Active Learning of Cortical Connectivity from Two-Photon Imaging Data

Martín Bertrán[1◕*], Natalia Martínez[1◕], Ye Wang[2], David Dunson[2], Guillermo Sapiro[1,3‡], Dario Ringach[4‡]

1 Electrical and Computer Engineering, Duke University, Durham, North Carolina, USA;
2 Statistical Science Program, Duke University, Durham, North Carolina, USA;
3 BME, CS, and Math, Duke University, Durham, North Carolina, USA;
4 Neurobiology and Psychology, Jules Stein Eye Institute, Biomedical Engineering Program, David Geffen School of Medicine, University of California, Los Angeles, California, USA;

◕These authors contributed equally to this work.
‡Equal senior contributors.
* martin.bertran@duke.edu

Abstract

Understanding how groups of neurons interact within a network is a fundamental question in system neuroscience. Instead of passively observing the ongoing activity of a network, we can typically perturb its activity, either by external sensory stimulation or directly via techniques such as two-photon optogenetics. A natural question is how to use such perturbations to identify the connectivity of the network efficiently. Here we introduce a method to infer sparse connectivity graphs from *in-vivo*, two-photon imaging of population activity in response to external stimuli. A novel aspect of the work is the introduction of a recommended distribution, incrementally learned from the data, to optimally refine the inferred network.. Unlike existing system identification techniques, this "active learning" method automatically focuses its attention on key undiscovered areas of the network, instead of targeting global uncertainty indicators like parameter variance. We show how active learning leads to faster inference while, at the same time, provides confidence intervals for the network parameters. We present simulations on artificial small-world networks to validate the methods and apply the method to real data. Analysis of frequency of motifs recovered show that cortical networks are consistent with a small-world topology model.

Introduction

A fundamental question of system neuroscience is how large groups of neurons interact, within a network to perform computations that go beyond the individual ability of each one. One hypothesis is that the emergent behavior in neural networks results from their organization into a hierarchy of modular sub-networks, or motifs, each performing simpler computations than the network as a whole [1].

To test this hypothesis and to understand brain networks in general we need to develop methods that can reliably measure network connectivity, detect recurring motifs, elucidate the computations they perform, and understand how these smaller

modules are combined into larger networks capable of performing increasingly complex computations.

Here we focus on the first of these problems, which is a pre-requisite to the rest: the identification of network connectivity from *in-vivo*, two-photon imaging data. Advances in two-photon imaging are giving us the first look at how large ensembles of neurons behave *in-vivo* during complex behavioral tasks [2–4]. Developing methods capable of analyzing the connectivity between a large number neurons, from noisy, stochastic activations, and limited recording time, is a significant challenge.

It is often the case that we can probe the networks under study, instead of merely observing their ongoing activity. For example, in studying visual cortex we can select a specific visual stimulus [5–8], or we can stimulate individual neurons directly via two-photon optogenetics [9, 10]. This active observation has been shown critical for system identification beyond brain networks, and is the direction here pursued.

We introduce a method that infers a sparse connectivity graph from available simultaneous recording of external stimuli and individual neural spiking rates, and recommends the future distribution of external stimuli to apply in order to optimally refine the inferred network. We show how such iterative "active learning" leads to faster inference while at the same time providing confidence measurements for the computed network components. The proposed decision-making approach takes into account information we may already have about network connectivity, the stochastic nature of the neural responses, and the uncertainty of connection weights.

It is important to note that the inferred connectivity graph implies functional, but not necessarily anatomical, connectivity. The recovered functional connectivity must be interpreted carefully when the recorded neural population is only a subset of a larger population. A directed connection might be caused, for example, by a common (but unobserved) driving factor, or hidden intermediate nodes. Nonetheless, common driving factors associated with stimulus modulations are partially accounted for by including stimulus information, this effect has also been discussed in [11]

Our framework consists of modelling the neuron spike trains with a Poisson Generalized Linear Model (GLM) [12], using past spike trains and applied stimuli as regressors. This type of point process model on neural spiking activity was popularized by [13]. The coefficients of these regressors in the GLM make the edge weights of the directed network. A variable selection approach is used to make the network sparse (set most edges to zero). Active learning is then used to decide, at any given point in time, the next sets of stimuli that allow for optimal inference of the network connectivity. Fig 1 shows a visual representation of the proposed framework.

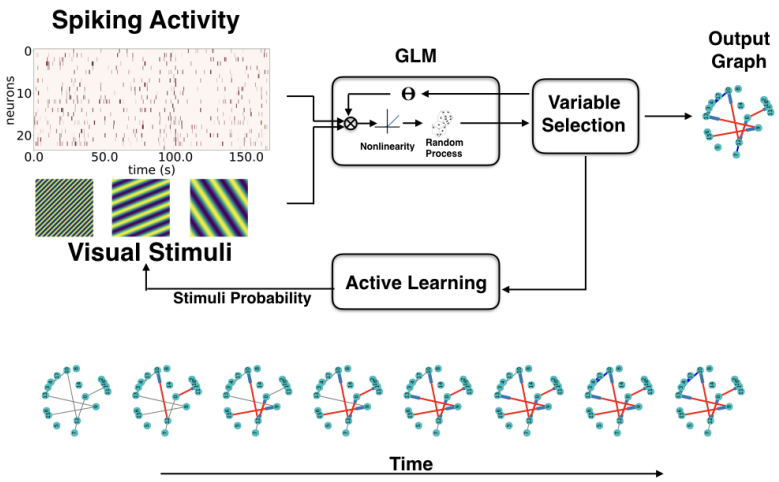


Fig 1. Recordings of spiking activity of a neuron population and the presented visual stimuli are fed into a GLM. The GLM and Variable Selection blocks work in tandem to decide which connections are relevant for explaining the system's behaviour (the data) and building the directed connectivity graph (network). The active learning component analyzes the data obtained so far to optimize the visual stimuli to be presented for the next step of data acquisition, this is done to reduce graph uncertainty. This process is iteratively repeated. The bottom row shows how the network is gradually reconstructed as a function of acquired samples. Gray edges represent yet undiscovered edges present in the network, while red and blue edges represent discovered excitatory and inhibitory edges respectively.

While the proposed framework is general, we illustrate it by applying it to two-photon imaging data from mouse primary visual cortex. We also validate the method's effectiveness on *in-silico* network simulations.

