}\right)^2\right] \tag{3.35}$$

Where μ and p_{max} and σ_{dist} either need to be determined by some form of cross validation or set reflecting known bio-physiological measurements. We set them in accordance with known bio-physiological measurements by [30] to $\mu = 0$, $p_{max} = 0.23$ and $\sigma_{dist} = 0.55$ to reflect our prior believe that spatially closer located neurons have a higher probability to be connected. The highest connection probability reported by [30] was 0.23 at maximum while rapidly declining with distance. The largest distance considered in studies we reviewed was $300\mu m$. The value $\sigma_{dist} = 0.55$ was obtained by fitting a Gaussian to the graphs reported in [30]. The value of d is normalized between 0 and 1, where 1.0 corresponds to 300 μm, which is a the largest distance for which we have found bio-physiological data concerning connection probability depending on distance. As we have discussed in Chapter 2 measurements form different areas and cell types slightly differ and we made a judgment call to stick to the values presented in [30].

The log posterior is computed from the log likelihood - penalty term + log prior. The log likelihood and penalty term depend on what underlying method is chosen for weight optimization. You can choose either SECI-L1 or SECI-L2 or simple L1 L2-norm regularization.

$$\begin{align}
\ln P(w_{ij}, c_{ij}|Y) &\equiv \ln P(Y|W) - \gamma + lnP(C) \tag{3.36}\\
&= \ln P(Y|W) \tag{3.37}\\
&\quad - \gamma \tag{3.38}\\
&\quad + \sum_{i=1}^{N}[\sum_{j=1}^{N} c_{ij}(\ln(p_{max}) - \frac{1}{2}(\frac{d_{ij}}{\sigma})^2) \tag{3.39}\\
&\quad + \sum_{j=1}^{N}(1-c_{ij})(\ln[1 - exp(\ln(p_{max}) - \frac{1}{2}(\frac{d_{ij}}{\sigma})^2)])]. \tag{3.40}
\end{align}$$

We can now derive the algorithm form the three methods we introduced above. We will refer to the algorithms as SECI-L1 (Spatial effective connectivity inference with L1-norm), SECI-L2 (Spatial effective connectivity inference with L2-norm) and HI-SECI (Hierarchical spatial effective connectivity inference). We do not need to construct separate algorithm for an algorithm using simple L1-norm regularization and/or L2-norm regularization as SECI-L1 simplifies to L1 regularized inference and SECI-L2 to L2 regularized respectively when setting D to be the unitary matrix (Matrix of all 1s). This allows us to easily compare their respective performance in later sections.

3.2.2 Derivation of the Algorithms (SECI-L1, SECI-L2, HI-SECI)

We are now able to derive three algorithms SECI-L1, SECI-L2 and HI-SECI. All algorithms were implemented in Python3.5 and the source code and data sets used to produce graphs in this thesis will be publicly available at `https://github.com/oist/pynci/SECI` upon peer reviewed publication of this work. Below we derive pseudo-code for the discussion of our algorithms. In the pseudo-code we use certain abbreviations. Since we consistently work with the logarithm of many terms we chose to use L to stand for log-likelihood and l for log-posterior for brevity of notation in the algorithm's pseudo-code. Furthermore, we drop certain practical steps such as saving and loading from files, plotting etc. and only show the algorithm's core logic.

Implementation of SECI-L1 and SECI-L2

The optimization problem of finding the best weight matrix W can be separated into N terms. We define an outer estimation function in Algorithm 1 called estimate(). It loops over all regularization parameters λ and all neurons N. In the pseudo-algorithms a call to the function MODEL.fit() invokes what the underlying estimation method eg. SECI-L1 (see Algorithm 2) or SECI-L2 (Algorithm 3).

Discussion of SECI-L1 Pseudo-code This section discussed Algorithm 2 line by line. We first initialize w and calculate the initial log likelihood, penalty term and error term in line 2 - 5. The for-loop only ever runs to $Iter_{max}$ when convergence isn't achieved before we reach $Iter_{max}$. This is to ensure termination. In practice we test for this condition and produce a warning, but omit this test in the pseudo-code. Convergence is assumed when the improvement threshold is reached (line 18). In line 7 - 9 the gradient is computed. In line 11 - 17 we update weights depending on a step size and re-calculates the log likelihood, penalty and error term. As long as the new error isn't smaller than the old one we continue taking increasingly smaller steps. Then we keep the current results and check for convergence.

Discussion of SECI-L2 Pseudo-code This section discusses Algorithm 3 line by line. In line 2 - 5 the weight is initialized and the log-likelihood, penalty term and error term computed. In line 7 - 12 we compute the gradient and Hessian then solve for them in line 13. In line 15 we update w in the direction of the error then re-calculate log-likelihood, penalty term and error term. In line 19 we save current results then perform a convergence check in line 20.

Algorithm 1 Outer Estimation Loop

```
    **Input:** $Y, X, D, \lambda s, method$

    **Output:** Estimated weight matrix W

1: function ESTIMATE($Y, X, D, \lambda s, method$)
2:     for all $\lambda s$ do
3:         Initialize W
4:         $MODEL.init(method, \lambda)$
5:         for $i = 1, \cdots, N$ do
6:             $w \leftarrow MODEL.fit(Y_i, X, d_i)$
7:             $W_i = w$
```

Algorithm 2 SECI-L1

```
    Input: $Y, X, D, \lambda$
    Output: Estimated weight vector w
 1: function FIT()
 2:     Initialize w
 3:     $L \leftarrow \ln P(Y|W)$
 4:     $\gamma \leftarrow \frac{1}{2}\lambda \sum d_{ij}^2 |w_{ij}|$
 5:     $\epsilon \leftarrow -\frac{L}{N} + \gamma$
 6:     for $k = 1, \cdots, Iter_{max}$ do
 7:         $g_L \leftarrow \sum_{t=1}^{T} (y_{ti} - r_{ti}\Delta t)\, x_t^\top$
 8:         $g_\gamma \leftarrow \lambda sign(w) d^2$
 9:         $g_\epsilon \leftarrow -\frac{g_L}{N} + g_\gamma$
10:         $step \leftarrow 1.0$
11:         while $\epsilon >= \epsilon'$ do
12:             $w' \leftarrow w - \text{step} g_\epsilon$
13:             $L' \leftarrow \ln P(Y|W)$
14:             $\gamma' \leftarrow \frac{1}{2}\lambda \sum d_{ij}^2 |w_{ij}|$
15:             $\epsilon' \leftarrow -L + \gamma'$
16:             $step \leftarrow 0.5 step$
17:         $L \leftarrow L', \gamma \leftarrow \gamma', \epsilon \leftarrow \epsilon', w \leftarrow w'$
18:         if $\epsilon - \epsilon' <$ improvement threshold then
19:             $converged \leftarrow True, break$
```

Algorithm 3 SECI-L2

```
Input: $Y, X, D, \lambda$
Output: Estimated weight vector w
1: function FIT()
2:     Initialize w
3:     $L \leftarrow \ln P(Y|W)$
4:     $\gamma \leftarrow \frac{1}{2}\lambda \sum d_{ij}^2 w_{ij}^2$
5:     $\epsilon \leftarrow -\frac{L}{N} + \gamma$
6:     for $k = 1, \cdots, Iter_{max}$ do
7:         $g_L \leftarrow \sum_{t=1}^{T} (y_{ti} - r_{ti}\Delta t)\, x_t^\top$
8:         $g_\gamma \leftarrow \lambda w d^2$
9:         $g_\epsilon \leftarrow -\frac{g_L}{N} + g_\gamma$
10:        $h_L \leftarrow \sum_{t=1}^{T} -\text{diag}(r_{ti}\Delta t) x_t^\top x_t$
11:        $h_\gamma \leftarrow \lambda diag(d^2)$
12:        $h_\epsilon \leftarrow -\frac{h_L}{N} + h_\gamma$
13:        Solve system of equations for $h_\epsilon$ and $g_\epsilon$
14:        while $\epsilon >= \epsilon'$ do
15:            Update w in the direction of the solution with increasingly smaller step
   size
16:            $L' \leftarrow \ln P(Y|W)$
17:            $\gamma' \leftarrow \frac{1}{2}\lambda \sum d_{ij}^2 w_{ij}^2$
18:            $\epsilon' \leftarrow -L + \gamma'$
19:        $L \leftarrow L', \gamma \leftarrow \gamma', \epsilon \leftarrow \epsilon', w \leftarrow w'$
20:        if $\epsilon - \epsilon' <$ improvement threshold then
21:            $converged \leftarrow True, break.$
```

Implementation of HI-SECI

HI-SECI is a hierarchical maximum a posteriori (MAP) estimation algorithm for effective connectivity inference. In contrast to SECI-L1 and SECI-L2 an additional distance prior is enforced in a hierarchical manner over the existence of connections C before estimating the weights W. HI-SECI utilizes Metropolis-Hastings sampling to sample connection assignments, which is a Markov chain Monte Carlo (MCMC) method. For a candidate connection assignment C' weights are estimated using either SECI-L1 or SECI-L2 and the log posterior calculated. A given candidate is then either accepted or rejected based on the posterior probability. The log posterior is calculated in accordance with Equation (3.36) from the log likelihood L and penalty term γ as shown in Algorithm 4. L and γ are determined by the underlying estimation method (eg.: SECI-L2 or SECI-L1).

Algorithm 4 logpost

```
    Input: L: log-likelihood, $\gamma$: penalty term, c: connectivity vector, i: neuron index
    Output: l: log-posterior
1: function LOGPOST(L, $\gamma$, C, I)
2:     l ← L - $\gamma$
           $+\sum_{j=1}^{N} c_{ij}(ln(p_{max}) - \frac{1}{2}(\frac{d_j}{\sigma})^2)$
           $+\sum_{j=1}^{N}(1 - c_{ij})(ln[1 - exp(ln(p_{max}) - \frac{1}{2}(\frac{d_j}{\sigma})^2))]$
```

The full algorithm relies on an underlying inference method like SECI-L1 or SECI-L2. In Algorithm 5 a call to function called MODEL always means the underlying method is invoked for estimation. In Algorithm 5 the following terms are used: γ the penalty-term. L the log likelihood. l the log posterior, l' temporary log posterior, l^* current best log posterior. First we initialize the weight and connection matrix. Then the underlying estimation method is initialized with MODEL.init(method, λ). For each neuron we then take an initial random guess at a candidate for the connectivity and make an initial evaluation. Then we enter the sampling loop in which new candidates for the connectivity are tested and either accepted or rejected.

Algorithm 5 HI-SECI

```
Input: Y, X, D, $\lambda$, method.
Output: $W^*$, $C^*$.
1: Initialize $W$ and $C$ with zeros
2: MODEL.init(method, $\lambda$)
3: for $i = 1, \cdots, N$ do
4:     Initalize $c_i$ randomly
5:     Select $X_{ci}$, $Y_i$ and $D_{ci}$ from $X$, $Y$ and $D$ in accordance with $c_i$
6:     Get initial estimate for $w_i$, $L$, $\gamma$ from $MODEL.fit(Y_i, X_{ci}, D_{ci})$
7:     $l = logpost(i, c_i, d_i, L, \gamma)$
8:     $c_i^* \leftarrow c_i, l^* \leftarrow l, w_i^* \leftarrow None$
9:     for $h = 1, \cdots, H_{max}$ do
10:        for $j = 1, \cdots, N$ do
11:            if $j! = i$ then
12:                $c_i' \leftarrow c_i$
13:                $c_{ij}' \leftarrow$ not $c_{ij}'$
14:                if $\sum c_i' > 1$ then
15:                    Select $X_{ci}$, $Y_i$ and $D_{ci}$ from $X$, $Y$ and $D$ in accordance with $c_i'$
16:                    $w_i, L, \gamma \leftarrow MODEL.fit(X_{ci}, Y_i, D_{ci})$
17:                    $l' = logpost(i, c_i', d_i, L, \gamma)$
18:                    if $l' > l$ then
19:                        $c_i \leftarrow c_i', l \leftarrow l'$
20:                    else if $randnum < exp(l' - l)$ then
21:                        $c_i \leftarrow c_i', l \leftarrow l'$
22:                    if $l' > l^*$ then
23:                        $c_i^* \leftarrow c_i', l^* \leftarrow l', w_i^* \leftarrow w_i$
24:        $W_i^* \leftarrow w_i^*, C_i^* \leftarrow c_i^*$
```

Parameter	Symbol	Value
Spike train parameters		
Calculation step size for spikes	Δ_s	1 ms
Maximum probability	p_{max}	0.23
Maximum weight	w_{max}	3.0
Decay parameter for spikes	τ_s	5 ms
Basal rate for spikes	b_s	5.0
Absolute refractory period	ref	4 ms
Trace parameters		
Calculation step size for traces	Δx	30 ms
Noise variance	σ_{noise}	0.09
Decay over time	τ_x	2.0
Fixed Jump height	a_x	1.0
Basal activation	b_x	1.0

Table 3.1: Summary of essential model parameters used in SECI.

3.3 Performance Evaluation on Simulated Data

We now present an evaluation of SECI-L1, SECI-L2 and HI-SECI on simulated data.

3.3.1 Data Simulation

For the investigation of SECI we produce data sets defined by two steps. First a placement step, then a wiring step. In our set up network wiring is dependent on distance between neurons thus the the overall neural population activation is influenced by placement. All data sets used in this evaluation section were produced by selecting placement coordinates from a uniform distribution scaled by 300, leading to distances between 0 and 300 μm. Alternative placement methods are Gaussian or grid-based evened out placement.

The existence or absence of a connection in connection matrix C is then decided in accordance with Equation (3.39) and (3.40). For existing connections the connection weight in the weights matrix W is then set by a normal distribution with mean 0 and variance based on the distance between neurons. This influence is inversely proportional where a large distance leads to small weights and a small distance to large weights. We then select 20 percent of neurons to function as inhibitory neurons. In our model set up which knows no cell types this means their influence on other neurons is negative instead of positive. Furthermore, self-connections were set to 0.

Spike trains Y and explanatory variable X where then produced in accordance with the LNP model as described in Chapter 2. Specific model parameters are summarized in Table 3.1. The given values were used for generation of simulated data for spike trains and trace simulations unless otherwise stated for evaluation purposes.

3.3.2 General Notes Regarding Analysis

For evaluation on simulated data for each result presented in subsequent figures, we first performed a quick run estimation with low data length and over a coarse array of regularization parameters to obtain an idea in which area the best regularization parameters fall. For the full analysis we use data length in accordance with the data length requirements we discuss in section 3.3.4 unless otherwise stated. During this analysis we give plots of the form seen in Figure 3.1. Figure 3.1 (a) shows the performance achieved measured by Pearson r correlation over a range of regularization parameters λ. We initially considered various forms of performance measures such as different types of means squared errors and hamming distance but Pearson r correlation coefficient between the ground truth connection weights and the estimated connection weights was judged to capture the intuition we want to convey the best. Furthermore, you will encounter regression plots of the type shown in 3.1 (b) illustrating the difference between ground truth and estimated weights. These are always for one specific regularization parameter, usually the best one as in this figure. Other types of graphs we show are of the form seen in 3.1 (c) and (d). These are visualization of the weight matrix W of ground truth and the best estimation result respectively in form of a heatmap graph.

3.3.3 Impact of Highly Synchronized Neural Activity

During initial runs of SECI we observed a phenomenon that was also encountered by [38]. We found a strong a sensitivity to synchronicity in the neural spike trains. An illustration of high synchronicity can be seen in Figure 3.2. In simulations it occurs when wiring in a population is particularly strong leading to neurons firing in regular intervals in unison.

In particular, we found for very high synchronicity that the estimation result deteriorated below the performance of non regularized estimation to levels below $r = 0.6$. This behavior is shown in 3.3 for all methods. As all methods are affected similarly by this phenomenon we only show graphs produced with SECI-L2 for following illustrations in this section. Level of such high synchronicity as used to produce Figure 3.3 are most likely unrealistic compared to experimental in-vivo recordings. We investigate this phenomenon further at both ends of the spectrum from very low to very high synchronicity to establish corner cases for the behavior SECI.

To produce data sets with varying synchronicity we artificially increased or decreased the wiring strength and refractory period. Absolute total wiring strength is not the only factor that leads to high synchronicity in our simulation set up. As wiring is distance dependent closely located neurons might show high synchronicity.

To get a better insight into this phenomenon we compared estimation performance on an array of data sets with increasing synchronicity. In Figure 3.4 a summary of a data set from the low end with reasonable performance is shown while Figure 3.5 shows results for a data set on the high end.

When considering weight distribution of ground truth vs estimated weights we observed that most problematic weights were located away from the weight distribution mean in the far negative range and where influenced by synchronicity. Comparing Fig-

Example of estimation outcome by SECI-L2

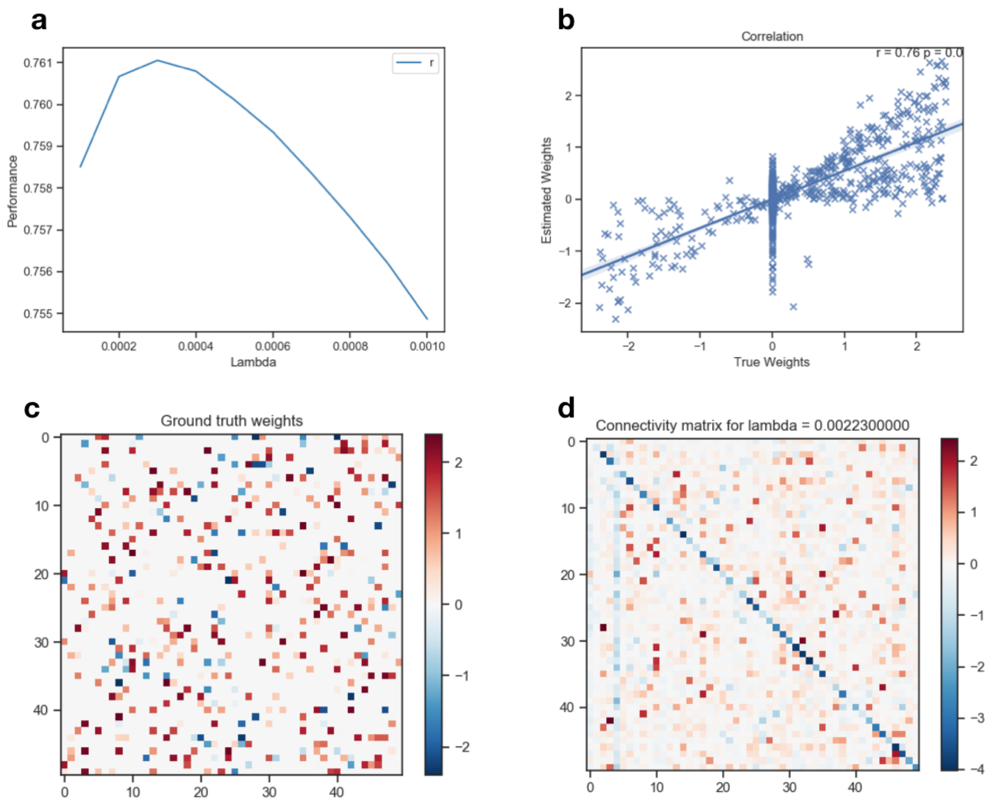

Figure 3.1: Example of an estimation result illustrating performance by showing **(a)** Pearson r over a range of regularization parameters λ **(b)** Regression plot between true weights (ground truth known from the simulation) and estimated weights for the estimation achieved with best $\lambda = 0.0023$ **(c)** Ground truth connection weight matrix **(d)** Estimated weights for best $\lambda = 0.0023$.

