Inferring microscale parameters from macroscale eeg-data with the help of simulation-based inference

by Katharina Anderer

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

First supervisor: Prof. Jakob Macke

Second supervisor: Prof. Martin Butz

Day Month 2022

Abstract

Abstract goes here

Declaration

Hiermit erkläre ich, dass ich diese schriftliche Abschlussarbeit selbstständig verfasst habe, keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe und alle wörtlich oder sinngemäß aus anderen Werken übernommenen Aussagen als solche gekennzeichnet habe.

Datum, Ort, Unterschrift

Acknowledgements

I want to thank...

Contents

Introduction

Understanding how macroscale signals evolve from microscale parameters is an interesting question in many domains, e.g. in research about global climate, gene expression or brain phenomena like the signal coming from an electroencephalography (EEG). The later is an example for a macroscale signal of the brain that evolves through the activation of many neurons that fire in parallel. In more detail, it measures the intracellular current flow in the long and spatially-aligned pyramidal neuron dendrites [2]. While macroscale signals are the product out of the combination of many signals, we are often interested in the origins of these signals - the underlying mechanisms. These mechanisms can be described by a so called mechanistic model that meets assumptions about e.g. the information flow circuits, the morphology of the cells in the brain or the weights between different neurons. The more parameters this mechanistic model has, the more difficult it gets to infer the underlying processes that have caused the output of an EEG signal.
One approach to find the underlying parameters of a macroscale signal is to simulate lots of different parameter settings with the help of a simulator that captures the met assumptions of a certain mechanistic model.

The Human Neocorical Neurosolver (HNN), developed by [2], is an example of such a simulator and was used in this work to gain thousands of simulations to later work with and infer a posterior density of the parameters of interest.
HNN is based on a model that tries to represent the neocortical circuits of pyramidal neurons and interneurons. The model has a 3-layered structure with pyramidal neurons and inhibitory interneuons in a 3-to-1 ratio of pyramidal to inhibitory cells [2]. The 3 layers that are modeled are Layer 2/3 (also referred to as supragraUnderstanding how macroscale signals evolve from microscale parameters is an interesting question in many domains, e.g. in research about global climate, gene expression or brain phenomena like the

signal coming from an electroencephalography (EEG). The later is an example for a macroscale signal of the brain that evolves through the activation of many neurons that fire in parallel. In more detail, it measures the intracellular current flow in the long and spatially-aligned pyramidal neuron dendrites [2]. While macroscale signals are the product out of the combination of many signals, we are often interested in the origins of these signals - the underlying mechanisms. These mechanisms can be described by a so called mechanistic model that meets assumptions about e.g. the information flow circuits, the morphology of the cells in the brain or the weights between different neurons. The more parameters this mechanistic model has, the more difficult it gets to infer the underlying processes that have caused the output of an EEG signal. One approach to find the underlying parameters of a macroscale signal is to simulate lots of different parameter settings with the help of a simulator that captures the met assumptions of a certain mechanistic model.

The Human Neocorical Neurosolver (HNN), developed by Neymotin et al. (2020), is an example of such a simulator and was used in this work to gain thousands of simulations to later work withNeymotin et al. (2020) and infer a posterior density of the parameters of interest.
HNN is based on a model that tries to represent the neocortical circuits of pyramidal neurons and interneurons. The model has a 3-layered structure with pyramidal neurons and inhibitory interneuons in a 3-to-1 ratio of pyramidal to inhibitory cells [2]. The 3 layers that are modeled are Layer 2/3 (also referred to as supragranual layer), Layer 4 and Layer 5 (also referred to as infragranular layer).
The HNN model distinguishes between so called proximal drives, coming from the thalamus and signaling to the supragranular layers of the cortex, and so called distal drives, representing cortical-cortical inputs or non-lemniscal thalamic drive that signals directly into the supragranual layers and from there further downwards to the infragranular layers. The timing and duration of these drives can be adjusted [2].
For each evoked drive, there are up to 7 parameters that can be tuned. These include the onset of the drive, the number of spikes and the weights of synaptic inputs to the specific layers (see [2] for further details)
Besides, tonic inputs can be modeled and describe somatic current clamps that can change the resting membrane potential in both ways, specifically get it closer or further from firing [2].