Materials and methods

Following the brief description of the data acquisition, we then describe the foundation of the proposed active network inference framework. In doing so, we use terminology that will be relevant for the particular application at hand: the estimation of connectivity in neural networks. However, the framework is general enough to be applied in other contexts and using other modalities.

Data acquisition

Animals: All procedures were approved by UCLA's Office of Animal Research Oversight (the Institutional Animal Care and Use Committee), and were in accord with guidelines set by the US National Institutes of Health. The present study used data already collected for other studies. Thus, no new animal experiments were performed for the purposes of the present study. A detailed account of the experimental methods can be found elsewhere [14]. A brief description follows.

Imaging: Imaging of GCaMP6f expressed in primary visual cortex was performed using a resonant, two-photon microscope (Neurolabware, Los Angeles, CA) controlled

by Scanbox acquisition software (Scanbox, Los Angeles, CA). The light source was a Coherent Chameleon Ultra II laser (Coherent Inc, Santa Clara, CA) running at 920nm. The objective was an x16 water immersion lens (Nikon, 0.8NA, 3mm working distance). The microscope frame rate was 15.6Hz (512 lines with a resonant mirror at 8kHz). Eye movements and pupil size were recorded via a Dalsa Genie M1280 camera (Teledyne Dalsa, Ontario, Canada) fitted with a 740 nm long-pass filter that looked at the eye indirectly through the reflection of an infrared-reflecting glass. Images were captured at an average depth of 260 µm.

Sequences of pseudo-random sinusoidal gratings [5, 15] and sparse noise stimuli were generated in real-time by a Processing sketch using OpenGL shaders (see `http://processing.org`). A detailed description is provided in [16]. The duration of the sequences was either 20 or 30 min, and gratings were updated 4 times a second on a screen refreshed at 60Hz. In a 20 min sequence, each combination of orientation and spatial frequency appeared at least 22 times on average.

In all experiments we used a BenQ XL2720Z screen which measured 60 cm by 34 cm and was viewed at 20 cm distance, subtending 112 x 80 degrees of visual angle. The screen was calibrated using a Photo-Research (Chatsworth, CA) PR-650 spectro-radiometer, and the result used to generate the appropriate gamma corrections for the red, green and blue components via an nVidia Quadro K4000 graphics card. The contrast of the stimulus was 80%. The center of the monitor was positioned with the center of the receptive field population for the eye contralateral to the cortical hemisphere under consideration. The location of the receptive fields were estimated by an automated process where localized, flickering checkerboards patches, appeared at randomized locations within the screen. This experiment was run at the beginning of each imaging session to ensure the centering of receptive fields on the monitor.

Image processing: The image processing pipeline was the same as described in detail elsewhere [14]. Briefly, calcium images were aligned to correct for motion artifacts. Following motion stabilization, we used a Matlab graphical user interface (GUI) tool developed in our laboratory to define regions of interest corresponding to putative cell bodies manually. Following segmentation, we extracted signals by computing the mean of the calcium fluorescence within each region of interest and discounting the signals from the nearby neuropil. Spikes were then estimated using the algorithm described in [17], available at `https://github.com/darioringach/Vanilla`. The present results are based on the inferred spiking activity.

Generalized linear models: Poisson point process

Let $X \in \mathcal{Z}_+^{m,n_c}$ and $R \in \mathcal{R}^{m,n_r}$ be two sets of random matrices called the target variables and regressor variables respectively. Each column c of the X matrix, denoted $X_c; X_c \in \mathcal{Z}_+^m$ contains the spike train series of neuron c. Similarly, each column r of the R matrix, denoted $R_r; R_r \in \mathcal{R}^m$, is a time series that can be either the past spiking activity of an observed neuron or the history of one of the presented external stimuli.

We say that the set X is a Poisson Point Process if the conditional probability distribution (CPD) of each column X_c is of the form

$$X_c|R \sim Poisson\Big(f\Big(R\Big)\Big). \tag{1}$$

where f is a non-negative function, $\forall\, c = 1, ..., n_c$.

A Poisson Generalized linear model (GLM) is a special case of a general Poisson point process where

$$X_c|R \sim Poisson(\lambda(b_c + \sum_{i \in PA_c} R_i \times w_{ic})). \tag{2}$$

$\lambda : \mathbf{R} \to \mathbf{R}^+$ is a predefined nonlinearity, w_{ic} are the influence weights, b_c is the bias weight, and PA_c is the parent set of X_c, and is the subset of the regressors R that carry information on the behaviour of neuron c. In our case PA_c consists of visual stimuli and past neuron activity that directly affect the spiking rate of neuron c. The parameter w_{ic} represents the magnitude of that influence.

The overall goal of network learning is finding the set (PA_c, w_{ic}), for each neuron c, that best represents the recorded data (stimuli and neural activity).

In the following sections we will present the model in more detail and the chosen algorithm for selecting the relevant regressors PA_c and influence weights w_{ic}. Then we will introduce an experimental design method to select the stimuli that are more relevant in discovering the structure of the network. We want to emphasize that we are especially interested in correctly inferring the regressor set PA_c, this can be seen as a binary classification problem, since a given regressor either belongs or does not belong on PA_c for any given neuron c. This will directly translate into the decision of which sets of stimuli are relevant. This decision will be based on improving the PA_c classification performance.

Model description

The system under study consists of a set of n_c neurons $\{C\}$ and n_s stimuli or external inputs $\{S\}$, that can be used to perturb the neuron's activity.

To differentiate the influence of spiking activity and visual stimuli we will split the set of regressors R explicitly into spike activity $X_j(t), j = 1, .., n_c$, and stimuli activity $I_i(t), i = 1, .., n_s$.

To incorporate information about the past observations we further redefine the regressors as $\hat{X}_j(t) = \sum_{u=t-D_c^u}^{t-D_c^l} X_j(u) = (\mathbf{1}_{[t-D_c^l, t-D_c^u]} * X_j)(t), j = 1, .., n_c$, and $\hat{I}_i(t) = \sum_{u=t-D_s^u}^{t-D_s^l} I_i(u) = (\mathbf{1}_{[t-D_s^l, t-D_s^u]} * I_i)(t), i = 1, .., n_s$, which are the convolution of past spiking activity $(X_j, j = 1..n_c)$ and past stimuli activity $(I_i, i = 1..n_s)$ with the boxcar influence function up to delays D_c^u and D_s^u respectively.