Illustration of highly synchronized firing in a simulation of 50 neurons

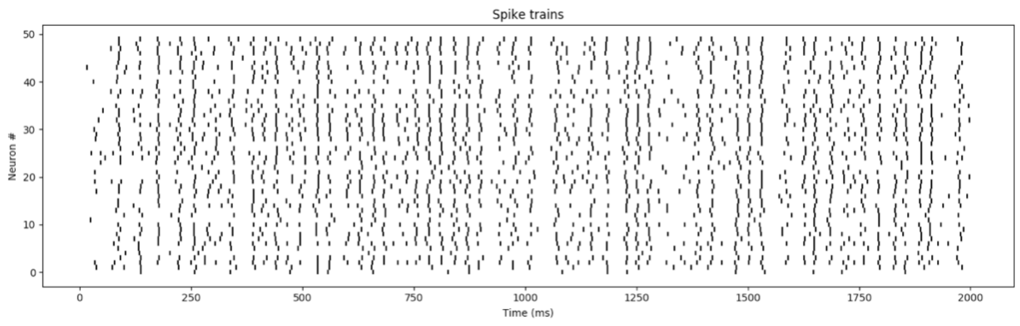

Figure 3.2: Example of high synchronicity in the spike trains of a population of 50 neurons.

Illustration of performance decline for all methods compared to non prior on highly synchronized data

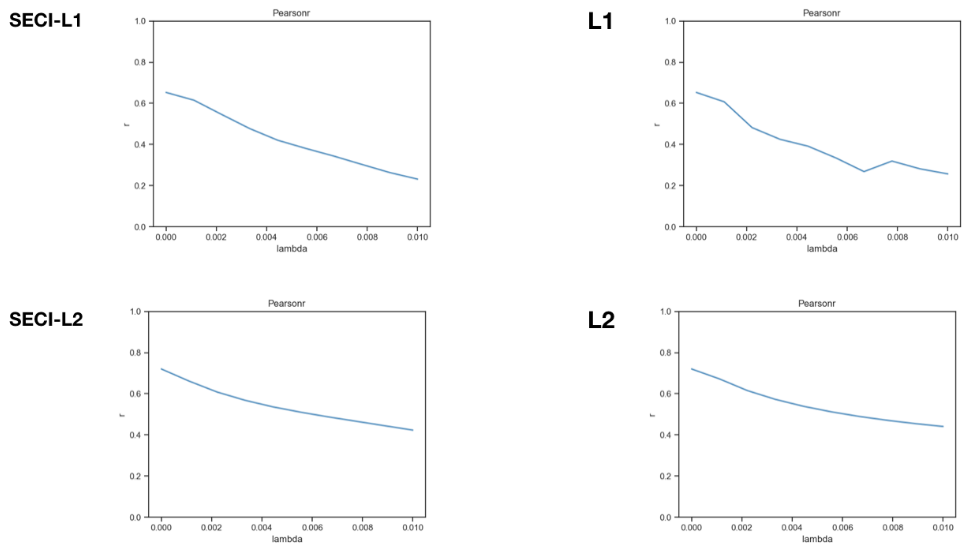

Figure 3.3: A rapid performance decline for all methods at very high end of synchronicity lets levels fall below non prior estimation performance.

ure 3.4 (j), which shows good fit for low synchronicity, to Figure 3.5 (j), which shows bad fit for high synchronicity, illustrates this point. Observation of the estimation matrix revealed strong negative weights on the diagonal which corresponds to self-connections. Recall that in the simulation set up our model sets self-connections to zero. The hypothesis as to why the estimation produces negative self-connections is that the estimation method use the extra available weight to encode influences for which we don't explicitly account for in the model parameters. Such parameter that influence the estimation of self-connection are such influences as refractory period and/or external inputs. In the case of these data sets no external influence exists therefore refractory period which can be expressed as negative self connections is the most likely reason for this. Thus we discovered the most of the performance drop was to self-connections diverging form the ground truth in which they are 0. We thus compared performance plots for all weights including self-connections vs excluding self-connections. A summary of this can be seen in Figure 3.6

While the major impact that lead to the observed performance decline is on self-connections synchronicity still has a strong impact on overall performance and can make it difficult to compare data sets. We artificially influenced synchronicity here but it dynamically rises from placement in our model set up and even seemingly similar placement can produce different levels of synchronicity influencing performance results. We need to find a way to quantify the synchronicity behavior to be able to select data sets which have the same level for comparison.

Performance analysis for low synchronicity

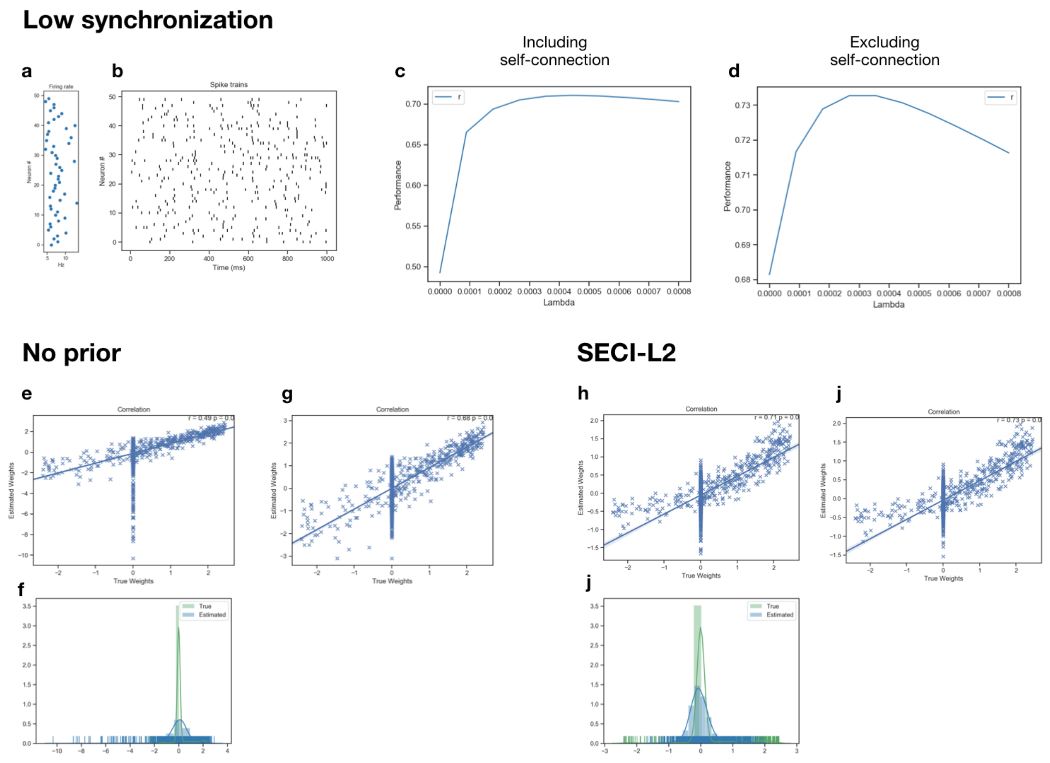

Figure 3.4: **(a)** Firing rate for a low synchronized data set. **(b)** Excerpt of the spike trains for the first 1000 ms of data set. **(c)** Performance measured for all weights over a range of regularization parameters. **(d)** Performance measured for connections excluding self-connections over a range of regularization parameters. **(e)** Regression plot of weights for an estimation without prior including self-connections. **(f)** Connection weight distribution for no prior run. **(e)** Regression plot of weights for an estimation without prior excluding self-connections. **(h)** Regression plot of weights for best performing estimation with SECI-L2 including self-connections. **(i)** Connection weight distribution for best SECI-L2 run on low synchronicity shows a good match of the weight distributions even when including self-connections. **(j)** Regression plot of weights for best performing estimation with SECI-L2 excluding self-connections.

Performance analysis for high synchronicity

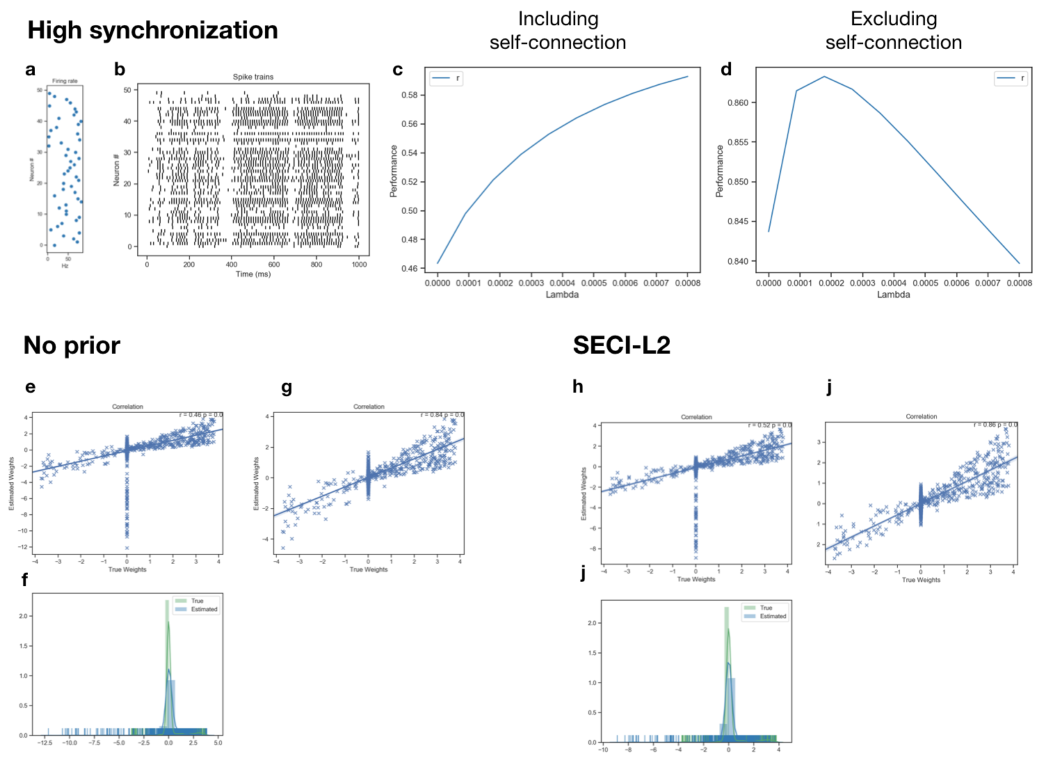

Figure 3.5: **(a)** Firing rate for a highly synchronized data set is still in a normal range. **(b)** Spike trains for the first 1000 ms show highly synchronized firing. **(c)** Performance measured for all weights over a range of regularization parameters. **(d)** Performance measured for connections excluding self-connections over a range of regularization parameters. **(e)** Regression plot of weights for an estimation without prior including self-connections. **(f)** Connection weight distribution for no prior run shows the mismatch in the negative range. **(e)** Regression plot of weights for an estimation without prior excluding self-connections. **(h)** Regression plot of weights for best performing estimation with SECI-L2 including self-connections. **(i)** Connection weight distribution for best SECI-L2 run shows mismatch in the weight distribution with most problems in the negative range. **(j)** Regression plot of weights for best performing estimation with SECI-L2 excluding self-connections shows how the overall performance score is influenced by self-connections.

Comparison of results including and excluding self-connections

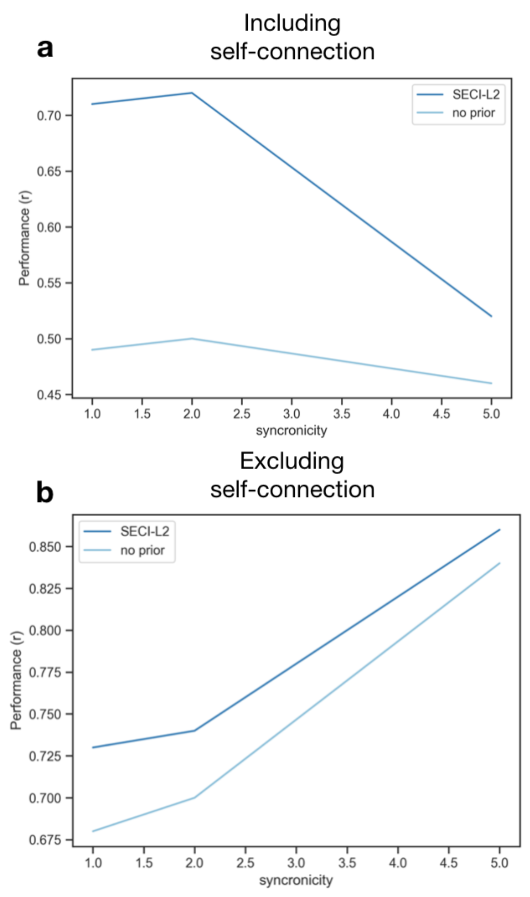

Figure 3.6: Visualization of the difference in results when considering the weights matrix including or excluding self connections. Performance is measured in Pearson r. The unit on the synchronicity axis is arbitrary and represents a scaling factor pending definition. This figure does not show a full range of synchronicity. **(a)** Performance of SECI-L2 estimated connections when including self-connections declines rapidly with increased synchronicity. **(b)** In contrast if we only consider weights which are not self-connections performance does not decline for the range shown here.

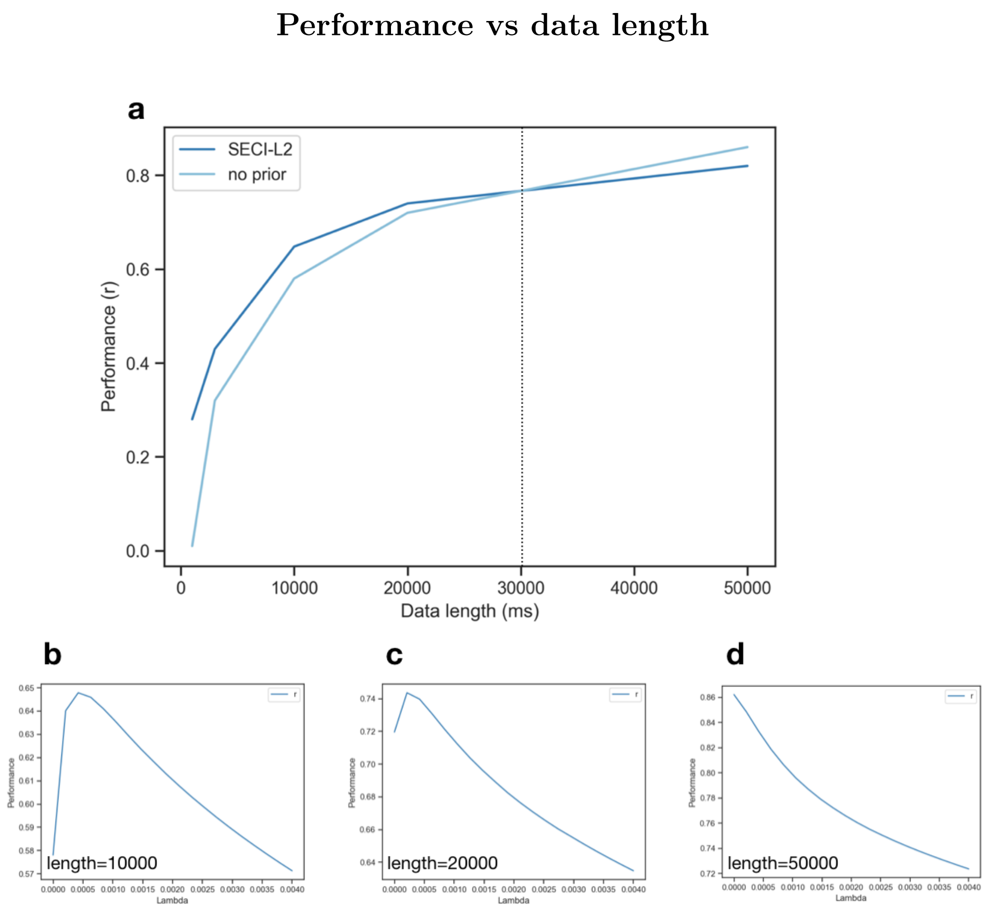

Figure 3.7: **(a)** Performance vs data length on a data set of 50 neurons estimated with SECI-L2. For high data length regularization becomes less relevant compared to non prior estimation. **(b)** Performance over range of regularization parameter λ performed on 10000ms. **(c)** Performance over range of regularization parameter λ performed on 20000ms. **(d)** Performance over range of regularization parameter λ performed on 50000ms.

3.3.4 Impact of Data Length

When evaluating how inference performance is influenced by data length we found that performance for all methods increased with data length. In Figure 3.7 we show this for a data set of 50 neurons and estimation results of SECI-L2 vs no prior condition. At large data lengths the un-regularized (no prior) condition outperforms the regularized methods. This makes sense as regularization becomes less impactful if we have large volumes of data for estimation available. The specific data length at which regularization becomes less useful depends on the overall population size as data length requirements scale with population size.

3.3.5 Impact of Population Size

Performance for higher numbers of neurons stay stable if we account for higher data length requirements. The relationship between increase in population size and required data length is linear. Increasing numbers of neurons, and thus required data length, influences computational cost. SECI-L2, due to the use of Newton's method, deals with this increase the best as it converges faster than SECI-L1 which relies on Gradient Decent. In particular, HI-SECI, which relies on sampling for the determination of the connection matrix on top of optimization of the weights, reaches impractical run times on a standard desktop machine for the estimation on set as small as 50 neurons. For SECI-L2 and SECI-L1 sets of up to 150 neurons can comfortably be calculated on a standard desktop but for the investigation of larger sets our methods are in need of implementing a parallelized version to run on a high performance computing cluster.