HNN is based on the NEURON environment. Taken on from NEURON,

membrane voltages are based on Hodgkin-Huxley equations and current flow between compartments is modeled by cable theory [2].
Further, the model captures different ion channels like Na, K, Km, KCa and others and codes the thresholds for these [2].

Given this HNN tool, one is able to simulate signals like an EEG or MEG that is based on all these assumptions and domain knowledge about the architecture and information flow processes in the brain. One can then use these simulations in order to evaluate which microscale parameters are probably to have caused a certain EEG or MEG signal.

As we have many different parameters involved and further signals like EEG or MEG have a stochastic nature, the likelihood function $p(x|\Theta$ is intractable as one would have to trace every possible parameter set and compute the integral of this [1].

nual layer), Layer 4 and Layer 5 (also referred to as infragranular layer). The HNN model distinguishes between so called proximal drives, coming from the thalamus and signaling to the supragranular layers of the cortex, and so called distal drives, representing cortical-cortical inputs or non-lemniscal thalamic drive that signals directly into the supragranual layers and from there further downwards to the infragranular layers. The timing and duration of these drives can be adjusted [2].
For each evoked drive, there are up to 7 parameters that can be tuned. These include the onset of the drive, the number of spikes and the weights of synaptic inputs to the specific layers (see [2] for further details)
Besides, tonic inputs can be modeled and describe somatic current clamps that can change the resting membrane potential in both ways, specifically get it closer or further from firing [2].

HNN is based on the NEURON environment. Taken on from NEURON, membrane voltages are based on Hodgkin-Huxley equations and current flow between compartments is modeled by cable theory [2]. Further, the model captures different ion channels like Na, K, Km, KCa and others and codes the thresholds for these [2].

Given this HNN tool, one is able to simulate signals like an EEG or MEG that is based on all these assumptions and domain knowledge about the architecture and information flow processes in the brain. One can then use these simulations in order to evaluate which microscale parameters are probably to have caused a certain EEG or MEG signal.

As we have many different parameters involved and further, signals like EEG or MEG have a stochastic nature, the likelihood function $p(x|\Theta)$ is intractable as one would have to trace every possible parameter set and compute the integral of this [1].

For an approximation of the likelihood or also directly the posterior density, neural networks can be used. One particular neural density estimation technique is called normalizing flows, in which one starts with a simple base distribution which is put into the network and then transformed through multiple inverse transformations with a tractable Jacobian [1].

- to include:
- SBI advantages -> indeterminacy example, parameter space
- importance to investigate EEG through parameter inference, possible applications
- issues to tackle: high dimensionality, ground truth often not available, we try to tackle high dimensionality with new approach that divides the parameter inference process in separate time steps

0.1 Temporal Sequential Inference

As the parameter space gets larger, more and more simulations are required. We can approximately calculate the number of simulations that is necessary if we assume a uniform prior. If we want to sample draws within +-10% around the true parameters in each dimension, only 20% of our draw will be in this region. So, for e.g. 18 parameters $0.2^{18} = 2.62144 \cdot 10^{-13}$ we would need 10^{13} simulations in order to get around 2.6 draws within the target region.

Our proposed approach tries to tackle this issue by iteratively inferring parameters in a certain time order. The posterior of the already inferred parameters is then used as a prior in the next step. This prior is combined with the prior of the successive parameters that haven't been inferred so far.