To model the spiking train of neuron c, we also define the sets of relevant regressors of neuron c: $C'_c = \{PA_c \cap C\}$ and $S'_c = \{PA_c \cap S\}$.

Under these conditions, the spiking train of neuron c can be modeled as

$$\eta_c(t) = b_c + \langle W_c^T, \{\hat{X}_u(t)\}_{u \in C'_c} \rangle + \langle H_c^T, \{\hat{I}_v(t)\}_{v \in S'_c} \rangle \; \forall t, \tag{3}$$

$$\lambda_c(\eta_c(t)) = \frac{\log(1 + e^{\kappa \eta_c(t)})}{\kappa}, \tag{4}$$

$$X_c(t) | \{\hat{X}_u(t)\}_{u \in C'_c}, \{\hat{I}_v(t)\}_{v \in S'_c} \sim \text{Poisson}(\lambda_c(\eta_c(t)), \tag{5}$$

where $< . >$ denotes inner product. The parameters of interest are:

- $W_c \in R^{|C'_c|}$: Edge weights between parent neurons in C'_c and c. These weights collectively represent the inter-neuron connectivity matrix
$$W = \{w_{ic} : i, c = 1, ..., n_c\} = \begin{cases} 0 \leftrightarrow i \notin PA_c, \\ W_c(i) \leftrightarrow i \in PA_c. \end{cases}$$

- $H_c \in R^{|S'_c|}$: Edge weights between parent stimuli in S'_c and c. These weights collectively represent the direct stimuli response matrix
$$H = \{h_{jc} : j = 1, ..., n_s, \; c = 1, ..., n_c\} = \begin{cases} 0 \leftrightarrow j \notin PA_c, \\ H_c(j) \leftrightarrow j \in PA_c. \end{cases}$$

- $b_c \in R$: Bias of neuron c. This number encodes the base spiking rate of neuron c independently of the state of the other regressors.

Note that the model is time homogeneous, i.e., W_c, H_c and b_c do not depend on time. The motivation for using the nonlinearity function Eq. (4) instead of the canonical exponential nonlinearity is that Eq. (4) is approximately exponential for low values of η_c, but saturates to a linear dependency for large η_c. This allows the model to capture strong inhibitions while at the same time ensuring excitations in the model do not grow out of control. The nonlinearity proposed in Eq. (4) depends on a static calibration constant κ, we provide additional insight into what this parameter does in the context of model selection in the Appendix.

Parameter estimation

For each observed neuron c we want to find the regressor sets C'_c and S'_c that best explain the data without over-fitting.

We first define the likelihood function of the proposed model,

$$\begin{aligned}\mathcal{L}(\mathrm{X_c}|\{\hat{\mathrm{X}}\}_{\mathrm{C'_c}}, \{\hat{\mathrm{I}}\}_{\mathrm{S'_c}}, \mathrm{W_c}, \mathrm{H_c}) &= \log[\mathrm{f}(\mathrm{X_c}|\{\hat{\mathrm{X}}\}_{\mathrm{C'_c}}, \{\hat{\mathrm{I}}\}_{\mathrm{S'_c}}, \mathrm{W_c}, \mathrm{H_c})] \\ &= \sum_{t=1}^{m}[X_c(t)log[\lambda_c(\eta_c(t))] - \lambda_c(\eta_c(t))] + \mathcal{C}.\end{aligned} \tag{6}$$

where $\mathcal{C}$ is an additive constant A good regressor should provide a significant improvement in the model likelihood, and should have a tight confidence interval around its estimated edge weight. These notions are formalized using the Bayesian Information Criterion (BIC) [18] and the Wald test [19] respectively. A derivation of both in the context of our model is provided below.

To estimate the values for a set of parameters $\theta = \{W_c, H_c, b_c\}$ we utilize the standard Maximum Likelihood Estimation (MLE) framework. The MLE estimate is denoted as $\hat{\theta} = \{\hat{W}_c, \hat{H}_c, \hat{b}_c\}$ and is obtained as the solution of

$$\hat{\theta} = \{\hat{\mathrm{W}}_\mathrm{c}, \hat{\mathrm{H}}_\mathrm{c}, \hat{\mathrm{b}}_\mathrm{c}\} = \mathrm{argmax}_{\{\mathrm{W_c,H_c,b_c}\}} \sum_{\mathrm{t=1}}^{\mathrm{m}}[X_c(t)log[\lambda_c(\eta_c(t))] - \lambda_c(\eta_c(t))]. \tag{7}$$

Furthermore, the parameter obtained from this estimation asymptotically follows a Normal distribution around the true value θ_0 [20]. Under further regularity conditions [21], the variance of the estimator can be computed as shown in Eq. (8) (this will form the theoretical basis for the notion of tight confidence intervals),

$$\begin{aligned}\lim_{t\to\infty} P(|\hat{\theta} - \theta_0| > \epsilon) = 0 \ \forall \epsilon > 0, \\ \hat{\theta} \approx N[\theta_0, \{I(\theta_0)\}^{-1}]. \\ \mathcal{I}(\theta_0) = -E[\partial^2\mathcal{L}/\partial\theta_0\partial\theta'_0].\end{aligned} \tag{8}$$

The quantity $\mathcal{I}(\theta_0)$ is the Fisher information matrix [21]. Since we do not have access to the true parameter θ_0 we use the observed Fisher information ($\mathcal{J}(\hat{\theta}) = -\partial^2\mathcal{L}/\partial\hat{\theta}\partial\hat{\theta}'$) as an approximation. The observed Fisher information is sometimes referred to as the Hessian of the negative log-likelihood. The quantity $\mathcal{I}\{(\theta_0)\}^{-1}$ is a lower bound for the variance of any unbiased estimator, as stated by the Cramér-Rao lower bound [22, 23]. The observed Fisher information of the model can be found on Eq. (29) in the Appendix.

It is important to note that, in general, the Fisher information matrix is singular in the under-sampled regime (i.e., when there are more parameters than observations). This issue is later circumvented by the regressor selection process described below. This is accomplished naturally since the largest observed variance is bounded from below with a number that grows arbitrarily large for ill-conditioned Fisher Information matrices. A more detailed explanation is given in the Appendix.