3.3.6 Comparison of SECI-L1, SECI-L2, HI-SECI, L1, L2 and No Prior Inference

To conclude this performance analysis we compare our three algorithms as well as L1-Norm, L2-Norm and non regularized inference. Recall that SECI-L1 and SECI-L2 reduce to L1-Norm and L2-Norm regularized inference when we set the distance matrix D to a unitary matrix (a matrix of all-ones) and to non regularized inference for $\lambda = 0$.

We analyzed the following conditions first: no prior, L1-Norm, L2-Norm, SECI-L1, SECI-L2 and provide a summary in Table 3.2. We give representative[2] results on data sets of $N = 50$. In general we find that SECI-L2 outperforms other methods by a small margin. In the representative data set highest performance as $r = 0.82$. The results reported in Table 3.2 are though not averaged. Occasional we observed performance above 0.9. The issue with reporting averages is the influence of synchronization we discussed in Section 3.3.3. Comparison over a data set with the same placement is comparable but to average over multiple data sets we first need to resolve the issue of quantifying synchronicity.

Comparison on 50 neurons on 20000 data points

Regularization	including self-connections	excluding self-connections
1. No prior	0.63	0.74
2. L1	0.66	0.78
3. SECI-L1	0.66	0.78
4. L2	0.65	0.80
5. SECI-L2	0.73	0.82

Table 3.2: Representative performance results for different inference types on 20000 data points of data for a set of 50 neurons. SECI-L2 reaches the higest performance for this data set at $r = 0.82$ if we disregard self-connections.

[2]We have shown that performance fluctuates depending on synchronicity which can make it challenging to produce averages over data sets as is is hard to produce data sets with the same synchronicity but different but statistically comparable wiring.

HI-SECI In the case of HI-SECI it is possible to combine the hierarchical prior over the existence of connections with all the previous inference methods. In combination with any of the methods shown in the above table, HI-SECI exhibits a slight improvement over the results of just using the underlying regularization for a prohibitively high computational cost. While HI-SECI is a theoretically sound extension to SECI the small performance increase over SECI-L2 doesn't seem to warrant the extensive computation cost particularly for larger number of neurons.

HI-SECI can also be used to impose just one prior instead of having two hierarchical steps if the underlying method uses no prior. In this case it provides results comparable to SECI-L2 but at higher run time.

One of the major insight we obtained in this section is the influence of self-connections on performance scores when comparing performance. Self-connections can make an otherwise very good result which correctly estimates neuron to neuron connections look worse. It is thus advisable to consider neuron to neuron connections weights separately. When one excludes self-connections from the comparison a clearer picture of the actually benefits regularization emerge. Furthermore, the influence of synchronicity while worst on self-connections still influences overall comparison and we need to work on quantifying this to make stable comparison between different data sets.

In conclusion, we find that the incorporation of distance regularization provides a performance boost in the case of SECI-L2 and HI-SECI over non and spares regularized inference results.

3.4 Application to Optical Recordings of the Mouse Brain

In this section we apply our algorithms to optical recordings from the mouse brain. We first give a description of the data set used and then apply SECI-L2, SECI-L1 and HI-SECI to a subset of this data set. The subset consists of a selection of 58 to 688 neurons from the parietal cortex while the whole data set is described below:

3.4.1 Data Set Description

The data set was collected by Dr. Aki Funamizu, a post doctoral fellow of the Neural Computation Unit and the Optical Nanoimaging Unit at OIST. Dr. Funamizu has given his permission to use his data set in our work. We refer to [16] for the original study for which this data was collected. In the following we provide a detailed description of the data set.

Site of recording. The recording site was located in layers 2, 3 and 5 of the parietal and secondary visual cortex. The parietal cortex is proposed to integrate sensory and motor input and represent ego-centric information about the position on a map.

Data acquisition. The data set comprises in-vivo 2-photon microscopy recordings from layer 2, 3 and 5 of the parietal and secondary visual cortex of 10 mice performing a auditory virtual navigation task. An adeno-associated virus was used to deliver the gene of the genetically encoded calcium indicator, GCaMP6f.

Experimental design. In the set up the mouse's head is restrained and the mouse placed on a rotatable Styrofoam ball, thus giving the impression of free movement to the mouse. In the task twelve speakers were placed around the mouse. The loudness and angle of input varied depending on the movement of the mouse on the spherical treadmill, generating an auditory virtual environment. Upon successfully approaching a target indicated by sound cues the mouse was rewarded with sugar water.

Data pre-processing. The neurons in each image were identified manually [3] and fluorescence traces were extracted then calculated as the ratio $\frac{\Delta F}{F}$. Furthermore, the distance between neurons was calculated as the euclidean distance between the center of a neuron to each other neuron in the data set. For experimental data the position of a neuron might be considered ill defined as neurons can have various shapes, sometimes spanning over an elongated area. We choose to use the cloud point center as the position of each neuron.

Data format. The data is a time series of the activity of each neuron. The sampling rate was 30.9 Hz. The field of view was 400 x 400 μm^2 recorded in a resolution of 512 x 512 pixel. This time series is what we so far denoted as the feature matrix X in the theoretical formulation. To obtain spike trains Y we utilized the OASIS software package for python. OASIS is a fast online de-convolution method for calcium data developed by Panisnski et al [13].

3.4.2 Results on Experimental Data

We first calculated the correlation between 688 neurons of one mouse as seen in Figure 3.8 graph (a). Plotting the correlation in a network graph as seen in Figure 3.8 graph (b) made it apparent that only a limited number of neurons show significant correlation. Reviewing the fluorescence traces of the 688 neurons we found only a limited number of neurons show a clear activity that seemed to be contributing to the navigation task. Thus, instead of analyzing all 688 neurons to begin with we chose to first work on a subset of 58 neurons. We selected the most responsive and least noisy neurons for this. A correlation plot (Figure 3.10 of these 58 neurons show that these were also the strongest positive correlated neurons as compared to the whole set of 688 neurons.)

The data length of the recording is 2000 points. As the recording was done at 30.9 HZ we have about 60000 ms of data length. Compared to the simulated data sets we used in previous sections this is a much larger time step. In the simulated data sets we calculated at detail levels as low as 1 ms for spike trains. Thus this experimental data set is much coarser than our simulated data sets. This is likely to influence the inference result.

[3]Please note recent advances in tracing algorithms have made hand tracing obsolete. This data set was produced before these algorithms became available.

Correlation Heatmap and Correlation-Based Network Graph of 688 Neurons

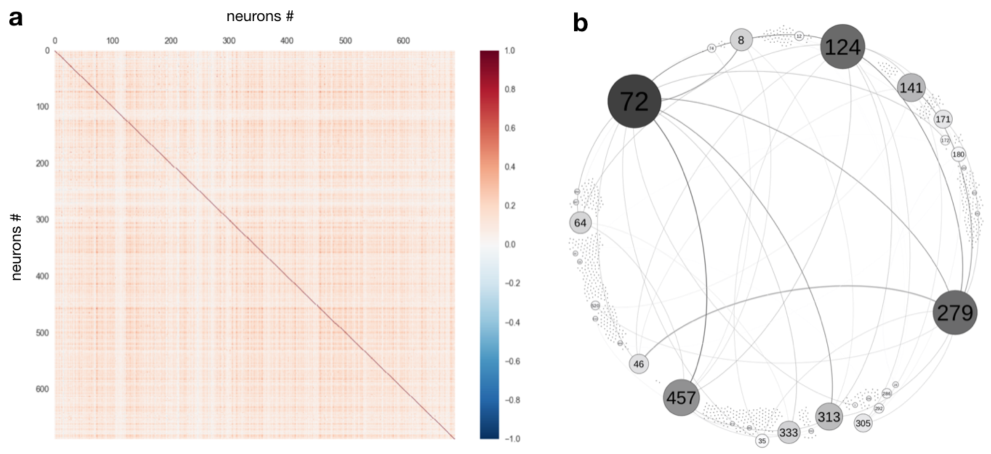

Figure 3.8: Pearson product-moment correlation coefficients calculated from experimental 2-photon microscopy recordings of 688 neurons visualized as **(a)** Heatmap of correlations and **(b)** Network graph based on correlations. The number inside each node corresponds to neuron index in graph a. Size of node and strength of edge correspond to correlation strength. This network graph illustrates that only a few neurons have a strong correlation while the majority for neurons have a correlation close to 0.

One unexpected result we found for all algorithms is strong positive self-connections as shown in the weight matrix diagonal. Self-connections in the analysis of simulated data sets were always negative and most likely corresponded to refractory period effects. Possible hypotheses for these strong positive self-connections could be influences from the reprocessing step performed using OASIS or the extraction of this subset from a larger set. It is possible that all other neurons from the set had an overall excitatory effect. The inference algorithm might thus be compensating for an unexplained overly positive excitatory input from the neurons outside the set by proxy of these high self-connections.

Estimation Results with SECI-L1 In the estimated weight matrix produced by SECI-L1 we see vertical bands in Figure 3.11 (b). From our experience with simulated data, such bands in the estimated weight matrix are an indication for an ill fitting regularization parameter λ or too short data length. We re-ran SECI-L1 on this data with a wide range of regularization parameters but always observed bands. The most likely explanation is that SECI-L1 (and generally L1-norm based regularization) needs either more data points or a higher time resolution in the recordings. The influence of time resolution to an L1 regularized method was shown by [37].

Estimation Results with SECI-L2 The estimation results obtained by SECI-L2 look plausible. They are shown in Figure 3.11 (a). SECI-L2 does not produce bands like SECI-L1 does. It is possible that SECI-L2 can perform on shorter data length than

Fluorescent traces and spike trains of 58 neurons

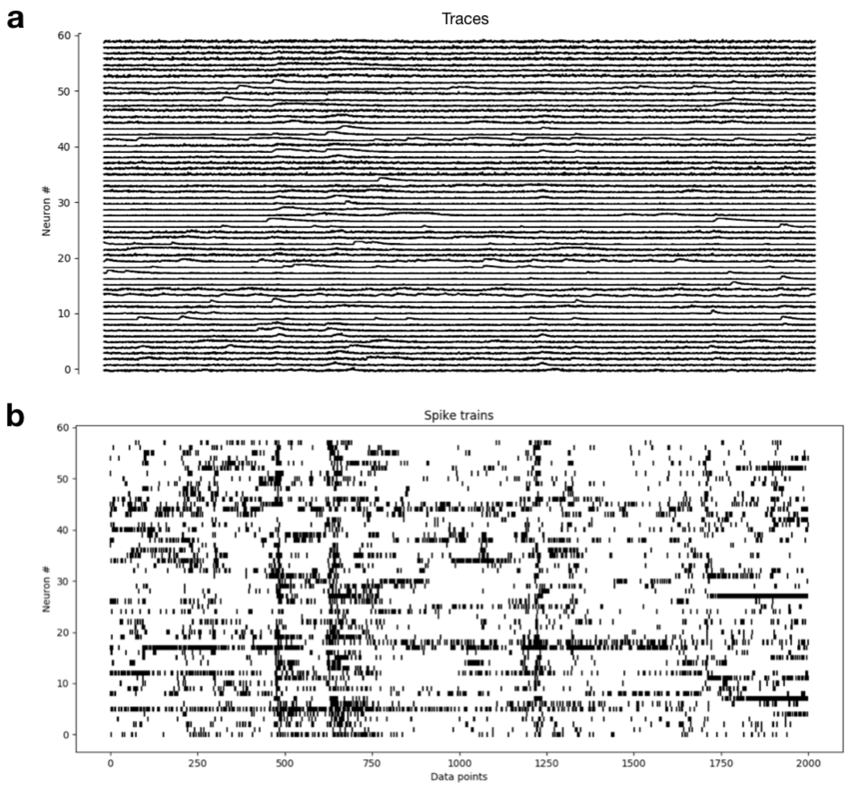

Figure 3.9: Fluorescent traces and spike trains obtained from optical 2-photon microscopy recordings of 58 neurons from the parietal and secondary visual cortex of a mouse brain while the mouse is undergoing a navigation task. One data point corresponds to one measurement at 30.9 Hz **(a)** 58 fluorescent traces ($\frac{\delta F}{F}$) and **(b)** 58 corresponding spike trains obtained by using the OASIS software package.

Correlation between 58 neurons

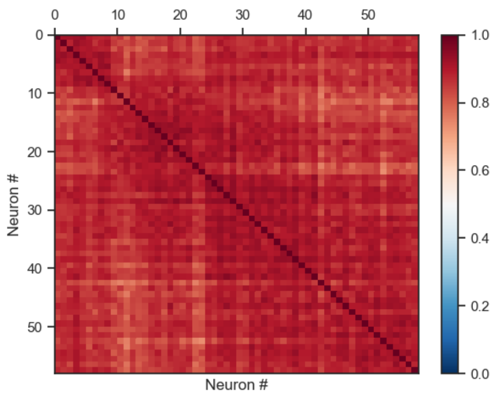

Figure 3.10: Pearson product-moment correlation coefficients of 58 most active least noisy neurons extracted from full set of 688 neurons. The correlation of these 58 neurons is over all high compared to the full set.

Estimation Results on Experimental Data for all three SECI variants

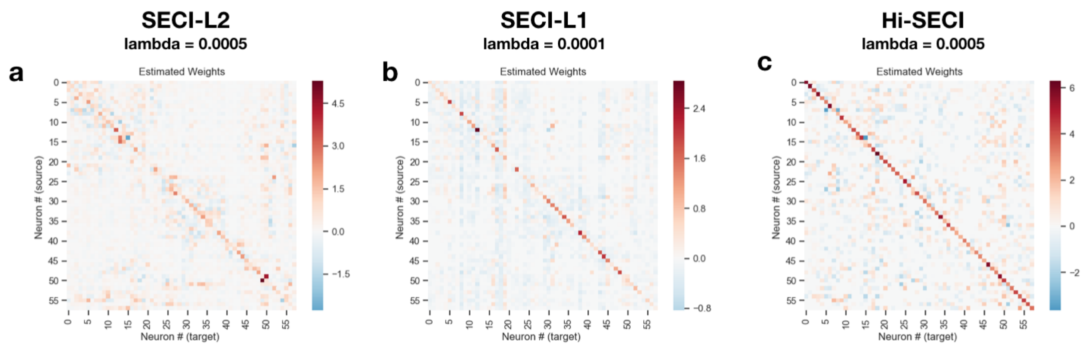

Figure 3.11: Showing the resulting estimation of weight matrix W for 58 neurons recorded at 30.9Hz with 2000 data points for all three methods. **(a)** Result weight estimation with SECI-L2 for $\lambda = 0.0005$. **(b)** Resulting weights with SECI-L1 with $\lambda = 0.0001$. **(c)** Resulting estimation weights with Hi-SECI with prior over connections and underlying run of SECI-L2.

SECI-L1. Unfortunately, we did not anticipate the data requirements for SECI-L1 and SECI-L2 would differ by a large amount so we did not investigate this condition in the simulated data section.

Estimation Results with HI-SECI Finally, we show the application of HI-SECI with underlying SECI-L2 in Figure 3.11 (c). In this method we hierarchically enforce a prior over the existence of the connections and then estimate the weights for existing connections with an underlying method. Here the underlying method used was SECI-L2.

3.5 Discussion

3.5.1 Main findings

We derived three algorithms (SECI-L1, SECI-L2 and HI-SECI) based on the assumption that distance influences the probability of neural connectivity in population of neurons up to distances of 300 μm. SECI-L1 and SECI-L2 captured distance effects by regularizing a weight matrix of connections weights while HI-SECI first imposed a prior over the existence of connections and then estimated the weight matrix. SECI-L2 shows performance improvement over previous proposed methods which only impose sparse regularization as well as SECI-L1. HI-SECI showed only small improvement over SECI-L2 for high addition computational costs.

While the incorporation of distance knowledge aids the inference performance in appropriate data sets the benefit of regularization compared to regularized inference becomes less important with increased data length.

3.5.2 Limitations

Data set variability and quantification of synchronicity Synchronicity introduced a major hurdle in analyzing performance of our methods. We discussed synchronicity as a factor that influences inference performance at length but there are still open questions. We intend to work on quantifying synchronicity (suggested measures like firing rate or correlation didn't manage to capture it). Since other studies also encountered this phenomenon [38] it seems like an important point to clarify as this seems to be a general problem for neural connectivity inference algorithms and is not just specific to our approach. Given that it is an issue not specific to our approach we are confident in the results we reported regarding SECI. The overall performance of SECI-L2 and Hi-SECI was consistently higher than other methods on each given data set. Nevertheless, we still plan to re-evaluate SECI on a data set with a grid placement to remove possible fluctuations and report averages. In retrospect it becomes apparent that the way we simulate data sets for the analysis of SECI by placing neurons using a uniform distribution and then wiring them, in the manner discussed earlier, likely contributes to fluctuations in synchronicity which then show up as a differences in performance when comparing estimation results on different sets to each other.

We will encounter a phenomenon related to synchronicity in the next chapter. In contrast to synchronicity this phenomenon can easily be quantified based on firing rate alone.

Estimation results for self-connections While we did not explicitly built self-connections into our model (In fact in the simulated data self-connections are set to be zero), we observe estimation of self-connections by SECI. These are either negative or positive. Negative self-connections are possibly indicative of a refractory period effect that is not adequately captured by the LNP model as it has no explicit refractory period parameter. Positive self-connections, as observed in the application to experimental data of the PPC might have several explanations such as 1) long lag auto-correlations [4, 39] or 2) tuning to slowly varying behavioural correlates. This phenomena requires further investigation and possibly an extension of the model to account for such effects.

Hyper-parameter optimization for experimental data application As we had only a short data set available we did not go into much detail beyond a proof of concept that the algorithms are ready to be applied to experimental data and yield plausible results. Nevertheless, we would like to point a potential not well optimized hyper parameter lambda in Figure 3.11 in the case of SECI-L2 as the distance influence is rather strong measured by the difference of the estimated weight matrix to the prior distance information. Finding good hyper parameters is though a general challenge of such methods and not specific to our method. Future applications to experimental data sets are thus in need of better hyper parameter optimization.

Data set requirements for experimental data application We had selected the experimental data set used in this chapter before the completion of the algorithms' analysis. It became apparent after the fact that we required longer data length at the data set's sampling rate. The number of data points was only 2000 points at a sampling rate of 30.9 Hz for the experimental data set used in this chapter. For future application we need to secure longer recordings at around 20000 data points for 50 neurons as illustrated in section 3.3.4. This requirement scales linearly with the number of neurons in the population but this data length requirement is estimated based on simulated data. For experimental recordings extra data is desirable for example for optimizing hyper-parameters as well as validating the estimated model against further data not used in the optimization process.