Let's assume a parameter set Θ_1 for which only the time up to $time = i$ matters, where $i < m$ and we assume that

$$p(\Theta_1|x_{1:i}) = p(\Theta_1|x_{1:m})$$

As we do not work with the whole data x, but instead use summary statistics $s(x)$ that describe the data, we define these summary statistics in a way that there exists a time order as well:

$$s(x) = (s_1(x), ..., s_m(x))$$

such that it's true that:

$$p(\Theta_1|s_{1:i}) = p(\Theta_1|s_{1:m})$$

Now, having already inferred the first parameter set Θ_1, we define the prior for the next step in the following way:

$$\begin{aligned} p(\Theta_1, \Theta_2) &= p(\Theta_1|s_{1:i}, \Theta_2) \cdot p(\Theta_2) \\ &= p(\Theta_1|s_{1:i}) \cdot p(\Theta_2) \end{aligned}$$

as Θ_1 is not dependent on Θ_2.

In practice, we define a new distribution class that takes the posterior of Θ_1 and the prior of Θ_2 and samples the thetas of both separately. So, $\forall \theta \in \Theta_1$, we sample from $p(\Theta_1|s_{1:i})$ and $\forall \theta \in \Theta_2$, we sample from $p(\Theta_2)$.

The log-likelihood is then defined as:

$$\mathcal{L}_{\Theta_1,\Theta_2} = \mathcal{L}_{\Theta_1} + \mathcal{L}_{\Theta_2}$$

Idea: we could 'evaluate' also by mathematically investigating how much narrower the prior space already gets and how much less simulations we would therefore need. If we show that we already narrowed the parameter space to an area +-% around the true parameter, we only need as many simulations in every temporal inference step as the not already considered, new parameters need by themselves.

Let's assume Θ_1 consists of 4 parameters and Θ_2 as well. Instead of having a chance of $0.2^8 = 2.56 \cdot 10^{-6}$ to be within the 20% range around the true parameter, we have a $0.2^4 = 0.0016$ for each temporal inference round.
Instead of requiring about $5 \cdot 10^5$ simulations in order to get one draw within the 20% range, we then need less then 1000 simulations for each temporal inference round.

This would need to be validated in the results part of the thesis.. training budget, testing posterior coverage with simulation-based calibration.. benchmark paper: different metrics..

Algorithm 1: Temporal Sequential Inference

```
1  First round:;
2  Set $\widetilde{p_1}(\Theta_1) = p(\Theta_1)$;
3  for $j$ = 1: $N$ do do
4      Sample $\Theta_{1j} \sim p(\Theta_1)$;
5      Simulate $x_j \sim p(x|\Theta_j)$
6  end
7  ;
8  Set $\hat{p}(\Theta_1|x_o) = q_{F(x_o,\phi)}(\Theta_1)$ ;
9  ;
10 Second round:;
11 Set $\widetilde{p_2}(\Theta_1, \Theta_2) = \hat{p}(\Theta_1|x_o) \cdot p(\Theta_2)$;
12 for $j$ = 1: $N$ do do
13     Sample $\Theta_{1j} \sim \hat{p}(\Theta_1|x_o)$;
14     Sample $\Theta_{2j} \sim p(\Theta_2)$;
15     Simulate $x_j \sim p(x|\Theta_{1j}, \Theta_{2j})$
16 end
17 ;
18 Set $\hat{p}(\Theta_1, \Theta_2|x_o) = q_{F(x_o,\phi}(\Theta_1, \Theta_2)$ ;
19 ;
20 ... M rounds ...;
21 return Samples from $\hat{p}(\Theta_1, \Theta_2, ..., \Theta_M|x_o)$
```

Bibliography

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

[2] Samuel A Neymotin, Dylan S Daniels, Blake Caldwell, Robert A McDougal, Nicholas T Carnevale, Mainak Jas, Christopher I Moore, Michael L Hines, Matti Hämäläinen, and Stephanie R Jones. Human neocortical neurosolver (hnn), a new software tool for interpreting the cellular and network origin of human meg/eeg data. *Elife*, 9:e51214, 2020.