This Gaussian assumption in the non-asymptotic regime is partly justified by the fact that the log-likelihood function is concave with respect to the model parameters, with a single, well defined global optimum. This property is also shared by the Gaussian distribution. From these observations we realize that both the finite sample and the asymptotic parameter distributions are log concave and have a well defined global optimum. This makes the Gaussian approximation in the finite sample case a more reasonable approximation. This approximation has also been used in [24] for similar reasons

The MLE estimators combined with the observed Fisher information provide a confidence interval for each parameter of interest. We start from the null hypothesis H_0 that the edge is irrelevant (zero value), and use the Wald test [19], to accept or reject this. The probability of parameter θ_j belonging to the null hypothesis can be computed as

$$\frac{\hat{\theta}_j^2}{[\hat{\mathcal{I}}(\hat{\theta})]^{-1}_{j,j}} \sim \mathcal{X}_1^2, \tag{9}$$

$$p_{H_0} = 1 - F_{\mathcal{X}_1^2}\left(\frac{\hat{\theta}_j^2}{[\hat{\mathcal{I}}(\hat{\theta})]^{-1}_{j,j}}\right), \tag{10}$$

where $\mathcal{X}_1^2$ is the chi-square distribution with one degree of freedom, $F_{X_1^2}(z)$ is the cumulative distribution function $\mathcal{X}_1^2$ evaluated at z, and p_{H_0} is the p-value associated with the null hypothesis.

At this point we have derived a measure of the probability of the parameter being different from zero.

To measure how informative a given regressor is we use the Bayesian information criteria (BIC). This quantity decreases with a higher likelihood $\hat{L}$ and increases with the number of parameters currently used in the model ($|\widehat{PA}_c|$), and the number of observations (m):

$$BIC = \ln(m) \times |\widehat{PA}_c| - 2 \times \hat{\mathcal{L}}. \tag{11}$$

When comparing two models, the model with the lowest BIC value is preferred. The quantity $\ln(m) \times |\widehat{PA}_c|$ penalizes model complexity, reducing the number of noisy weak connections.

This quantity is introduced to favor sparsely connected networks, since edges that have a low overall effect on the predicted likelihood of the model are ignored in favor of model sparsity.

For each observed neuron c, our objective is finding the parent set estimate $\widehat{PA}_c$ that minimizes the BIC, subject to a p-value restriction γ which forces the selected regressors to have a tight confidence interval:

$$\begin{aligned} \{\widehat{PA}_c\} &= \text{argmin}_{PA_c}\{BIC(\hat{\mathcal{L}}[\{PA_c\}]\}, \\ &\text{s.t. } max(p_{H_0}(PA_c)) \leq \gamma. \end{aligned} \tag{12}$$

This combined approach looks for a sparse model representation able to capture most of the information present in the observed spike trains using a series of parameters (edge weights) with well-defined values, with tight uncertainty intervals.

In the following section we will briefly explain how we approach this minimization.

Selecting relevant regressors

The optimization problem presented in Eq. (12) can be stated concisely as the problem of finding the set of regressors $\widehat{PA}_c$ that yields the best BIC score (Eq. 11) subject to a p-value restriction. This is done to ensure that the regression model has good prediction capabilities and generalizes well to non observed data points.

The problem as stated in Eq. (12) is combinatorial in nature and cannot be directly optimized. There is a rich literature on model selection using various search strategies and evaluation criteria [25–31]. Usual model evaluation criteria include BIC and Akaike Information Criterion (AIC) [32] among others, while search algorithms include stochastic search variable selection [31], forward model selection, backward elimination, and stepwise methods in general, among others [26].

For this particular problem, we decided to use a greedy elastic-forward subset selection algorithm; an extensive overview of subset selection strategies can be found in [26]. In addition, we take several randomly selected subsets from the training dataset, each containing a fraction ν of available samples, and we evaluate the BIC performance and p-value restriction for regressor candidates across all bootstrapping subsets to find candidates that are consistently relevant across subsets. In our experiments the fraction ν is set to 0.7. We show the performance of varying the ν parameter on simulated data in the Appendix.

The algorithm starts from $PA_c = \emptyset$, and iteratively includes regressors that improve the median BIC score across the random subsets, as well as the BIC score over the full dataset, while satisfying the p-value constraint.Since this algorithm is rerun for every new batch of data, this effectively means that edges can be "removed" between successive data acquisitions The algorithm is described in detail in the Appendix. The results section compares the performance of this algorithm against the LASSO method for variable selection [33].

Now that we have described the model and algorithm, we proceed to describe the active learning strategy.

Experimental design - Active learning

Our goal is to develop a method to select, at any time, the optimal action (or network perturbation) that is expected to yield the maximum information about its currently computed connectivity. For the purposes of this paper, our action set will consist of selecting which set of visual stimuli will be presented next.

We are interested in gathering samples from network connections (edges) that show a promising improvement in the likelihood of the model but have not yet been added to it, we refer to these edges as candidate edges. That means we want to improve the parent set estimate $\widehat{PA}_c$, $c = 1, ..., n_c$, and edge weight estimates w_{ic}, $i = 1, ..., n_r$, $c = 1, ..., n_c$, as in equations (12) and (7) respectively. The samples are collected by presenting stimuli that directly or indirectly generate activations in the parent nodes of the promising candidate edges.

To address this, we define a relevance score for each potential stimulus. This score considers the expected rate change of every regressor when presenting a given stimulus more frequently than the rest, weighted by the deviance statistic [34] of every edge associated with that regressor not currently present in the model. This in effect means that stimuli that directly trigger neurons associated with good candidate edges will be

presented more often than other stimuli during the next intervention. The exact formulation is presented next.

Defining a score for each stimuli

Define $\hat{W}^l$, $\hat{H}^l$ as the estimated adjacency and stimuli response matrices up to sample m_l, where l is an iteration counter. Our objective is to obtain a probability distribution vector $P^{l+1} = [p_1^{l+1}, ..., p_s^{l+1}, ..., p_{n_s}^{l+1}]$ for presenting each stimulus $s = 1, ..., n_s$ at intervention $l + 1$.