Pre-processing steps In the application of SECI to experimental calcium recordings we relay on an external software module to estimate spike trains. The actual influence of such a pre-processing step needs to be analyzed as it might introduce undesirable effects. Furthermore, in the analysis on simulated data we simulate a trace like explanatory variable. The extend to wish this simulation is realistic and the detail at which it is performed might also influence estimation results.

3.5.3 Future Work

The most apparent points identified for future work are listed below:

1. Investigating of the influence of external neurons/refractory period on self connection proxy effect:

   We observed the use of the self-connections as a type of proxy for other quantities such as possibly refractory period or external input or unobserved neurons. We suggest investigating this influence and then constructing a model that explicitly estimates such quantities.

2. Extension to a model with explicit estimation of refractory period and external inputs.

   This directly results as a consequence of the previous point.

3. Combination of SECI with other algorithms:

   As we have repeatedly pointed out SECI is a localized estimation algorithm for which we only have support on areas up to 300 μm^2. A combination with other methods which can work on a larger areas but use SECI as a support for its local estimation is another direction we consider.

3.5.4 Conclusion

In this chapter we have introduced the Spatial Effective Connectivity Inference (SECI) approach with three specific sub versions of the main idea. SECI-L1 which combines spatial regularization with L1 norm. SECI-L2 which combines spatial regularization with L2 norm. HI-SECI which applies the same idea of using distance based prior but in a full hierarchical MAP approach which combines a prior over connections and a regularization of weights. We have analyzed SECI on simulated data and observed a performance increase for SECI-L2 and HI-SECI over non prior and L1 and L2 regularization methods. We illustrated the application of SECI-L1, SECI-L2 and HI-SECI to an experimental data set.

Chapter 4

Modular Effective Connectivity Inference (MECI)

4.1 Introduction

As discussed in Chapter 2, previously published Bayesian approaches to neural connectivity inference have been shown to benefit from priors like sparseness of connections [38] or smoothness of temporal responses [59]. In Chapter 3 we described an investigation of spatial information, specifically distance between neurons, as another valid prior for this type of connectivity inference.

In addition to such topographical organization, modular organization, like cortical columns are commonly observed in the brain.

In this chapter the focus is on the development of an effective neural connectivity inference algorithm with a modularity based prior which can, in principle, capture different types of modular organization. For a plausible starting point to develop such a method we consider the assumption that synaptic connectivity within modules might be significantly stronger than connections between modules. Such possible differences could be expressed in density of connectivity or connection strength. This is just one possible modularity assumption we choose as a starting point to investigate here. Extending this idea to a more general formulation that allows different assumption over the distribution of connections within and between modules are briefly discussed in the discussion section.

In the remainder of this chapter we first give a theoretical formulation of how to incorporate a modularity prior by assuming two different Gaussian distributions for within and between module connection strength. Then we derive a modularity-based effective neural connectivity inference algorithm (MECI) which utilizes Gibbs sampling and Newton's Method to estimate module membership and connections weights in section 4.2.2. We proceed on to an in depth evaluation of MECI on a simulated data set. To give an idea of how MECI behaves on a bio-physiological realistic data we apply it to data sets of the model of the Basal Ganglia in 4.4. We then compare MECI to SECI on data set which exhibits modules based as a side effect of neural placement and distance based wiring. Finally, we have a discuss about our findings including limitations and a list of possible extensions of the algorithm to highlight future directions.

4.2 Methods

4.2.1 Generative Models of Modular Network and Neural Spikes

We first formulate a generative model of neural networks with modular organization. For a number of neurons N, we assume the number of modules C. Each neuron is assigned to a specific module according to a module assignment matrix $Z \equiv [z_{ci}] \in \{0, 1\}^{C\times N}$, where z_{ci} is an indicator variable such that $z_{ci} = 1$ if the ith neuron belongs to the cth cluster and $z_{ci} = 0$ otherwise. $z_i = [z_{1i}, \cdots, z_{Ci}]^\top$ is an indicator vector representing the cluster assignment of the ith neurons. Z can be written as $Z = [z_1, \cdots, z_N]$.

We define a simple non-informative prior module assignment as

$$P(Z) = \prod_{i=1}^{N} P(z_i), \text{ where } P(z_{ci} = 1) = 1/C. \tag{4.1}$$

Accordingly,

$$\ln P(Z) = -NC \ln C \tag{4.2}$$

Note that it is constant and independent of Z.

We then assume a prior distribution of the connection weights depending on the module assignment:

$$P(w_{ij}|z_i, z_j) = \left[\frac{1}{\sqrt{2\pi\sigma_w^2}} \exp\left\{-\frac{w_{ij}^2}{2\sigma_w^2}\right\}\right]^{z_i^\top z_j} \left[\frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left\{-\frac{w_{ij}^2}{2\sigma_b^2}\right\}\right]^{1-z_i^\top z_j} \tag{4.3}$$

This expression implies that weights are distributed according to two Gaussians: $N_w(0, {\sigma_w}^2)$ for with-in module weights and $N_b(0, {\sigma_b}^2)$ for between module weights.

Accordingly, the prior of the connection weight matrix W under a given module configuration Z is

$$P(W|Z) = \prod_{i=1}^{N}\prod_{j=1}^{N} P(w_{ij}|z_i, z_j). \tag{4.4}$$

Then the log prior is represented as

$$\ln P(W|Z) = \sum_{i=1}^{N}\sum_{j=1}^{N} z_i^\top z_j \left[-\frac{\lambda_w}{2} w_{ij}^2 + \frac{1}{2}\ln\lambda_w - \frac{1}{2}\ln(2\pi)\right] \tag{4.5}$$

$$+ \sum_{i=1}^{N}\sum_{j=1}^{N} (1 - z_i^\top z_j) \left[-\frac{\lambda_b}{2} w_{ij}^2 + \frac{1}{2}\ln\lambda_b - \frac{1}{2}\ln(2\pi)\right] \tag{4.6}$$

where we use inverse variance notation $\lambda_w = \frac{1}{{\sigma_w}^2}$ and $\lambda_b = \frac{1}{{\sigma_b}^2}$.

Taking the logarithm of the probability simplifies this to the sum of the logarithms of the individual components. This will later simplify the derivation.

Spike generative model We use a linear-nonlinear Poisson (LNP) cascade model [9, 54, 55] as a model of neural spike responses. This is defined fully in Chapter 2.

The spike data matrix is given as $Y \equiv [y_{ti}] \in \{0,1\}^{T\times N}$ with T the number of time bins with the bin size Δt and

y_{ti} the number of spikes of i-th neuron in the t-th time bin. $y_t = [y_{t1}, \cdots, y_{tN}]$ is a vector representing the numbers of spikes detected at the t-th time bin. Y can be written as $Y = [y_1^\top, \cdots, y_T^\top]^\top$.

$X \equiv [x_{ti}] \in \mathbb{R}^{T\times N}$ is the feature matrix extracted from the history of neural activity before the tth time bin.

We estimate the connectivity weight matrix $W \equiv [w_{ij}] \in \mathbb{R}^{N\times N}$ with w_{ij} indicating the connectivity weight from the jth to the ith neurons. We denote by $w_i = [w_{i1}, \ldots, w_{iN}]^\top$ the connectivity weight vector projecting to the ith neuron.

The ***likelihood***, based on the definition of the linear-nonlinear Poisson cascade model introduced in Chapter 2, becomes:

$$P(Y|W) = \prod_{t=1}^{T}\prod_{i=1}^{N} P(y_{ti}; w_i) \tag{4.7}$$

where

$$P(y_{ti}; w_i) = \frac{(r_{ti}\cdot\Delta t)^{y_{ti}}}{y_{ti}!}\exp(-r_{ti}\cdot\Delta t), \tag{4.8}$$

with the firing rate defined as:

$$r_{ti} = \exp\left\{\sum_{j=1}^{N} w_{ij}x_{jt}\right\} = \exp\left\{w_i^\top x_t^\top\right\}. \tag{4.9}$$

4.2.2 Inference of Module Membership and Connection Weights

We now turn towards the formulation of a Bayesian inference approach for the module membership and connection weights based on different regularization parameters depending on the module membership.

The ***log-likelihood*** of the membership Z and the connection weights W given the spike train Y is

$$\begin{aligned}\ln P(Y|W) &= \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti}\left\{\ln r_{ti} + \ln\Delta t\right\} - \Delta t\cdot r_{ti} - \ln y_{ti}!\right] && (4.10)\\ &= \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti}\left\{w_i^\top x_t^\top + \ln\Delta t\right\} - \Delta t\exp\left(w_i^\top x_t^\top\right) - \ln y_{ti}!\right] && (4.11)\end{aligned}$$

The ***joint distribution of all stochastic variables*** takes the form

$$P(Y,Z,W) = P(Z)P(W|Z)P(Y|W). \tag{4.12}$$

Taking the logarithm we obtain,

$$\begin{aligned}
\ln P(Y,Z,W) &= \ln P(Z) + \ln P(W|Z) + \ln P(Y|W) && (4.13)\\
&= -CN \ln C && (4.14)\\
&+ \sum_{i=1}^{N}\sum_{j=1}^{N} z_i^\top z_j \left[-\frac{\lambda_w}{2} w_{ij}^2 + \frac{1}{2}\ln\lambda_w - \frac{1}{2}\ln(2\pi)\right] && (4.15)\\
&+ \sum_{i=1}^{N}\sum_{j=1}^{N} (1 - z_i^\top z_j) \left[-\frac{\lambda_b}{2} w_{ij}^2 + \frac{1}{2}\ln\lambda_b - \frac{1}{2}\ln(2\pi)\right] && (4.16)\\
&+ \sum_{t=1}^{T}\sum_{i=1}^{N} \left[y_{ti}\left\{w_i^\top x_t^\top + \ln\Delta t\right\} - \Delta t \exp\left(w_i^\top x_t^\top\right) - \ln\Gamma(y_{ti}+1)\right] && (4.17)
\end{aligned}$$

Inference of weights given module assignment

Conditional distribution **of w_i for each $i = 1, \cdots, N$** is given as

$$P(w_i|Y,Z,W_{-i}) = \frac{P(Y,Z,W)}{\int P(Y,Z,W)dw_i} \tag{4.18}$$

Thus the ***log posterior probability*** of w_i given module configuration Z is

$$\begin{aligned}
\ln P(w_i|Y,Z,W_{-i}) &= \ln P(Y,Z,W) + (w_i\text{-independent}) && (4.19)\\
&= -\sum_{i=1}^{N}\sum_{j=1}^{N} z_i^\top z_j \frac{\lambda_w}{2} w_{ij}^2 - \sum_{i=1}^{N}\sum_{j=1}^{N}(1 - z_i^\top z_j)\frac{\lambda_b}{2} w_{ij}^2 && (4.20)\\
&+ \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti} w_i^\top x_t^\top - \Delta t \exp\left(w_i^\top x_t^\top\right)\right] && (4.21)\\
&+ (w_i\text{-independent}) && (4.22)\\
&= -\frac{\lambda_b}{2}\sum_{j=1}^{N} w_{ij}^2 - \left(\frac{\lambda_w - \lambda_b}{2}\right) z_i^\top \sum_{j=1}^{N} z_j w_{ij}^2 && (4.23)\\
&+ \sum_{t=1}^{T}\sum_{i=1}^{N}\left[y_{ti} w_i^\top x_t^\top - \Delta t \exp\left(w_i^\top x_t^\top\right)\right] && (4.24)\\
&+ (w_i\text{-independent}) && (4.25)
\end{aligned}$$

Now we can derive the ***gradient*** (first order derivative) with respect to w_i:

$$\mathbf{g}_i \equiv \nabla_{w_i} \ln P(w_i|Y, Z, W_{-i}) \tag{4.26}$$

$$= -\lambda_b w_i - (\lambda_w - \lambda_b)\left(Z^\top z_i \circ w_i\right) \tag{4.27}$$

$$+ \sum_{t=1}^{T} x_t^\top y_{ti} - \Delta t \sum_{t=1}^{T} x_t^\top \exp\left(w_i^\top x_t^\top\right) \tag{4.28}$$

$$= -\lambda_b w_i - (\lambda_w - \lambda_b)\left(Z^\top z_i \circ w_i\right) + \sum_{t=1}^{T} \left(y_{ti} - r_{ti}\Delta t\right) x_t^\top, \tag{4.29}$$

where $\circ$ denotes the Hadamard product, i.e. element-wise product. The ***Hessian*** (second order derivative) is given by

$$\mathbf{H}_i \equiv \nabla_{w_i}^2 \ln P(w_i|Y, Z, W_{-i}) \tag{4.30}$$

$$= -\lambda_b I - \operatorname{diag}\left[(\lambda_w - \lambda_b)\, Z^\top z_i\right] - \sum_{t=1}^{T} (r_{ti}\Delta t) x_t^\top x_t. \tag{4.31}$$

Having access to the Hessian allows us to use Newton's method for optimization when constructing our algorithm in the next section. Furthermore, if we set the maximum point of $\ln P(w_i|Y, Z, W_{-i})$ as w_i^*, the following approximation becomes available:

$$\ln P(w_i|Y, Z, W_{-i}) \approx -\frac{1}{2}\left(w_i - w_i^*\right)^\top \left(-H_i^*\right)\left(w_i - w_i^*\right) \tag{4.32}$$

Thus,

$$P(w_i|Y, Z, W_{-i}) \approx \mathcal{N}(w_i; w_i^*, -[H_i^*]^{-1}), \tag{4.33}$$

where

$$w_i^* = \underset{w_i}{\operatorname{argmax}} \left[\ln P(w_i|Y, Z, W_{-i})\right]. \tag{4.34}$$

$$H_i^* = -\lambda_b I - \operatorname{diag}\left[(\lambda_w - \lambda_b)\, Z^\top z_i\right] - \sum_{t=1}^{T} (r_{ti}^*\Delta t) x_t^\top x_t. \tag{4.35}$$

$$r_{ti}^* = \exp\left\{(w_i^*)^\top x_t^\top\right\} \tag{4.36}$$

This approximation is valid in the sense of Laplace approximation method.

Computing the weight matrix W is an optimization problem of N^2 variables. The equation is concave in w and can be separated into N terms to be optimized separately by optimization methods like Gradient Decent or Newton's method etc.

Inference of module assignment Next we consider the inference for the module assignment matrix Z. The **conditional distribution** of z_i for each $i = 1, \cdots, N$ is given by

$$P(z_i|Y, Z_{-i}, W) = \frac{P(Y, Z, W)}{\sum_{z_i} P(Y, Z, W)} \tag{4.37}$$

Thus the **log conditional probability** of z_i becomes,

$$\begin{aligned}\ln P(z_i|Y, Z_{-i}, W) &= \ln P(Y, Z, W) + (z_i\text{-independent}) &(4.38)\\ &= \sum_{i'=1}^{N}\sum_{j'=1}^{N} z_{i'}^{\top} z_{j'} \left[-\frac{\lambda_w}{2} w_{i'j'}^2 + \frac{1}{2}\ln\lambda_w\right] &(4.39)\\ &+ \sum_{i'=1}^{N}\sum_{j'=1}^{N}(1 - z_{i'}^{\top} z_{j'}) \left[-\frac{\lambda_b}{2} w_{i'j'}^2 + \frac{1}{2}\ln\lambda_b\right] &(4.40)\\ &+ (z_i\text{-independent}) &(4.41)\end{aligned}$$

So the conditional probability of assigning neuron i to module k given explanatory variable Y, current module assignment Z_{-i} of other neurons and current weight matrix W is:

$$P(z_{ik} = 1|Y, Z_{-i}, W) = \frac{\phi(z_{ik} = 1|Y, Z_{-i}, W))}{\sum_{k'=1}^{C} \phi(z_{ik'} = 1|Y, Z_{-i}, W))}, \quad (4.42)$$

where

$$\begin{aligned}\ln \phi(z_{ik} = 1|Y, Z_{-j}, W)) &= \sum_{i'=1}^{N}\sum_{j'=1}^{N} z_{i'}^{\top} z_{j'} \left[-\frac{\lambda_w}{2} w_{i'j'}^2 + \frac{1}{2}\ln\lambda_w\right] &(4.43)\\ &+ \sum_{i'=1}^{N}\sum_{j'=1}^{N}(1 - z_{i'}^{\top} z_{j'}) \left[-\frac{\lambda_b}{2} w_{i'j'}^2 + \frac{1}{2}\ln\lambda_b\right], &(4.44)\end{aligned}$$

and $z_{ij} = \delta_{jk}$. With δ being the Kroeneker's delta which is defined as $\delta_{ij} = 1$ if $i = j$ and 0 otherwise.

4.2.3 Formulation of the Algorithm (MECI)

We are now able to formulate the algorithm for modular effective connectivity inference (MECI). We present pseudo-code in this section to discuss the main algorithm.

The algorithm and all simulations were implemented in Python3.5 and all source code and data sets used to produce graphs in this thesis will be made publicly available at `https://github.com/oist/pynci/MECI` upon peer reviewed publication of this work.

Implementation

In Algorithm 2 we construct a Markov chain Monte Carlo (MCMC) algorithm for modular effective connectivity inference (MECI). MECI is devided into two main blocks: The weight estimation block [line 3 - 10] in which we use Newton's method for optimizing weights based on the current module assignment and the module assignment block [line 11 - 17] in which neurons are re-assigned to the most likely module, by updating module assignment matrix Z taking into consideration the best weights found in the last run of the weight estimation block. The two blocks are wrapped by an outer

Algorithm 6 MECI

```
Input: Y, X, C, $\sigma_b$, $\sigma_w$
Output: Estimated weight matrix W, module assignment matrix Z
1: Initalize $W$ and $Z$ appropriately.
2: for $h = 1, \cdots, H_{max}$ do
3:     for $i = 1, \cdots, N$ do
4:         repeat
5:             $r_{it} \leftarrow \exp\left[w_i^\top Z x_t\right] \quad (t = 1, \cdots, T)$
6:             $g \leftarrow -\lambda_b w_i - (\lambda_w - \lambda_b)(Z^\top z_i \circ w_i) + \sum_{t=1}^{T} (y_{ti} - r_{ti}\Delta t)\, x_t^\top,$
7:             $H \leftarrow -\lambda_b I - \mathrm{diag}\left[(\lambda_w - \lambda_b) Z^\top z_i\right] - \sum_{t=1}^{T} (r_{ti}\Delta t) x_t^\top x_t.$
8:             $w_i \leftarrow H^{-1} g$
9:         until $w_i$ converged
10:        Sample $w_i$ according to the Gaussian with mean $w_i$ and covariance $H^{-1}$.
11:    for $j = 1, \cdots, N$ do
12:        for $k = 1, \cdots, C$ do
13:            $z_{k'j} \leftarrow (\delta_{1,k}, \cdots, \delta_{C,k})^\top$
14:            $\phi_{kj} \leftarrow \sum_{i'=1}^{N} \sum_{j'=1}^{N} z_{i'}^\top z_{j'} \left[-\frac{\lambda_w}{2} w_{i'j'}^2 + \frac{1}{2}\ln\lambda_w\right] + \sum_{i'=1}^{N} \sum_{j'=1}^{N} (1 - z_{i'}^\top z_{j'}) \left[-\frac{\lambda_b}{2} w_{i'j'}^2 + \frac{1}{2}\ln\lambda_b\right],$
15:            $q_{kj} \leftarrow \frac{\exp[\phi_{kj}]}{\sum_{k'=1}^{C} \exp[\phi_{kj}]}$
16:            Sample $k$ according to the categorical distribution: $\mathrm{Cat}(q_{1j}, \cdots, q_{Kj})$.
17:            $z_j \leftarrow (\delta_{1,k}, \cdots, \delta_{C,k})^\top$
18:    if converged then break
```

sampling loop [line 2]. We refer to this outer loop when we speak about MECI taking a specific number of iterations to converge. The number of iteration of this loop can sometimes seem low because we also have inner loops. The outer sampling loop does not have to fully run until it reaches H_{max}. If convergence is achieved we can break out of the outer sampling loop. But as briefly discussed in Chapter 2 convergence for MCMC methods is not provable. We can only make reasonable assumptions.

The structure of this algorithm is reminiscent of an Expectation Maximization (EM) algorithm. It is through a Gibbs sampling approach (See definition in Chapter 2). There exists a relationship between Gibbs sampling and generalized EM [11].

Imagine that X, Y and θ are a hidden variable, an observation, and a model parameter, respectively. The EM algorithm consists of two steps. The E-step defined the so-called expected log-likelihood function as $Q(\theta'|\theta) = E[\ln P(X, Y|\theta')|Y, \theta]$, then the M-step updates θ by $\text{argmax}_{\theta'} Q(\theta'|\theta)$. With repeated iterations this converges to a maximum likelihood estimate θ^*. This can be transferred into a formulation of Gibbs sampling. Instead of maximizing at each of these two steps, we use the conditional distribution and sample from it. You can see this in both the weight blocks and the module block: In line 10 we sample for the weight and in line 16 we sample for the module assignment.

We iteratively update W as well as Z. In the weight block W is optimized by Newton's method. The model parameters are updated in the direction of Hessian error and gradient error. Then we re-evaluate and repeat until convergence for this particular assignment has been achieved. Initial weights can be set to zero or an initial estimation taking module assignment into account. We chose to set a random normal assignments for Z and then initialize W according to it. In the module assignment block we reassign module membership based on the conditional probability $q_{kj} \equiv p(z_{kj} = 1|Y, Z_{-j}, W)$. In this sampling step we sample from the categorical distribution based on q_{kj}. This iterative manner produces a chain of samples based on the conditional distributions which on the long run approximates the joint distribution of W and Z.

4.3 Performance Evaluation on Simulated Data

We know turn to an in depth analysis of the algorithm's performance and behavior on simulated data, starting with a purely theoretical data set constructed to match the algorithms assumptions.

This chapter uses only simulated data sets for which all model parameters are known and controlled so we have a ground truth to compare our algorithm's estimations against.

We now discuss how the data set is constructed, then evaluate performance for several different cases and sub sets, and discuss the results.

4.3.1 Data Simulation

For the use in evaluation we are first constructing a data set with purely theoretical features which we will refer to as Faithful. Faithful directly match the algorithm's assumption as detailed below. This simulation setup does not follow realistic bio-physiological parameters. We will consider a more bio-physiological informed data sets in section 4.4. The reason for constructing such a benchmark set is to identify performance baselines, constraints and best cases. A data set that is constructed to match the the algorithms assumptions faithfully is the easiest case to apply the algorithm to. Any performance constraints we find should therefore translate directly to more difficult inference cases on data sets which don't directly match this assumption as a type of lower bound.

The purpose of this simulation is not to produce a bio-physiologically realistic neural activation model but to show how the algorithm behaves on a simple data sets and evaluate factors like the module size or number which might influence the estimation algorithm.

Faithful is a collection of many subsets all true to the following simulation setup and only differ by the initialization of the number of neurons N, Module number C and parameters σ_w and σ_b. These data set reflects the algorithms assumption that neurons either belong to a module or not and the connection strength within modules and between them is sampled from two different normal distributions $N(0, \sigma_w)$ and $N(0, \sigma_b)$. This gives us neural populations which are wired in the following manner.

Weight matrix W takes:

$$W = \begin{cases} w_{ij} \sim N(0, \sigma_w^2), & \text{if } z_i = z_j \\ w_{ij} \sim N(0, \sigma_b^2), & \text{otherwise} \end{cases}$$

Furthermore, self-connections are set to 0 so neurons do not influence themselves but a hard refractory period is imposed during which neurons cannot produce a spike for a fixed time period of 4 ms after last having done so.

We are using the Linear-nonlinear Poisson (LNP) cascade model [9, 54, 55] as introduced in Chapter 2 and use specific model parameters as stated in Table 4.1 while only adjusting neuron number N, σ_w^2, σ_b^2 and module assignment matrix Z.

Parameter	Symbol	Value
Spike train parameters		
Calculation step size for spikes	Δ_s	1 ms
Maximum probability	p_{max}	0.23
Maximum weight	w_{max}	3.0
Decay parameter for spikes	τ_s	5 ms
Basal rate for spikes	b_s	5.0
Absolute refractory period	ref	4 ms
Trace parameters		
Calculation step size for traces	Δx	30 ms
Noise variance	σ_{noise}	0.09
Decay over time	τ_x	2.0
Fixed Jump height	a_x	1.0
Basal activation	b_x	1.0

Table 4.1: Summary of essential model parameters (MECI).

4.3.2 Impact of Bursting and Sparse Activity

In this first performance evaluation case we present two conditions under which MECI cannot perform at optimally. As a performance measure adjusted Rand index (RI) [24] for module assignment and Pearson correlation coefficient (r) for weight estimation of weights between neurons are reported. It should be noted that both measures are measured on a 0.0 - 1.0 scale but the Rand index is quite sensitive to even minor changes. For example a drop form 1.0 to 0.8 might just signal one mismatch. Therefore a result of 0.8 is still considered a very good result. Specifically, these two conditions are types of anomalous neural activity in spike trains Y: 1) Unrealistically strong bursting activity and 2) Very sparse activity. We illustrate this behavior in Figure 4.1 and 4.2. Figure 4.1 shows good estimation performance on a small data set of 30 neurons. In comparison Figure 4.2 illustrates the same network but with an increased firing rate obtained by increasing the connection strength in the network. This increased firing rate leads to bursting behavior. Such bursting behavior influences estimation performance. We can observe rapid deterioration of performance for certain firing ranges. In Figure 4.2 plot (g) shows a regression plot which illustrates performance deterioration down to r = 0.5. For the same data set but without bursting behavior good performance at r = 0.8 as observed.

We quantify this by classifying what firing rate ranges are not acceptable for MECI to run on. For data sets which fall into these categories performance deteriorates gradually with increased bursting on the high firing rate end and sparse activity on the low firing rate end respectively. In Figure 4.3 we plot this relationship of performance vs firing rate.

In the case of bursting behavior in a data set the average firing rate over the entire population can seem low (eg. below 60 Hz[1]) so we plot the average firing rate of the

[1]The reason for this seemingly low whole population average might be that unrealistically high

Performance on a normal data example

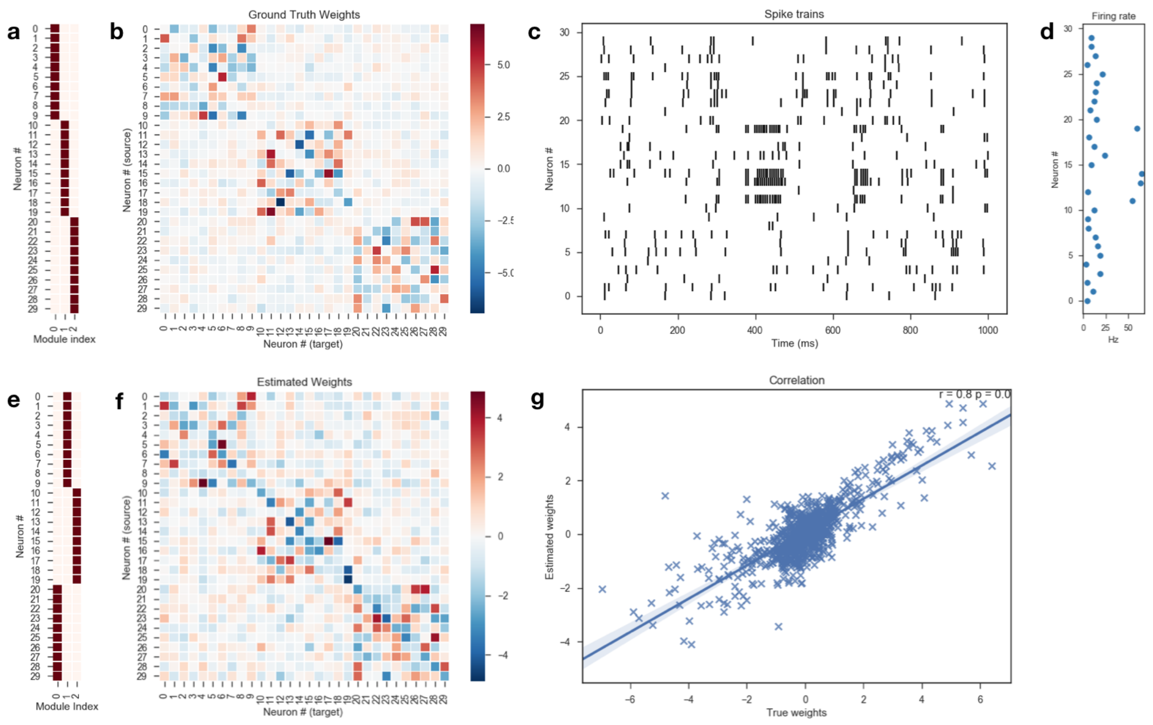

Figure 4.1: An example of a data set with 30 neurons 3 modules on which MECI performs well. (**a**) Ground truth module assignment. (**b**) Ground truth connection weights between neurons. (**c**) The first 1000 ms of spike trains produced by using the connection weights shown in b. (**d**) Average firing rate. (**e**) MECI's estimated module assignment. Note that module indices are interchangeable, thus a and e refer to the same module assignment and this assignment has an adjusted Rand index of 1.0 (highest possible score). (**f**) MECI's estimated connection weights between neurons. (**g**) Regression plot with Pearson correlation coefficient r and p-value of neuron to neuron connection weights comparing ground truth to estimated weights.

Performance on an anomalous data example with high bursting

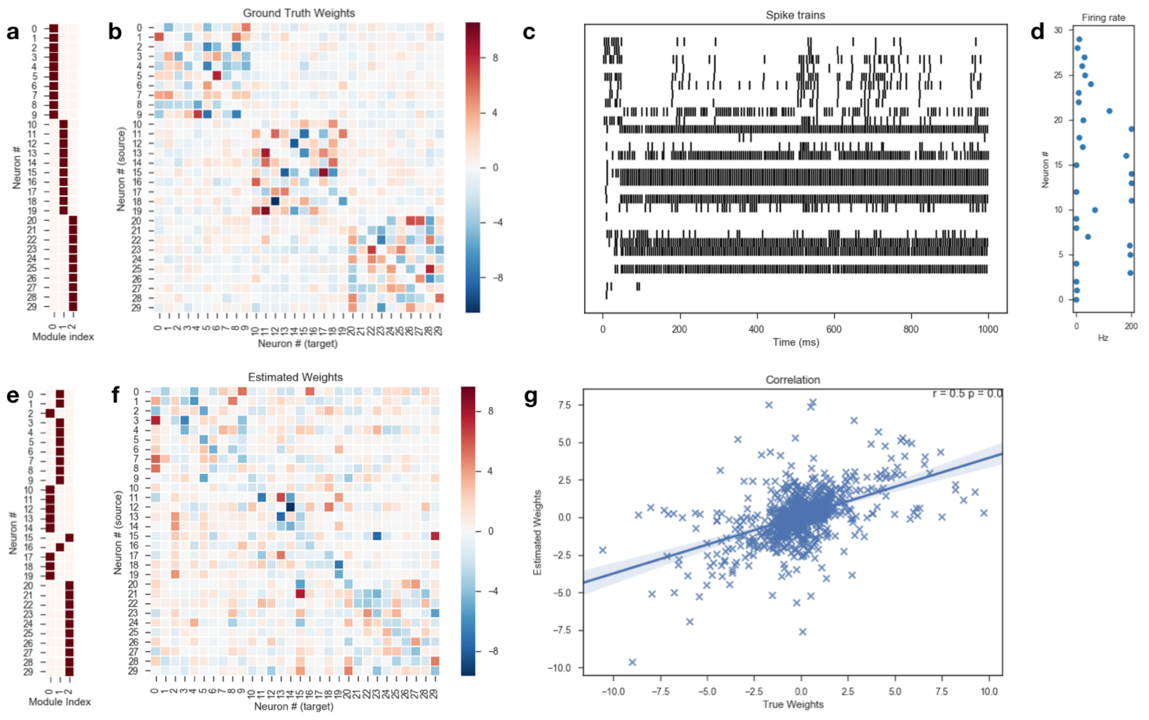

Figure 4.2: Example of performance on a data set of 30 neurons with 3 modules exhibiting high bursting. The bursting behavior influences estimation accuracy. Adjusted Rand score is still high at around 0.8 but the weight estimation deteriorated for to r = 0.5.

20% most active neurons in Figure 4.3 plot a and c. In the case of bursting a few neurons in a module showing firing rates over 140 Hz is enough to disrupt the entire estimation. Furthermore, firing rates above 200 Hz were not analyzed as such high numbers lead to singular matrices during calculations[2] so no unique solution can be found by with the linear solver we are using. As a solver the *numpy.linalg.solve* [44] package was used.

In the case of very low activity as shown in Figure 4.3 b we also observe performance deterioration. Particularly, in the presence of almost silent neurons.

Based on our findings we recommend a safe firing range for MECI to operate on: An average firing rate above 9Hz seems sufficient to make sure the data is not too sparse. In the case of bursting behavior no neuron should fire above 140 Hz. Even a small number of such neurons is detrimental to estimation performance.

While establishing this range we frequently considered correlation plots of spike trains and traces and found that while MECI's performance deteriorates on sparse data it is more robust to sparse activity than correlation analysis. For some data for which a correlation analysis is not able to reason about which neurons influence each other can still be analyses with MECI.

4.3.3 Data Length Requirements and Impact of Population Size

Data length required by MECI appears to linearly increase with the number of neurons. We illustrate this for two data sets of n = 40 and n = 80 in Figure 4.4 plot a and b respectively.

It should be noted that there exists a point at which data length overpowers the usefulness of module assignment. It is possible for exceedingly long data length to lead to good weight estimation without finding a good module assignment. This makes sense as the weight estimation loop starts with a given module assignment and optimizes from there. If the estimation problem of finding weights is made easy by the availability of large amounts of data the weight optimization loop might settle on well performing weight estimates without a good module assignment.

4.3.4 Robustness to Module Number and Size Variability

Initially we believed that performances might be influenced by modules size or variability of size in a population. After adjusting for sufficient data length we found that MECI is robust to these changes. In Figure 4.5 we show performance on a data set which has two modules. We gradually increase the module size of the second module up to four times the original size. Performance stays roughly stable. Furthermore, we investigated the impact of increasing the total number of modules in a set. The investigation of this property was made difficult by some underlying properties of the Faithful data set: With increased number of modules it becomes more likely to produce

bursting in a small number of neurons can suppress the overall population activity for example when the bursting neuron is inhibitory.

[2]This is most likely due to the total suppression of some neurons by a bursting inhibitor neuron thus leading to some neurons being completely silenced. No all zero components like this is a requirement of the linear solver package we use so the only way to circumvent this is to remove silent neurons.

Performance decline on anomalous data

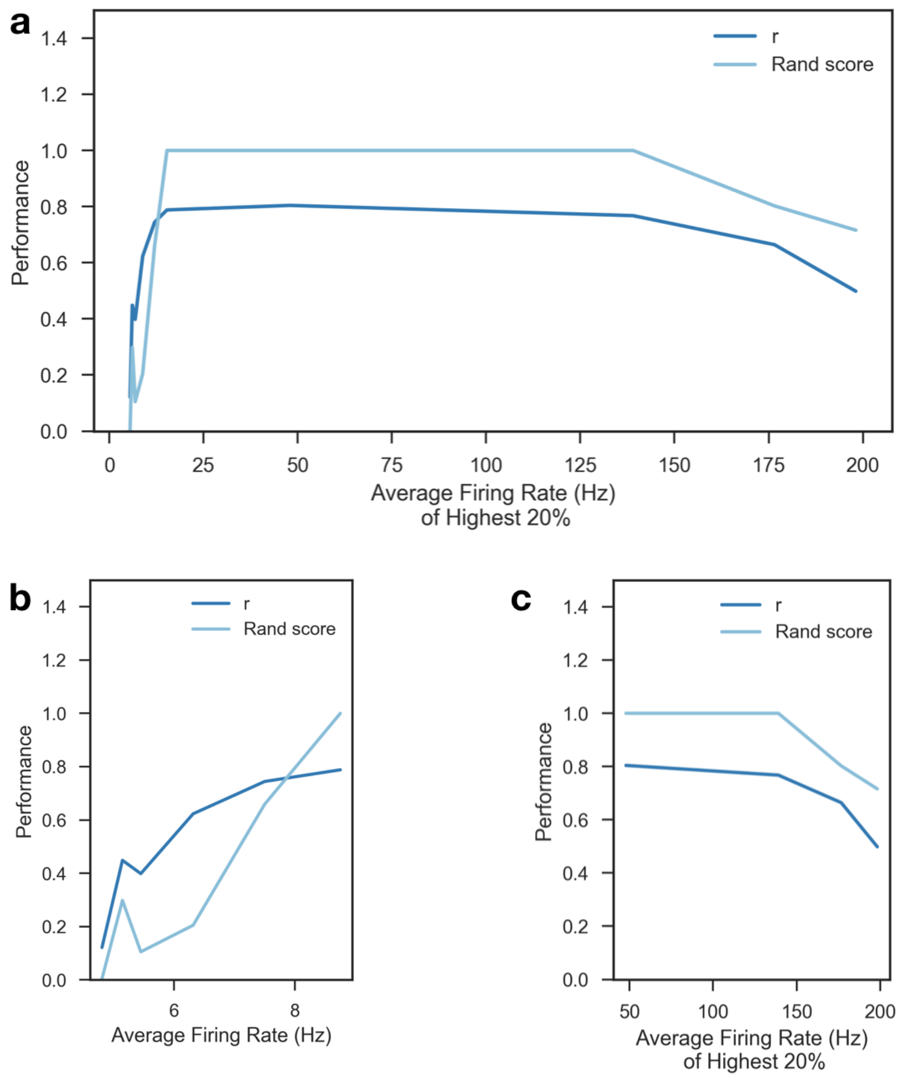

Figure 4.3: Performance measured by Rand score of module assignment and Pearson r of regression between estimated weights and ground truth weights over average firing rate (Hz) of 20% highest active neurons and average firing rate of the whole population. Highest value or r and rand score is 1.0 **(a)** Performance deteriorates in the presence of high bursting neurons with a firing rate of above 140 Hz and for populations of neurons firing in a very sparse manner. (b) and (c) show the beginning and end of graph (a) in more detail with (b) showing average instead of highest 20%. **(b)** Performance deteriorating rapidly for very sparse firing of an average firing rate below 9 Hz. **(c)** Performance over average firing rate of the 20% most active neurons showing a performance decline for neurons firing at above 140 Hz.