Future stimuli sequences will be sampled from this distribution. We introduce this intermediary stimuli distribution probability instead of simply applying the stimulus with the highest expected change in utility for two reasons. Firstly, we can sample several consecutive stimuli from this distribution, this way, we can apply a small batch of stimuli before recomputing the optimum stimulus, reducing the computational costs. Secondly, this approach is less greedy than exclusively presenting the stimulus that has the highest expected change in utility. This can be beneficial in settings where the inferred model has high uncertainty.

We start by computing the expected firing rate change on every neuron $c = 1, ..., n_c$ of presenting each stimulus $s = 1, ..., n_s$ more frequently. For each stimulus s, we define a surrogate probability distribution vector associated with it: $\hat{P}_s = \{\hat{p_s}_j : j = 1, ..., n_s\}$, where stimulus s has the highest occurrence probability. Given the previous observations, we want to predict the expected change in spiking rate of each neuron c when preferentially applying stimulus s using $\hat{P}_s$, as opposed to the baseline where all stimulus are presented equally: $\hat{P} = \{\frac{1}{n_s}\}$. Formally the probability vector that favors stimulus s ($\hat{P}_s$) is defined as

$$\begin{aligned} \hat{P}_s &= \{\hat{p_s}_j : j = 1, ..., n_s\}, \\ \hat{p_s}_j &= (1 - \beta) \times \delta_{s-j} + \beta \times \frac{1}{n_s}, \end{aligned} \tag{13}$$

where $\hat{p_s}_j$ is the j-th element of the probability vector $\hat{P}_s$ and corresponds to the probability of presenting stimulus j in the surrogate probability distribution vector associated with s, β is a smoothing constant that satisfies $0 \leq \beta \leq 1$ and controls the overall probability of other stimuli appearing, and n_s is the number of available stimuli. We will define $\widetilde{ERC}_{s,c}$ as the expected rate change of stimulus s on neuron c.

The estimated firing rate change on neuron c caused by stimulus s at iteration $l + 1$ ($\widetilde{ERC}_{s,c}[l + 1]$) is presented in Eq. (14) next. This compares the expected firing rate of using a stimuli distribution $\hat{P}_s$ on neuron c when compared to using the baseline (uniform) distribution $\hat{P}$. The quantity $\widetilde{ERC}_{s,c}[l + 1]$ shows the rate increase of preferentially presenting stimuli s on the spiking rate of neuron c (λ_c) according to our previous m_l observations,

$$\begin{aligned} \widetilde{ERC}_{s,c}[l + 1] &= \frac{E_{X,S\sim\hat{P}_s}[X_c|\hat{W}^l, \hat{H}^l]}{E_{X,S\sim\hat{P}}[X_c|\hat{W}^l, \hat{H}^l]} \\ &= \frac{E_{X,S\sim\hat{P}_s}[\lambda_c|\hat{W}^l, \hat{H}^l]}{E_{X,S\sim\hat{P}}[\lambda_c|\hat{W}^l, \hat{H}^l]} \end{aligned} \tag{14}$$

Next we compute the deviance statistic [34] for every possible outbound edge of neuron c that did not satisfy the parameter selection criteria in Eq. (12) and therefore

$c \notin \widehat{PA}_{c_i}$ for some neuron c_i. This is twice the log-likelihood difference (noted as $\hat{D}_{c,c_i}[m_l]$) between the network model up to sample m_l (where $c \notin \widehat{PA}_{c_i}$) and a network model where c is included as a parent of c_i:

$$\hat{D}_{c,c_i}[m_l] = 2(\hat{L}[\hat{X}_{c_i}(m_l)|(\widehat{PA}_{c_i} \cup c)(m_l) - \hat{L}[\hat{X}_{c_i}(m_l)|\widehat{PA}_{c_i}(m_l)])]. \tag{15}$$

To compute the deviance statistic to any given edge from neuron c to neuron c_i, we need to recompute the Maximum Likelihood (ML) estimate under the new regressor subset $\widehat{PA}_{c_i} \cup c$. Fortunately, this ML estimate is fast to compute, the computation can be made faster by initializing the ML minimizer to the previously obtained regressor values.

Note that Eq. (15) is always non-positive, a highly negative value indicates a strong possibility of neuron c influencing neuron c_i. By acquiring more samples from this interaction (samples where the candidate parent node is active), we can either disprove this notion, or gather enough evidence to add this edge into the regressor set (by satisfying the BIC and p-value criteria for adding an edge to the model).

We therefore define the score of stimulus s associated with inter neuron edges W as

$$\widehat{SC}_{s,W}[l+1] = \sum_{c \in C} [\widetilde{ERC}_{s,c}[l+1] \times \sum_{c_i : c \notin \widehat{PA}_{c_i}(m_l)} \frac{\hat{D}_{c,c_i}[m_l]}{|\{c_i : c \notin \widehat{PA}_{c_i}(m_l)\}|}]. \tag{16}$$

We have so far assigned a score that considers the deviance impact of preferentially applying a stimulus s on the inter neuron connectivity matrix W. It is important to note that the set $\{c_i : c \notin \widehat{PA}_{c_i}(m_l)\}$ refers to the set of cells that up to sample m_l were not included as children of cell c, and therefore are considered as candidate edges for the purpose of the score. In this score, the first summation considers the impact of stimulus s over each cell c multiplied by the second summation, which is the mean log-likelihood difference over all the cells that were not considered as children of neuron c up to sample m_l.