Data length requirements

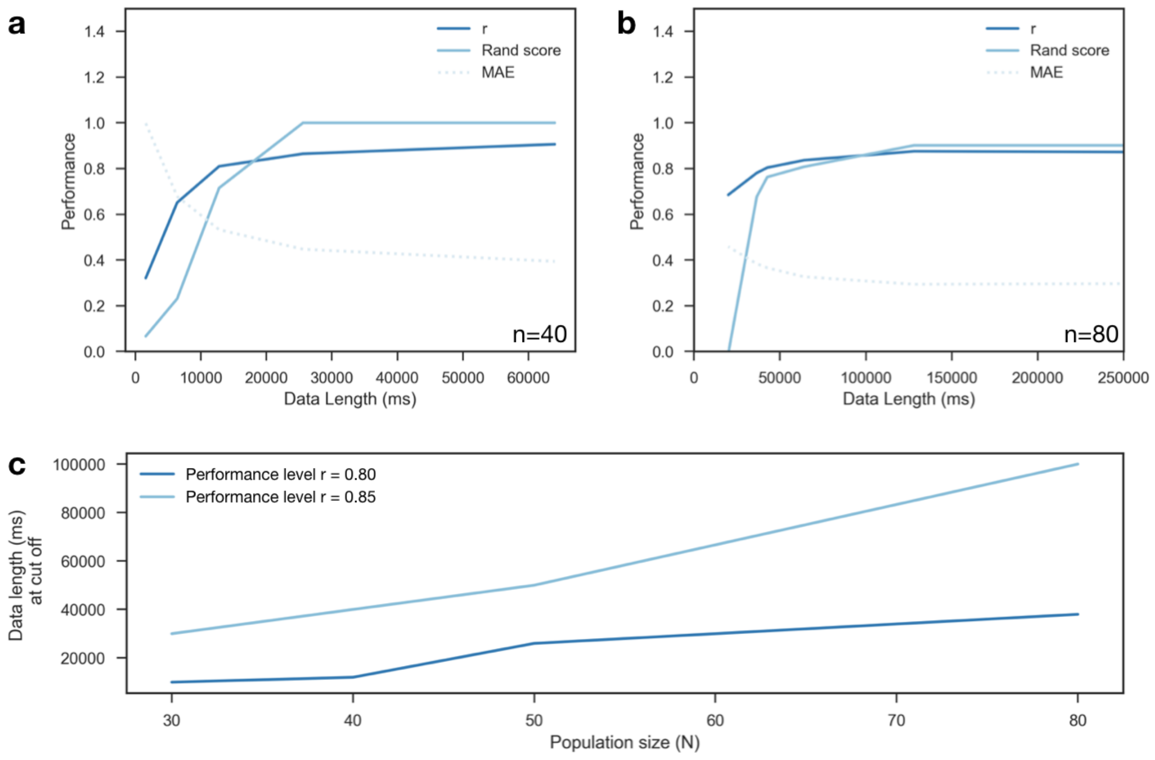

Figure 4.4: Estimation performance scales with data length. The relationship between data length requirement and number of neurons seems to be linear. **(a)** Performance over data length used for estimation on a data set of 40 neurons measured by adjusted Rand index, Person's r, and MAE. **(b)** Same as a but for 80 neurons. **(c)** Number of neurons over data length at which a performance level of r = 0.8 and 0.85 was reached respectively.

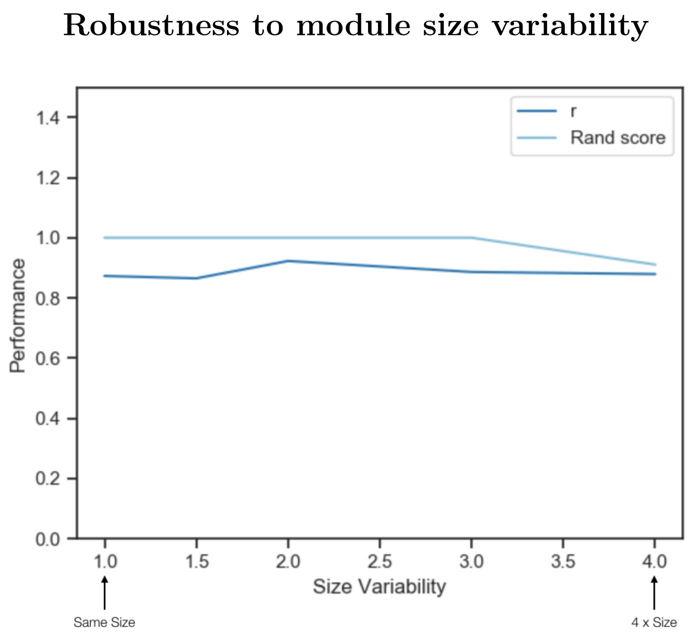

Figure 4.5: Estimations were run on data sets with two modules each starting from both being the same size (marked on the x axis at 1.0) and then one module being gradually increased in size up to 4 times the original size. As this increases the total number of neurons in the estimation data length was kept at 2000 * N data points.

a data sets which has anomalous bursting behavior (as defined and illustrated in Section 4.3.2). This is a short coming of the Faithful simulation set up and not MECI. As we have shown MECI requires bursting and sparse behavior to not accede a required range. As a result of this anomalous firing behavior in the data sets we observed a fluctuation of performance with increasing module number that we initially thought reflected a problem with the estimation algorithm itself. When strictly selecting data sets to fall in a realistic firing range performance appears to remain stable and good. We illustrate this on a data set of 8 and 20 modules. In Figure 4.6 MECI achieved strong performing at r = 0.88 and Rand score = 0.81 on a data set of 8 modules. Even at 20 modules we still observed r = 0.82 and Rand score of 0.71 (see Figure 4.7).

While the performance seems to slightly drop this is very encouraging as we do not anticipate needing such high numbers of modules for the estimation on experimental recordings.

4.3.5 Robustness to Mismatching Estimation Parameters

We now evaluate how robust MECI is to mismatching parameters initialization. The main parameters that currently have to be supplied as an initial guess are the number of modules as well as then two parameters governing weight distribution, σ_b and σ_w.

When no knowledge of cluster number is given and one tries to manually narrowing down on the appropriate number of clusters C it is advisable to start with a higher number and gradually reduce it in subsequent runs of MECI. Quality of weight estimation is less impacted by over estimating of C than by underestimating of it. Furthermore,

Example result on 8 Modules

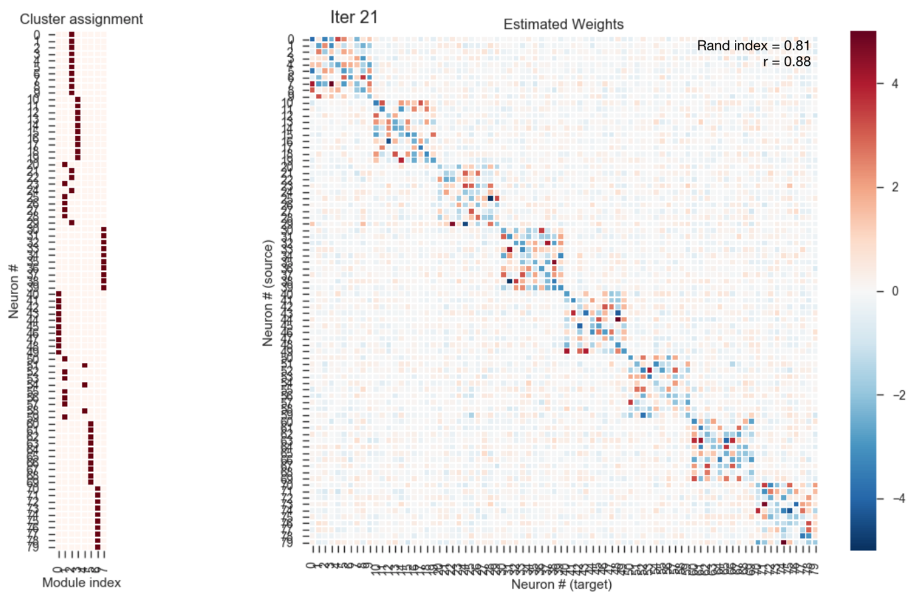

Figure 4.6: Example of robust performance at r = 0.88 and Rand score = 0.81 on a data set of 8 modules.

in the best case the algorithm will (with sufficient data length and iterations) settle on not assigning any neurons to the extra modules as seen in Figure 4.8 (b). In cases where the estimation has not converged yet, or the data set is too ambiguous, shared expression by two or more modules for one underlying module will occur as seen in Figure 4.8 (d). This can visually be identified by repeatedly running MECI. If module assignment is shared this usually shows in a flip-floping behavior of neuron assignment between shared modules. This type of behavior indicates the estimation cannot settle down on a specific assignment with high confidence.

You can see in Figure 4.8 (e) that for this particular data set performance as measured by r actually did not decline. This sometimes happens when the data length is sufficiently long enough to outweigh the benefits of module assignment as described in section 4.3.3.

The algorithm is robust to the over-estimations of σ_b and σ_w. In fact, in cases where faithful σ_b is very close to zero a slightly higher initial value for σ_b has been found to be beneficial for estimation and can sometimes turn estimation performance around from mediocre to rather good. The reason for this is a follows: If between module connections are estimated to be too close to zero assigning a neuron to a different module has very little impact on these weights even if the assignment is wrong as the small size negates the negative effect such a wrong assignment might have. By setting σ_b slightly higher the consequences of assigning a neuron to an incorrect module will show more pronounced effects thus making it easier to identify unfavorable assignments.

The algorithm reacts badly to underestimation of both sigmas but has a higher tolerance to overestimation. In the case of σ_b this is bounded by the requirement that σ_b has to be strictly smaller than σ_w. If we overestimate both σ_w and σ_b though the

Example result on 20 modules

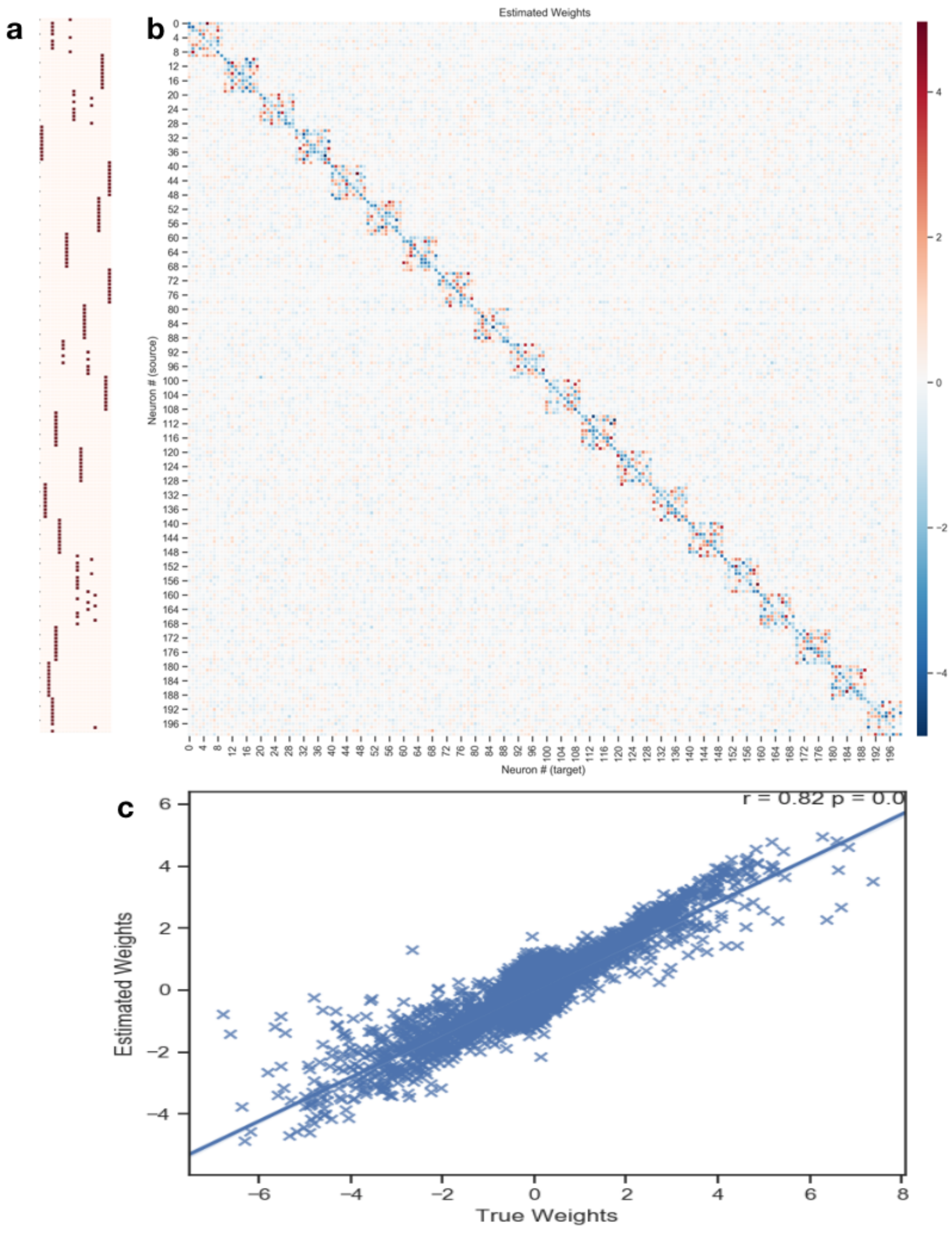

Figure 4.7: Example of robust performance at r = 0.82 and Rand score = 0.71 on a data set of 20 modules.

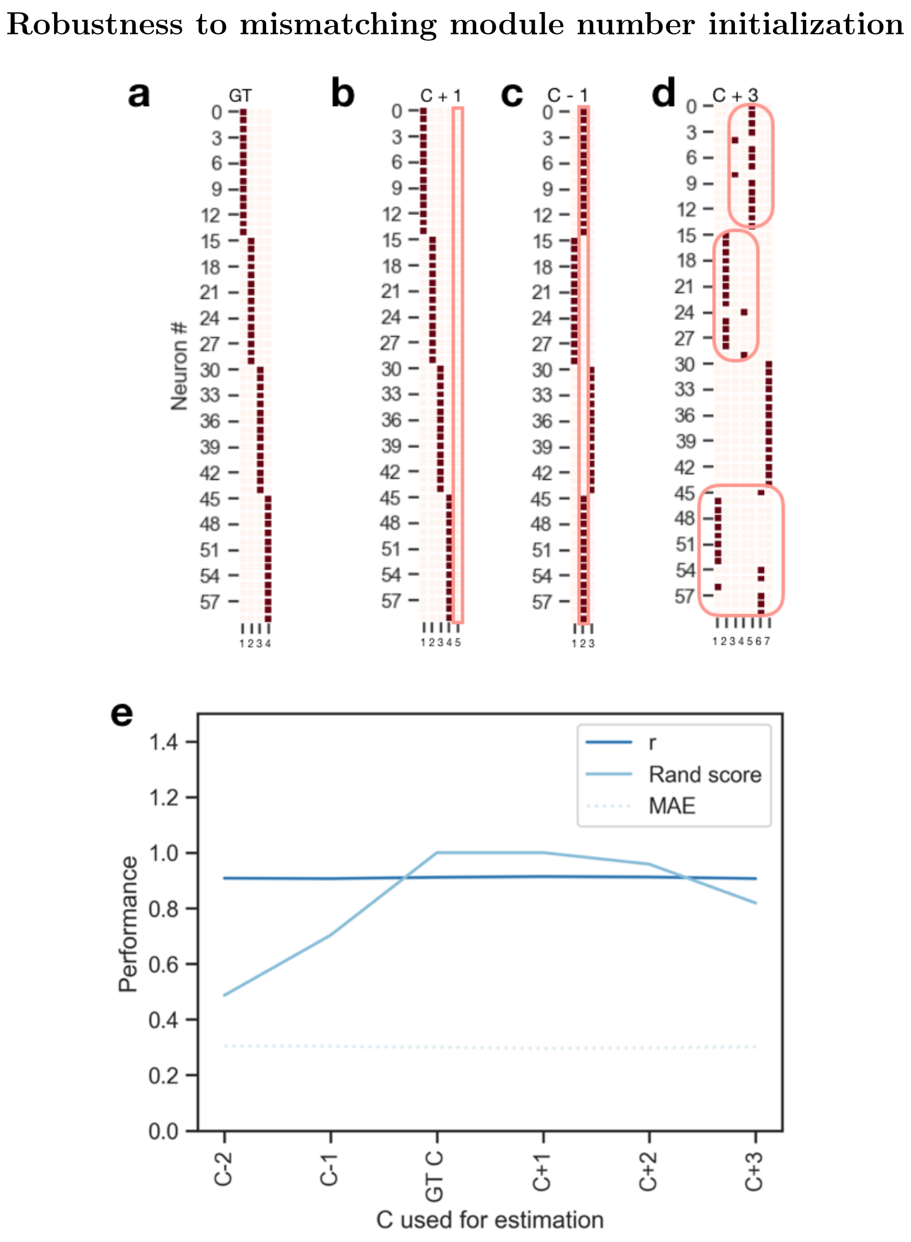

Figure 4.8: **(a)** Ground truth module assignment with C = 4 modules used in this estimation. **(b)** Best case behavior when with assuming module number to be one higher than ground truth (C+1). MECI converged to a clear assignment of all neurons to three separate clusters leaving the extra module empty. **(c)** Clear collapse of two modules into one assignment in the case of unfaithful assumption of C -1. **(d)** A more typical example of modules sharing assignments in case of assuming three extra modules. It is still possible to clearly see all four modules but some neurons are different enough in their activity to be considered belonging to a separate module. **(e)** Overview of how the performance measured in Rand index and r declines with increasing miss-match.