In a similar fashion, we also need to consider the effect of prioritizing any given stimulus s on the stimuli response matrix H. In this case, the expected rate change of prioritizing stimulus s over stimulus s_i is the quotient of the s_i entry of the stimuli probability distribution vector associated with s ($\hat{P}_s$) over the baseline (uniform) probability distribution vector ($\hat{P}$),

$$\begin{aligned} \widetilde{ERC}_{s,s_i}[l+1] &= \frac{E_{X,S \sim \hat{P}_s}[I_{s_i}|\hat{W}^l, \hat{H}^l]}{E_{X,S \sim \hat{P}}[I_{s_i}|\hat{W}^l, \hat{H}^l]} \\ &= \frac{\hat{p}_{s,s_i}}{\frac{1}{n_s}} \\ &= (1-\beta) \times n_s \times \delta_{s-s_i} + \beta. \end{aligned} \tag{17}$$

The deviance score of the output edges of s that did not satisfy the parameter selection criteria in Eq. (12) and the score of stimulus s associated with the stimuli response matrix H can be analogously defined:

$$\hat{D}_{s_i,c_i}[m_l] = 2(\hat{L}[\hat{X}_{c_i}(m_l)|(\widehat{PA}_{c_i} \cup s_i)(m_l)] - \hat{L}[\hat{X}_{c_i}(m_l)|\widehat{PA}_{c_i}(m_l)]), \tag{18}$$

$$\widehat{SC}_{s,H}[l+1] = \sum_{s_i \in S} \left\{ \widetilde{ERC}_{s,s_i}[l+1] \times \sum_{c_i : s_i \notin \widehat{PA}_{c_i}(m_l)} \frac{\hat{D}_{s_i,c_i}[m_l]}{|\{c_i : s_i \notin \widehat{PA}_{c_i}(m_l)\}|} \right\}. \tag{19}$$

Finally, the combined score given to stimulus s is

$$\widehat{SC}_s[l+1] \quad = \widehat{SC}_{s,W}[l+1] + \widehat{SC}_{s,H}[l+1]. \tag{20}$$

At this point we have a score for each stimulus s that is able to capture how informative this stimulus might be based on the expected rate change it has on edges that are not included in the model so far. The next step is mapping these scores into a probability vector. For that we first compute the z-score of each stimulus; this is done as a normalization step of the score values, and allows the detection of outlying stimuli. The z-scores are then converted into a probability vector with the use of the well known softmax function. To avoid giving unnecessarily small or large probabilities to any given stimulus, we truncate the computed z-score into the $[-2, 2]$ range.

The formulation is as follows:

$$Z_s[l+1] \quad = \frac{\widehat{SC}_s[l+1] - mean(\{\widehat{SC}_s[l+1]\}_{s=1}^{n_s})}{std(\{\widehat{SC}_s[l+1]\}_{s=1}^{n_s})}, \tag{21}$$

$$Z_s^*[l+1] \quad = max(min(Z_s[l+1], 2), -2), \tag{22}$$

and the probability distribution vector for presenting each stimulus at intervention $l+1$ ends up being:

$$P^{l+1} = \left\{ \frac{\exp Z_s^*[l+1]}{\sum_{j=1}^{n_s} \exp Z_j^*[l+1]};\ s = 1, ..., n_s \right\}. \tag{23}$$

The use of the z-score as a normalization step allows the algorithm to dynamically pick up on the "relative quality" of the stimulation actions, and the truncation of the score provides a limit on how frequently or infrequently any given stimulus can be shown.

Active learning

The active learning strategy consists of iteratively evaluating the current network model using Algorithm 1 in the Appendix, then using equations (20) and (23) to compute the stimuli distribution probabilities for the next time interval. The full algorithm is described in Algorithm 2 in the Appendix.

Results

Simulated data

In order to validate the method before applying it to real datasets, we generated a number of artificial datasets where the connectivity is known.

Network topology was simulated using the small-world Watts-Strogatz model [35]. This type of network architecture has been used to model functional cortical connectivity in cats and macaques [36, 37], and has been theorized to be of use in understanding human functional connectivity [38].

We defined two separate networks SW1CL and SW3CL, network SW1CL is a single small-world cluster network with 18 neurons, while network SW3CL has three separate small-world clusters, each cluster has 18 neurons. All clusters have an average connectivity degree of 0.03. The increase in spiking rate for neuron to neuron edges in network SW1CL were drawn from a normal distribution $N(0.05, 0.005)$, the edge

weights in the adjacency matrix were set accordingly. Similarly, the increase in spiking rate for neuron to neuron edges for the three clusters in network SW3CL were drawn from normal distributions $N(0.075, 0.005)$, $N(0.05, 0.005)$ and $N(0.035, 0.005)$ respectively. Thirty percent of inter-neuron edges were made inhibitory.

These simulated networks were presented with 30 possible excitatory stimuli, most of which were designed to have no effect on the network. This was done to test that the active learning algorithm has the ability of navigating through confounders. The increase in spiking rate fro stimuli to neuron connections were drawn from the normal distribution $N(0.10, 0.014)$. Fig. 2 shows the connectivity matrix W and stimuli response matrix H for networks SW1CL and SW3CL. Boxcar influence functions as defined for Eq. (3) were set to $D_s^l = D_c^l = 5$ and $D_s^u = D_c^u = 2$ ($\mathbf{1}_{[t-2,t-5]}(t)$). These values were selected so that the average spiking rate of the simulated neurons were similar to the ones obtained in real data.

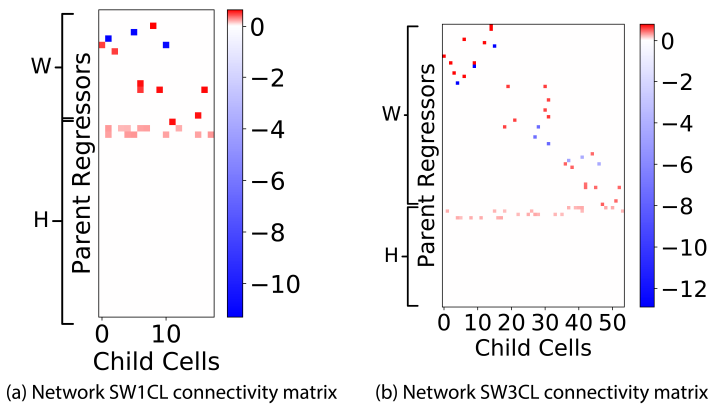


Fig 2. Adjacency matrices for networks SW1CL and SW3CL. Red entries in the adjacency matrices denote an excitatory relation between the regressor and the child neuron, while blue entries denote inhibitory connections. The block diagonal structure present in the W matrix for network SW3CL evidences the three cluster structure of the network. These connectivity matrices were computed to generate the required spiking rate change for model parameter $\kappa = 10$. On both networks, we can observe the large number of stimuli that have no effect on the network.

In the following sections we will present two experiments. The first experiment will show the performance difference in regressor selection when using the proposed Algorithm 1 compared against the Lasso method [33]. The second experiment will compare the performance in regressor selection when stimuli are chosen according to active learning Algorithm 2 versus random stimuli selection.