Robustness to mismatching sigmas

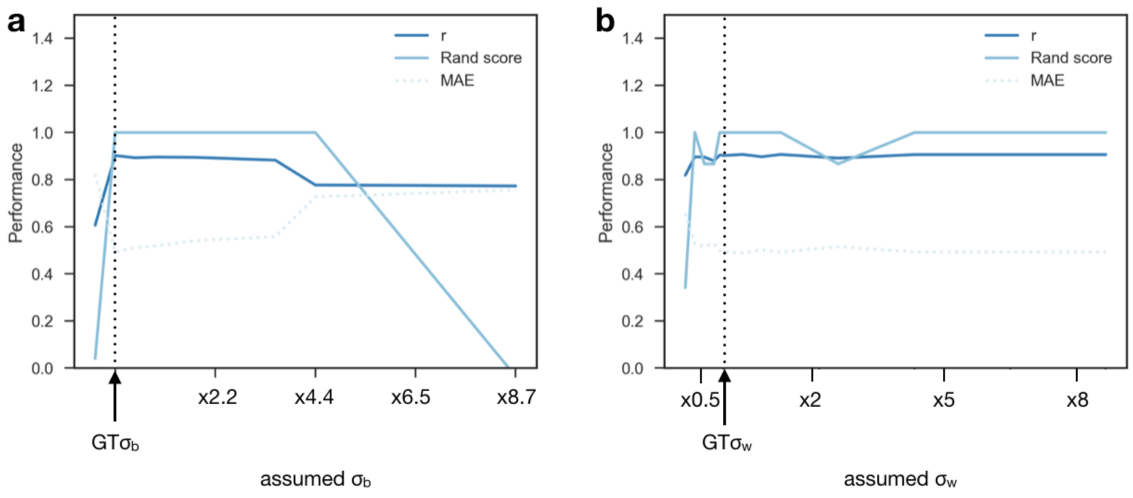

Figure 4.9: Showing robustness to unfaithful parameter assumptions for both sigmas on a data set for which ground truth was $\sigma_w = 2.3$ and $\sigma_b = 0.23$. **(a)** σ_b is bound below by 0 and above by σ_w. Therefore we only show adjustment from ground truth times 0.87 up to times 8.7 as any further to the right of the graph would be below zero and to the left above ground truth σ_w, which is prohibited by the algorithms assumption that $\sigma_b << \sigma_w$. Performance declines rapidly for underestimated values but shows a high tolerance up to close to σ_w. **(b)** The algorithm shows a high tolerance for overestimation of σ_w.

tolerance of σ_b can be extended to a certain degree as long as the ratio between the two stays in roughly correct.

MECI estimation on a distance based data set

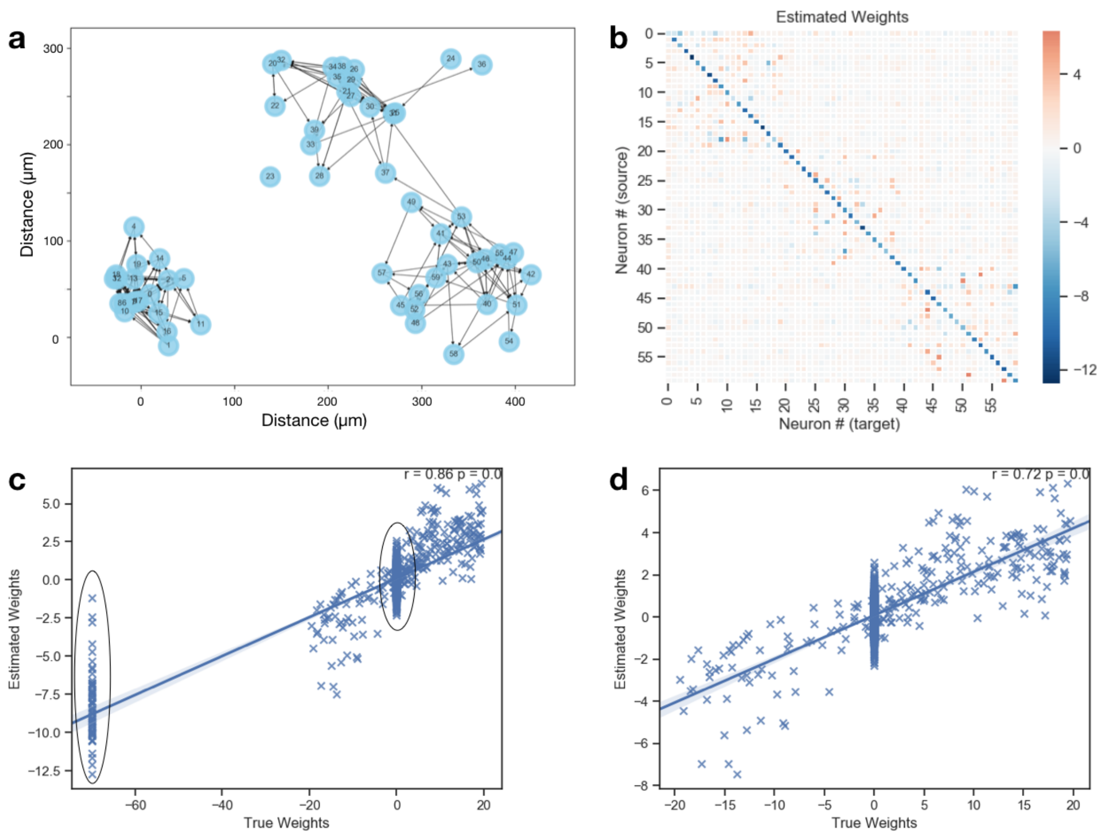

Figure 4.10: **(a)** Schematic of the neurons' placement during data simulation. **(b)** Estimated connection weights. **(c)** Regression plot between estimated and ground truth weights. Notable features are problematic areas for MECI are negative self-connections and sparse weights. **(d)** Same plot but with self connections removed to illustrate the influence on the regression line.

4.4 Comparison of MECI and SECI

Comparing MECI to any version of SECI on a data set like Faithful would be pointless as Faithful is a densely connected network while, in particularly SECI-L1 assumes a sparse connection matrix. It is though possible to compare MECI and SECI under certain conditions. SECI can be applied to a modular network if that modularity is produced as a side effect of spatial separation. MECI, on the other hand, can be applied to modular structures of either spatial or functional nature.

We now show an application of MECI to a distance based set in which modules are a side product of spatial distance. The construction of this set is the same as for Chapter 3. The only difference is that we selected neuron's positioning to follow three Gaussian distribution thus producing three clusters of neurons as seen in 4.10 (a). On such a set we can compare the performance of MECI to SECI.

4.4.1 MECI on Distance-Based Set with Modules

To illustrate the application of MECI to a data set that doesn't follow the same assumptions as well as to compare MECI and SECI we return to our distance based simulation setup as discussed in in Chapter 3. As you recall from Chapter 3 the assumptions for the simulation of this distance based data set differ from Faithful. Neurons are placed on a 500 μm x 500 μm plane and are wired based on their proximity taking bio-physiological recording statistics into consideration for this wiring. This setup also exhibits a sparse connectivity at below 20% of connections which is also more realistic than in Faithful.

Let's consider how MECI's prior assumptions differ from the set up of these population simulations. MECI enforces modules by using in-between and within module weight distribution parameters σ_b and σ_w which leads to the assumption of a weight distribution following two Gaussians. Visually this looks like two bell curves. In comparison the distance-based simulation produces weights distributions which are zero inflated. Furthermore, in this simulation weights aren't drawn from two distributions but are continuously assigned with different probability taking the distance between neurons into account thus each module would have a different weight distribution.

Despite these differences in assumption MECI shows robust estimation results. It is able to estimate both weights and module membership as approximated by distance clustering. If we assume spatial clusters are equivalent to modules MECI fully recovers the three modules and estimates their weights well. The biggest difference to runs on Faithful we observed was the number of iterations needed to converge. Runs on Faithful sometimes converge rather quickly with very few iterations of the outer sampling loop while on the distance-based data sets iteration number of at least several hundred iterations were needed. Given that sampling approaches are notorious for needing a lot of iterations it is not surprising we need a larger number of iterations on this data set.

The biggest problems encountered for this data set, is the overestimation of connections with weight 0 as MECI does not explicitly account for this type of sparseness. This is illustrated by the spread around zero in the regression plots. Since we rely on σ_b not being too close to zero one can understand the wider spread of weights supposed to be zero. It makes sense that these weights would be influenced by our assumed parameters in this way. Furthermore, you can observe the same peculiarity of negative self-connections. It uses self-connections as way to express refractory period. Like in Chapter 3 we illustrate the effect the negative self-connections have on the regression plot we show how the regression plots with and without self-connections differ as seen in 4.10 c and d. Contrary to the effects self-connections had on the regression plot in Chapter 3 we see them artificially inflated the performance here.

Nevertheless, when considering neuron to neuron connections MECI performance well on this type of data at r = 0.72 despite miss-matching assumptions, and as we show in the next section, comparably to SECI-L2.

4.4.2 SECI on Distance-Based Set with Modules

We show an application of SECI-L2 to the same data set. If we, as in the case of MECI, remove negative self connections from the weight comparison MECI and SECI-

SECI-L2 estimation on a distance based modules

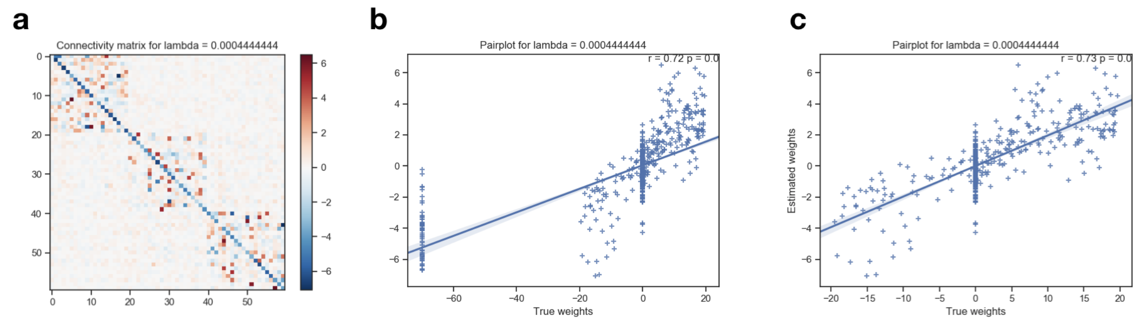

Figure 4.11: **(a)** Estimated weight matrix. **(b)** Regression plot including negative self-connections with r=0.72 **(c)** Regression plot between estimated and ground truth weights excluding self connections showing r=0.73

L2 perform comparably on this data set with SECI-L2 with r = 0.72 and r = 0.73 respectively. A bit of caution is needed when comparing these numbers. They are impacted by the spread around zero weights. SECI is directly aiming to reduce this spread while MECI is actually producing it as a side effect of parameter σ_b.

It is though encouraging to see that MECI does comparatively well on this set. It needs though much longer to find a good solution than SECI-L2. Thus while MECI can successfully produce estimations on this set SECI-L2 is still preferable for sets with distance-based properties as it matches the underlying wiring rules.

4.5 Discussion

We now discuss MECI from a broader perspective, considering limitations and suggesting possible future directions.

4.5.1 Main findings

We derived an effective neural connectivity inference (MECI) algorithm that imposes a prior over within and between module connections by assuming different distribution for each. We showed that MECI performs well on a simulated data set which was produced following these assumptions. MECI has been shown to be robust to various conditions and diverging inital model parameters. We compared MECI to SECI on a distance-based data set with modules which were formed as a side effect of distance based wiring of neural populations located in spatial clusters. MECI performed as well as SECI on this data set showing its applicability to a data set which doesn't follow it's assumptions completely but shows modularity.

4.5.2 Limitations

Simplistic weight distribution assumption As we have already briefly mentioned MECI takes an oversimplified view of neural connection statistics. Taking a simplified view is in general not a bad thing by Occam's razor it is preferable to choose the simpler explanation if it adequately captures the situation we want to express. Nevertheless, it should be noted that connectivity in the brain show various statistics. We showed one specific plausible assumption here. The assumption that between and within module connections differ and can be approximated by two Gaussian distributions with $\sigma_w > \sigma_b$. We assume here that within module connections are always stronger than between but for specific structures in the brain this is not the case. For example in a recent the Basal Ganglia Model connections strength between some modules are higher than within module connections [32].

Thus a more general formulation of a modularity prior could help express various connection statistics.

Straight forward adaptation that come to mind are, fitting different distributions per module instead of just two as well as distributions other than Gaussian.

4.5.3 Future Directions

To conclude this chapter we discuss a selection of future directions:

Autonomous search for best distribution In the current implementation of MECI two fixed Gaussian distributions defined by mean 0 and standard deviation $\sigma_w$for within and σ_b for between module connections needs to be supplied. An extension to consider is a dynamically search for the best parameters σ_w and σ_b.

More than two distributions MECI assumed two Gaussian distributions. One for all between module connections and one for all within module connections. It is though easy to imagine that the weight distribution might differ from module to module, particularly if we look at consider modules like anatomical regions that specialize for different types of computations. Thus each module could have their own distinct distribution.

Different distributions than Gaussian It is easy to see that two Gaussian distributions are a strong simplification. The straight forward extension is to explore different types of distributions.

Other forms of formulating modularity Modularity can be formulated in many different ways and we just explored but one of theme here. Another possibility we considered was using mean activation. In such an approach in between module connections could be approximated as a mean activity. We constructed a small proof of concept for his direction but didn't persue it further.

Constructing a more general formulation of the modularity prior. A more general formulation which can adapt to different modular organizations such as the three points above.

Larger populations and HPC cluster version In principle, given sufficiently long recordings MECI should perform well on larger populations of neurons. As computation cost increases with number of neurons the implementation of MECI for HPC clusters with parallelization of the N optimization terms is advisable.

Application to experimental recordings We didn't get to apply MECI to an appropriate data set of experimental recordings, yet. This had to be delegated to future work due to time constraints. We are aiming towards doing so before publishing to a journal or conference.

Extension to a hierarchical algorithm MECI can possibly be applied to larger populations of neurons to capture the influence of a global modular structure. It is thus conceivable that for example the combination with a local method like SECI could help in the estimation of larger populations of neurons which span over several anatomical areas. In such a combination local effects produced by distance could be captured by SECI and global effects produced by module structre by MECI.

4.5.4 Conclusion

In this chapter we introduced the Modular Effective Connectivity Inference (MECI) approach with one specific implementation of the main idea. The main idea is based on the assumption that neural wiring within a module is stronger than between modules.

We analyzed MECI's performance on simulated data and found that bursting and sparse neural activity have a significant impact on performance. Data length requirements increase linearly with number of neurons considered. MECI is robust to the number and size variability of modules in a data set. MECI is robust to mismatching estimation parameters to an extend.

Furthermore, we compared MECI and SECI on a distance based data set with modules produced as a side effect of spatial positioning and found that MECI can catch local spatial properties as well as SECI if these properties exhibit a module structure.

We conclude that the use of modularity-based priors has merrit in the greater context of neural connectivity inference and analysis.

Chapter 5

Discussion

In this discussion we first briefly look at practical challenges and work that paved the way towards the main body of research we presented in Chapter 3 and 4. Then we review the main findings and put them into the broader research context and conclude with a section on novelty and significance.

5.1 Review of Main Findings

In this thesis we investigated two types of priors and their viability in the use of a Bayesian approach to effective neural connectivity inference:

1. Distance-based prior
2. Modularity-based prior

5.1.1 Distance-based prior

We showed that including spatial information such as distance is a viable prior by formulating three algorithms which implement this type of prior: SECI-L1, SECI-L2 and Hi-SECI.

All of these relay on bio-physiological measurements which have reported a distance dependent connectivity probability for different types of neurons of up to 300 μm distance. Only Hi-SECI implements this fact in a direct fashion by a maximum a posteriori approach which imposes existence or absence of connections by a distance based prior over the connection matrix C via Metropolis-Hastings sampling. In contrast SECI-L1 and SECI-L2 rely on the implication that we can express this regularization implicitly on the weight matrix W itself as W implies the connection matrix C.

SECI-L1 extends L1 regularization by including a distance term in the penalty term and estimates connection weights by maximum likelihood estimation of model parameters of an LNP cascade model using gradient decent as an optimization methods.

SECI-L2 extends L2 regularization in the same fashion by adding a distance term and estimates connection weights by maximum likelihood estimation of model parameters of an LNP cascade model using Newton's method as an optimization methods.

Application of a distance-based prior leads to performance improvement as compared to sparseness regularization or inference with no prior. While Hi-SECI is the most theoretically appropriate formulation of the distance prior it is also the algorithm among the three versions which has the highest computation cost due to sampling method use. SECI-L2 is a much faster version as it directly estimated the weight matrix using Newton's method.

The performance gain of Hi-SECI over SECI-L2 is minimal despite SECI-L2 taking an extra step of abstraction by estimating weight strength directly. SECI-L1 in many data sets didn't show much improvement over L1 regularization.

A distance-based prior is though only supported for distances up to $300\mu m$ making SECI is localized inference method.

Lastly, we observed a sensitivity to module parameter changes which influence the overall synchronicity in a data set.

5.1.2 Modularity-based prior

We showed that taking modularity of neural populations into consideration in connectivity inference can be an effective prior for appropriate data sets.