Algorithmic performance will be presented based on precision, recall, and F_1 metrics:

$$\begin{aligned} \text{precision} &= \frac{\sum_{c\in C}|\widehat{PA}_c \cap PA_c|}{\sum_{c\in C}|\widehat{PA}_c|}, \\ \text{recall} &= \frac{\sum_{c\in C}|\widehat{PA}_c \cap PA_c|}{\sum_{c\in C}|PA_c|}, \\ F_1 &= 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \end{aligned} \tag{24}$$

$\widehat{PA}_c$ is again the recovered set of regressors, and PA_c is the true set of regressor edges for neuron c.

Relevant regressors

We first checked the performance of the regressor selection methods on networks SW1CL and SW3CL. We compare the performance of the elastic-forward BIC selection Algorithm 1 with bootstrapping versus standard Lasso [33] regression using the pyglmnet implementation [39].

Stimuli were sampled uniformly with replacement, each stimulus was presented for 4 consecutive frames. Spiking trains for the simulations were sampled from a Poisson random process with a spiking rate corresponding to the ground truth model from Eq. (5).

The l_1 regularization parameter for the Lasso method was selected using an oracle to provide the best possible F_1 score. This method was selected for comparison because it is one of the most common approaches to variable selection. The modified log-likelihood function used for the Lasso method was:

$$\mathcal{L}_{\mathrm{l_1}}(\mathrm{X_c}|\{\hat{\mathrm{X}}\}_{\mathrm{C'_c}}, \{\hat{\mathrm{I}}\}_{\mathrm{S'_c}}, \mathrm{W_c}, \mathrm{H_c}) \propto \sum_{t=1}^{m}[X_c(t)log[\lambda_c(\eta_c(t))] - \lambda_c(\eta_c(t))] - l_1||W_c||_1 - l_1||H_c||_1. \tag{25}$$

Fig. 3 shows the results; 10 independent trials were used to provide confidence intervals for the metrics.

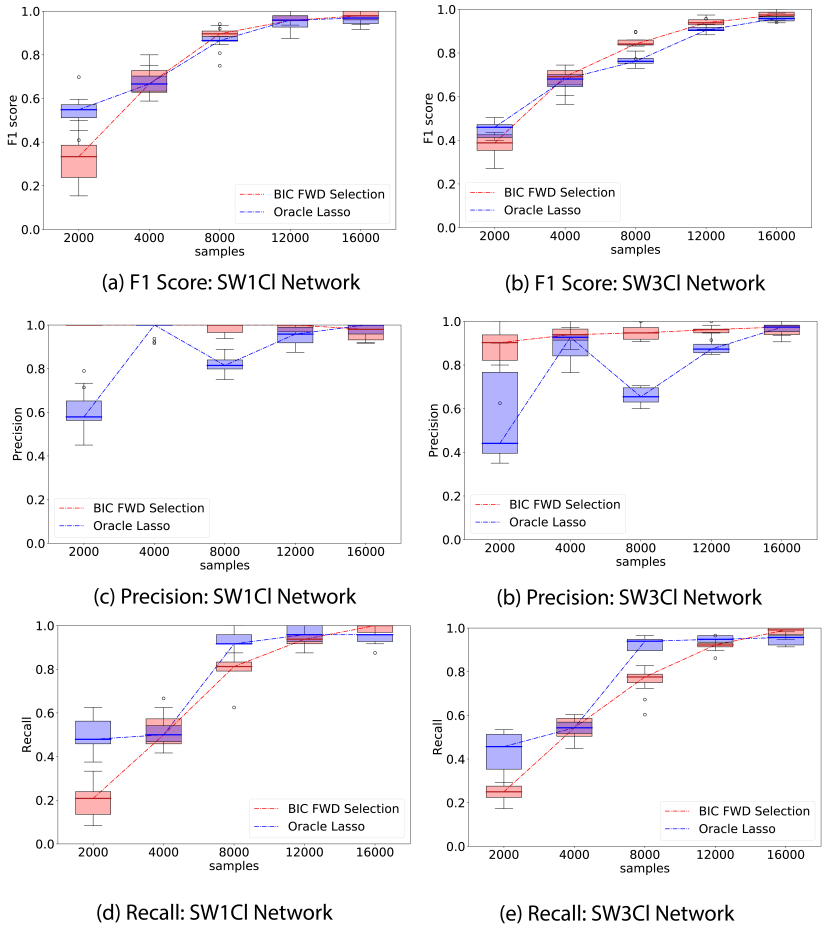


Fig 3. Whisker plot of performance indicators as a function of number of samples; elastic-forward BIC selection is shown in red, oracle lasso in blue. The whisker plot is obtained from 10 independent trials. External stimuli are drawn randomly from a uniform distribution; network SW1CL has a total of 24 non-zero parameters out 864 potential regressors, while network SW3CL has a total of 58 non-zero parameters out of 4536 potential regressors. The elastic-forward BIC selection outperforms the oracle lasso for larger sample sizes. This performance improvement is more noticeable in the SW3CL network, where edge weights are more diverse

Fig. 3 shows that the elastic-forward BIC method described in Algorithm 1 outperforms lasso for larger sample sizes, even when the l_1 regularization parameter is selected using an oracle. The improvement is more noticeable for network SW3Cl which has diverse edge weights. In all tested cases, the precision metric in edge recovery was better for elastic-forward BIC subset selection.

Active learning: Stimuli selection

We now evaluate the performance of the proposed active learning method, Algorithm 2. We compare it against uniformly sampling from all 30 possible stimuli.

Both strategies start from the same initial 500 samples, and each intervention adds an additional 500 samples. At the beginning of each intervention step, we compute the best network estimate so far using Algorithm 1, and show the performance in recovering

the set of regressor edges PA_c using the F_1, precision and recall metrics. At this stage, the active learning strategy described in Algorithm 2 recomputes the stimuli probability distribution to apply for the following 500 samples. Active learning parameter β was set to $\beta = 1/4$.

Fig. 4 compares the performance of uniform (Random) stimuli sampling and active learning (AL) sampling, while Fig. 5 shows the performance difference only on the entries of the stimuli response matrix H. Experiments were repeated 10 times to provide confidence intervals for the metrics.