We derived a modularity regularized inference algorithm called MECI in which we express modularity by including two different distributions into the LNP cascade model. One distribution for within module connections and one for between neuron connections. We approximate sampling from the joined distribution over module assignments and weights via a Gibbs sampling approach that iterates between optimizing weights and module assignments.

MECI performs very well on data sets which exhibit a modular structure. We show performance on both data sets which directly followed the assumption of having two distributions as well as on a data set that exhibits modules based on distance based wiring when we compared MECI to SECI. We thus showed that the modular structure doesn't necessarily need to strictly follow MECI's assumptions to achieve good estimation results.

Furthermore, MECI is robust to mismatching model parameters to a certain degree
.

Modularity-based inference shows more robustness than spatial-based inference. Modularity-based inference could be used in global data sets spanning across different anatomical regions. The modularity-based based inference formulation we showed here is but one way to express modularity and a more generalized formulation could lead to many variants for specific modular structures in the brain.

5.2 Contrasting SECI and MECI

SECI is a localized method. By virtue of its underlying assumption it is restricted to distances of probably not much more than $300\mu m$. MECI on the other hand is a globalized method. By virtue of its assumption which is more abstract than SECI's it can be applied to modularity of structural as well as functional nature. Possible application range from estimation of connectivity between cortical columns up to larger areas like anatomical brain regions that have been identified to contribute to different tasks.

Both approaches could possibly synergize well when combined. In larger recordings of neural population that local as well as global properties might need to both be considered to estimate meaningful module assignments and weights. We have shown that MECI can pick up local spatial effects as a side effect of its assumption. This suggest an extension to MECI which allows for both local effects and global structures to be combined.

5.3 Challenges and Limitations

Overarching challenges which aren't specific to either prior are:

Hidden neurons and external input We encountered potential effects of hidden neurons/external input to the network in this work. The consideration of such effects are an important challenge in neural connectivity inference and we see that recent approaches considered this influence [27] and [26] offer an information-theoretic framework based approach to address the challenge of unobserved neurons.

Pre-processing Prepossessing steps applied to the experimental data such as the use of tracing algorithms for the recognition of neurons in calcium imaging data or spike estimation methods like we used in this work (OASIS [13]) can have an impact on the subsequently used inference algorithm. This influence needs to be considered more closely.

Simplistic Model Structure In this work we chose simplicity of formulation over bio-pysiological accuracy in working with the LNP model. While the LNP model is an often used approximation its major drawback is the Poisson process's lack of accuracy in capturing spike train statistics [6, 28, 52]. Extensions of our proposed methods by adapting the existing model or by using more bio-physiological appropriate models are therefore desirable. We discuss this in more detail in Section 5.4.1

5.4 Future Directions

5.4.1 Technical improvements

Technical improvements that haven't been discussed in the individual discussion sections are as follows.

Model related improvements A strait-forward extension is either the use of more bio-physiological accurate model such as the calcium dynamic model as seen in [38]. Another option instead of moving on to a completely different model would be to extend the currently used LNP model to address specific challenges encountered in this work. Two of the main extensions which have the potential to lead to significant improvement are an explicit inclusion and estimation of (1) refractory periods and (2) external inputs to the whole network. Such an input variable could help explain potential behavioral correlates in the analysis of experimental recordings such as the audio input in the PCC date set we used in Chapter 3. Refractory periods aren't necessarily the same in the whole network. Different types of neurons as well as neurons in different modules can exhibit different length of refractory period. Thus, the combination of explicitly addressed external inputs as well as refractory periods could resolve the positive and negative self-connection proxy effect encountered to varying degrees in Chapter 3 and 4.

High performance computation cluster extension Computation costs makes application to larger data sets impractical at this stage. We thus only reported results for limited numbers of neurons here. A straight forward extension of the current implementations of SECI and MECI that allows for parallel estimation of N terms to be run on high performance computing clusters is though only a question of technicality and should scale well.

Optimization of run time While a certain degree of practical improvement can be achieved via parallel computing and the use of HPC clusters the main problem of high run time remains. Particularly, Hi-SECI, which is theoretically appealing but computationally costly due to the use of Metropolis Hastings sampling, could benefit greatly from being reworked into a less costly approach using for example Expectation Maximization and/or Simulated Annealing.

Combination of both approaches A combination of modular global regularization and local distance based regularization seem like a promising research direction.

Combination with other methods to address open challenges and limitations Some of the challenges discussed in Section 5.3 have already been investigated by other studies [26, 27] and our two approaches discussed in this thesis can be extended to synergze with such recent approaches which address the following challenges not addressed by our two methods:

- Cell type differentiation
- Hidden neurons
- External input

5.4.2 Application to additional data sets

Application to larger populations Due to computational cost restrictions we only applied SECI and MECI to data sets of limited size in this thesis. In principle the application to larger sets should scale linearly as shown in Chapter 3 and 4 and is one of our future aims.

Application of MECI to data set spanning several anatomical regions Data sets spanning several known anatomical regions do exist and would be very helpful in varying MECI's module membership estimation on experimental data. We are in discussions to obtain one such data set which spans over several regions of the mouse brain with more than 10000 neurons recorded.

Post-mortem data Recently, a combination of in-vivo recordings with subsequent post-mortem analysis such as [62] of the anatomical structure has become available. Such data sets are currently small in scale but the presented algorithms could be extended to analysis such data sets taking the anatomical measurements into account as another prior thus contributing to investigating the distinction between structural, functional and effective connectivity. As it stands now structural connectivity always informs functional and effective connectivity but distinguishing between structural effects in neural computations and purely functional effects is an interesting question.

Analysis of connectivity change over time One of the data sets we used in Chapter 3 suggests another interesting direction to take both inference algorithms. In this data set mice were trained over a long period of time and recordings from different periods in time are available. Considering changes in estimated connectivity weights such type of data is a very interesting possible application of both SECI and MECI. This type of change is frequently analyzed on fMRI data but not that much on the meso-scale data.

5.4.3 Transferability to other domains

While this thesis has been concerned with the development of tools for the neuroscience community we would like to take a broader view at applicability to other domains, in this last discussion.

Network structures with spatial and modular properties can be found in many domains. Any problem that can be formulated into an estimation of an unknown weighted graph structure in which nodes of the graph influence each other via their weighted connections to produce observable actions or outputs can potentially be a target for the kind of network inference algorithms presented here. In case of such application the model chosen would need to be adapted to the new domain but the algorithmic principles of including prior domain knowledge such as modules or distance would remain the same. Potential applications are for example in social media, in which people produce actions such as online posts. In such a social network information flow and is propagated by the members of a social network. Members are posting and re-posting information, such as news or fake news. The analysis of such networks could help identify clusters of people with strong contribution to propagation of such information. Other potential applications might be found in the stock market as each stock can be seen as members of a network which influences each other producing an output. With stocks we have both spatial information as well as modular organization, eg.: stocks form the same country or industry. These are just two examples of possible applications which exemplify the broader applicability of the suggested methods beyond the inference of effective neural connectivity.

Chapter 6

Conclusion

In this thesis we investigated the viability of using spatial and modular information in the inference of effective neural connectivity. We derived two main approaches SECI and MECI as well as sub variants.

While previous works, as discussed in Chapter 2, have studied the use of regularization such as spareness of connectivity, to the best of our knowledge, neither spatial (distance) nor modularity priors in the sense we presented them here has been investigated in the context neural networks on the meso-scale before.

The formulation of our two approaches thus presents a novel contribution.

We have shown both methods to perform well on simulated data sets. SECI-L2 and Hi-SECI performed better an sparse regularization alone. We expect spatial-based regularization to have a limited impact as the distance dependence rapidly falls over small distances of up to $300\mu m$ which leads to its application being reserved to populations of neurons within such a field of view. The distance based regularization might though be used in conjunction with other more global methods.

Modularity-based regularization on the other hand has potential to contribute to integrate of different scales of analysis. It is not restricted to only spatially organized modules and can be extended to other module structures. Our evaluation shows that the introduction of a modularity prior has merit and in the specific implementation of our MECI algorithm showed to be robust to many conditions, such as diverging model parameters initialization.

Bibliography

[1] A. Aertsen, T. Bonhoeffer, and J. Kruger. Coherent activity in neuronal populations: analysis and interpretation. *Physics of cognitive processes*, pages 1–34, 1987.

[2] A. Aertsen, G. Gerstein, M. Habib, and G. Palm. Dynamics of neuronal firing correlation: modulation of effective connectivity. *Journal of neurophysiology*, 61 (5):900–917, 1989.

[3] A. M. Aertsen and G. L. Gerstein. Evaluation of neuronal connectivity: sensitivity of cross-correlation. *Brain research*, 340(2):341–354, 1985.

[4] A. Akrami, C. D. Kopec, M. E. Diamond, and C. D. Brody. Posterior parietal cortex represents sensory history and mediates its effects on behaviour. *Nature*, 554(7692):368, 2018.

[5] L. Barnett, A. B. Barrett, and A. K. Seth. Granger causality and transfer entropy are equivalent for gaussian variables. *Physical review letters*, 103(23):238701, 2009.

[6] M. J. Berry II and M. Meister. Refractoriness and neural precision. In *Advances in Neural Information Processing Systems*, pages 110–116, 1998.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. ISBN 978-0387-31073-2.

[8] S. P. Brooks and A. Gelman. General methods for monitoring convergence of iterative simulations. *Journal of computational and graphical statistics*, 7(4):434–455, 1998.

[9] E. Chichilnisky. A simple white noise analysis of neuronal light responses. *Network: Computation in Neural Systems*, 12(2):199–213, 2001.

[10] I. M. de Abril, J. Yoshimoto, and K. Doya. Connectivity inference from neural recording data: Challenges, mathematical bases and research directions. *Neural Networks*, 102, 2018.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[12] A. K. Fletcher and S. Rangan. Scalable inference for neuronal connectivity from calcium imaging. In *Advances in Neural Information Processing Systems*, pages 2843–2851, 2014.

[13] J. Friedrich, P. Zhou, and L. Paninski. Fast online deconvolution of calcium imaging data. *PLoS computational biology*, 13(3):e1005423, 2017.

[14] K. Friston, C. Frith, P. Liddle, and R. Frackowiak. Functional connectivity: the principal-component analysis of large (pet) data sets. *Journal of cerebral blood flow and metabolism*, 13:5–5, 1993.

[15] K. J. Friston, A. P. Holmes, K. J. Worsley, J.-P. Poline, C. D. Frith, and R. S. Frackowiak. Statistical parametric maps in functional imaging: a general linear approach. *Human brain mapping*, 2(4):189–210, 1994.

[16] A. Funamizu, B. Kuhn, and K. Doya. Neural substrate of dynamic bayesian inference in the cerebral cortex. *Nature neuroscience*, 19(12):1682, 2016.

[17] A. Gelman, D. B. Rubin, et al. Inference from iterative simulation using multiple sequences. *Statistical science*, 7(4):457–472, 1992.

[18] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, 6(6):721–741, 1984.

[19] G. L. Gerstein. Joint peri stimulus time histogram (jpsth). *Encyclopedia of Computational Neuroscience*, pages 1–4, 2014.

[20] W. Gerstner, W. Kistler, R. Naud, and L. Paninski. *Neuronal Dynamics*, volume 1. Cambridge University Press, 2014.

[21] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[22] M. J. Hawrylycz, E. S. Lein, A. L. Guillozet-Bongaarts, E. H. Shen, L. Ng, J. A. Miller, L. N. Van De Lagemaat, K. A. Smith, A. Ebbert, Z. L. Riley, et al. An anatomically comprehensive atlas of the adult human brain transcriptome. *Nature*, 489(7416):391, 2012.

[23] D. H. Hubel, T. N. Wiesel, and M. P. Stryker. Orientation columns in macaque monkey visual cortex demonstrated by the 2-deoxyglucose autoradiographic technique. *Nature*, 269(5626):328, 1977.

[24] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1): 193–218, 1985.

[25] T. R. Insel, S. C. Landis, and F. S. Collins. The nih brain initiative. *Science*, 340 (6133):687–688, 2013.

[26] T. Iwasaki, H. Hino, M. Tatsuno, S. Akaho, and N. Murata. Estimation of neural connections from partially observed neural spikes. *Neural Networks*, 108:172–191, 12 2018.

[27] A. Karbasi, A. H. Salavati, and M. Vetterli. Learning neural connectivity from firing activity: efficient algorithms with provable guarantees on topology. *Journal of computational neuroscience*, 44(2):253–272, 2018.

[28] J. Keat, P. Reinagel, R. C. Reid, and M. Meister. Predicting every spike: a model for the responses of visual neurons. *Neuron*, 30(3):803–817, 2001.

[29] E. S. Lein, M. J. Hawrylycz, N. Ao, M. Ayres, A. Bensinger, A. Bernard, A. F. Boe, M. S. Boguski, K. S. Brockway, E. J. Byrnes, et al. Genome-wide atlas of gene expression in the adult mouse brain. *Nature*, 445(7124):168, 2007.

[30] R. B. Levy and A. D. Reyes. Spatial profile of excitatory and inhibitory synaptic connectivity in mouse primary auditory cortex. *Journal of Neuroscience*, 32(16): 5609–5619, 2012.

[31] X. Li and G. Ouyang. Estimating coupling direction between neuronal populations with permutation conditional mutual information. *NeuroImage*, 52(2):497–507, 2010.

[32] J. Liénard and B. Girard. A biologically constrained model of the whole basal ganglia addressing the paradoxes of connections and selection. *Journal of Computational Neuroscience*, 36(3):445–468, 2014.

[33] A. A. Markov. The theory of algorithms. *Trudy Matematicheskogo Instituta Imeni VA Steklova*, 42:3–375, 1954.

[34] H. Markram. The human brain project. *Scientific American*, 306(6):50–55, 2012.

[35] W. Melssen and W. Epping. Detection and estimation of neural connectivity based on crosscorrelation analysis. *Biological cybernetics*, 57(6):403–414, 1987.

[36] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[37] Y. Mishchenko, J. T. Vogelstein, and L. Paninski. A Bayesian approach for inferring neuronal connectivity from calcium fluorescent imaging data. *The Annals of Applied Statistics*, 5(2B):1229–1261, June 2011.

[38] Y. Mishchenko, J. T. Vogelstein, L. Paninski, et al. A bayesian approach for inferring neuronal connectivity from calcium fluorescent imaging data. *The Annals of Applied Statistics*, 5(2B):1229–1261, 2011.

[39] A. S. Morcos and C. D. Harvey. History-dependent variability in population dynamics during evidence accumulation in cortex. *Nature neuroscience*, 19(12):1672, 2016.

[40] V. B. Mountcastle. Modality and topographic properties of single neurons of cat's somatic sensory cortex. *Journal of neurophysiology*, 20(4):408–434, 1957.

[41] K. Ohki, S. Chung, Y. H. Ch'ng, P. Kara, and R. C. Reid. Functional imaging with cellular resolution reveals precise micro-architecture in visual cortex. *Nature*, 433(7026):597–603, 2005.

[42] H. Okano and P. Mitra. Brain-mapping projects using the common marmoset. *Neuroscience research*, 93:3–7, 2015.

[43] M. Okatan, M. A. Wilson, and E. N. Brown. Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity. *Neural Computation*, 17(9):1927–1961, 2005.

[44] T. E. Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[45] J. G. Orlandi, O. Stetter, J. Soriano, T. Geisel, and D. Battaglia. Transfer entropy reconstruction and labeling of neuronal connections from simulated calcium imaging. *PloS one*, 9(6):e98842, 2014.

[46] A.-M. M. Oswald and A. D. Reyes. Maturation of intrinsic and synaptic properties of layer 2/3 pyramidal neurons in mouse auditory cortex. *Journal of neurophysiology*, 99(6):2998–3008, 2008.

[47] L. Paninski. Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems*, 15(4):243–262, 2004.

[48] R. Perin, T. K. Berger, and H. Markram. A synaptic organizing principle for cortical neuronal groups. *Proceedings of the National Academy of Sciences*, 108 (13):5419–5424, 2011.

[49] J. W. Pillow, J. Shlens, L. Paninski, A. Sher, A. M. Litke, E. Chichilnisky, and E. P. Simoncelli. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995, 2008.

[50] L. Qiao, H. Zhang, M. Kim, S. Teng, L. Zhang, and D. Shen. Estimating functional brain networks by incorporating a modularity prior. *NeuroImage*, 141:399–407, 2016.

[51] A. E. Raftery and S. M. Lewis. Practical markov chain monte carlo: One long run with diagnostics: Implementation strategies for markov chain monte carlo. *Statistical science*, 7(4):493–497, 1992.

[52] D. S. Reich, J. D. Victor, and B. W. Knight. The power ratio and the interval map: spiking models and extracellular recordings. *Journal of Neuroscience*, 18 (23):10090–10104, 1998.

[53] T. Schreiber. Measuring information transfer. *Physical review letters*, 85(2):461, 2000.

[54] O. Schwartz, J. W. Pillow, N. C. Rust, and E. P. Simoncelli. Spike-triggered neural characterization. *Journal of vision*, 6(4):13–13, 2006.

[55] E. P. Simoncelli, L. Paninski, J. Pillow, O. Schwartz, et al. Characterization of neural responses with stochastic stimuli. *The cognitive neurosciences*, 3(327-338): 1, 2004.

[56] A. Singh and N. A. Lesica. Incremental mutual information: a new method for characterizing the strength and dynamics of connections in neuronal circuits. *PLoS computational biology*, 6(12):e1001035, 2010.

[57] O. Sporns. *Networks of the Brain.* MIT press, 2011.

[58] I. H. Stevenson and K. P. Kording. How advances in neural recording affect data analysis. *Nature Publishing Group*, 14(2):139–142, Feb. 2011.

[59] I. H. Stevenson, J. M. Rebesco, N. G. Hatsopoulos, Z. Haga, L. E. Miller, and K. P. Kording. Bayesian inference of functional connectivity and network structure from spikes. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 17(3):203–213, 2009.

[60] C. Stosiek, O. Garaschuk, K. Holthoff, and A. Konnerth. In vivo two-photon calcium imaging of neuronal networks. *Proceedings of the National Academy of Sciences*, 100(12):7319–7324, 2003.

[61] K. Svoboda and R. Yasuda. Principles of two-photon excitation microscopy and its applications to neuroscience. *Neuron*, 50(6):823–839, 2006.

[62] O. von Bohlen und Halbach, M. Lotze, and J. P. Pfannmöller. Post-mortem magnetic resonance microscopy (mrm) of the murine brain at 7 tesla results in a gain of resolution as compared to in vivo mrm. *Frontiers in neuroanatomy*, 8:47, 2014.