Active learning outperforms random sampling by a large margin, the inter-quartile ranges for random sampling and active learning do not overlap over a significant sample count. As expected, both strategies converge for large sample sizes, but the process is sped-up by selecting the correct set of stimuli. The most noticeable performance difference is obtained when recovering the stimuli response edges H, since these are the edges we have direct influence on.

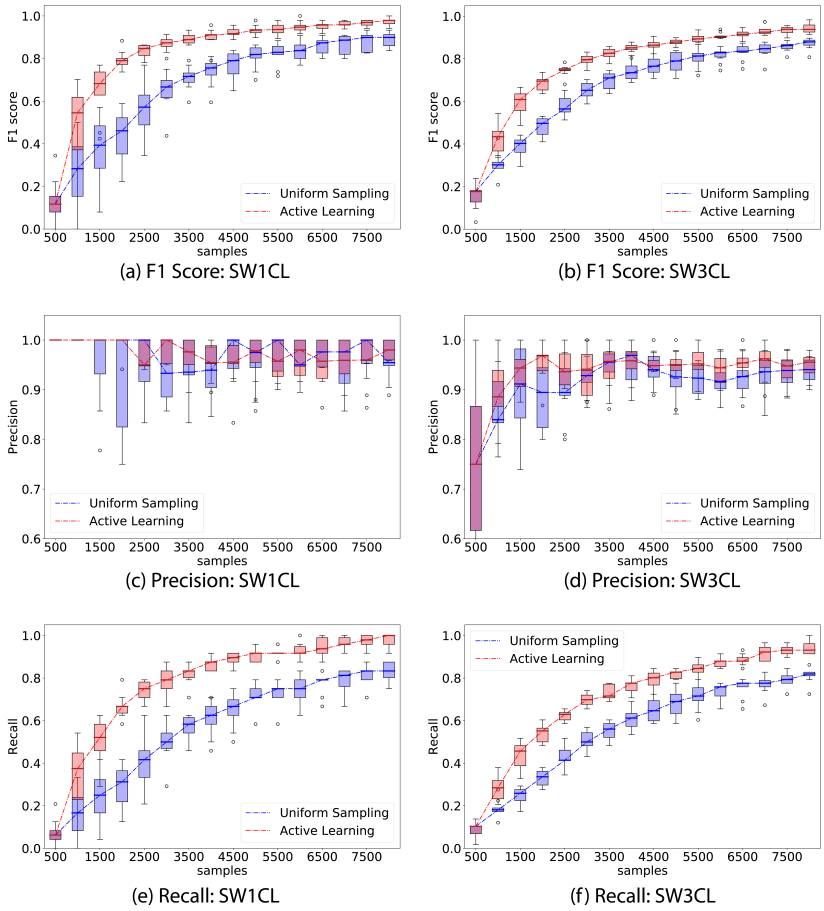


Fig 4. Comparison of performance between the proposed active learning (AL) method versus uniformly sampling (Random) from all stimuli. The experiment consisted of 500 sample interventions, with an initial 500 sample observation. Whisker plots are obtained from 10 independent trials. Left column show F_1, precision, and recall performance on network SW1CL and right column on network SW3CL.

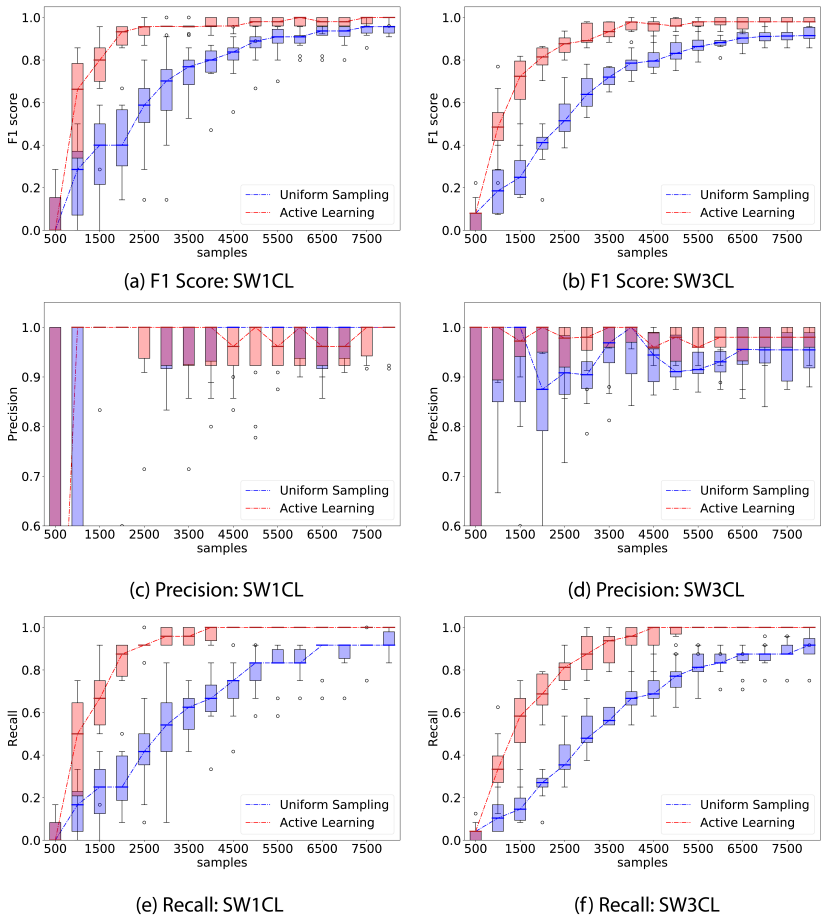


Fig 5. Comparison of performance between the proposed active learning (AL) method versus uniformly sampling (Random) from all stimuli only over the direct stimuli response matrices H. The experiment consisted of 500 sample interventions, with an initial 500 sample observation. Whisker plots are obtained from 10 independent trials. Left column shows F_1, precision, and recall performance on network SW1CL and right on network SW3CL. On average, computing the active learning intervention step took $20s \pm 9s$ on network SW1CL, and $98s \pm 50s$ on network SW3CL.

Fig. 6 shows a comparison between the elastic-forward BIC method described in Algorithm 1 and oracle lasso when applied to samples drawn from the recommended active learning distribution proposed in Algorithm 2. Both regressor selection methods perform similar under these conditions.

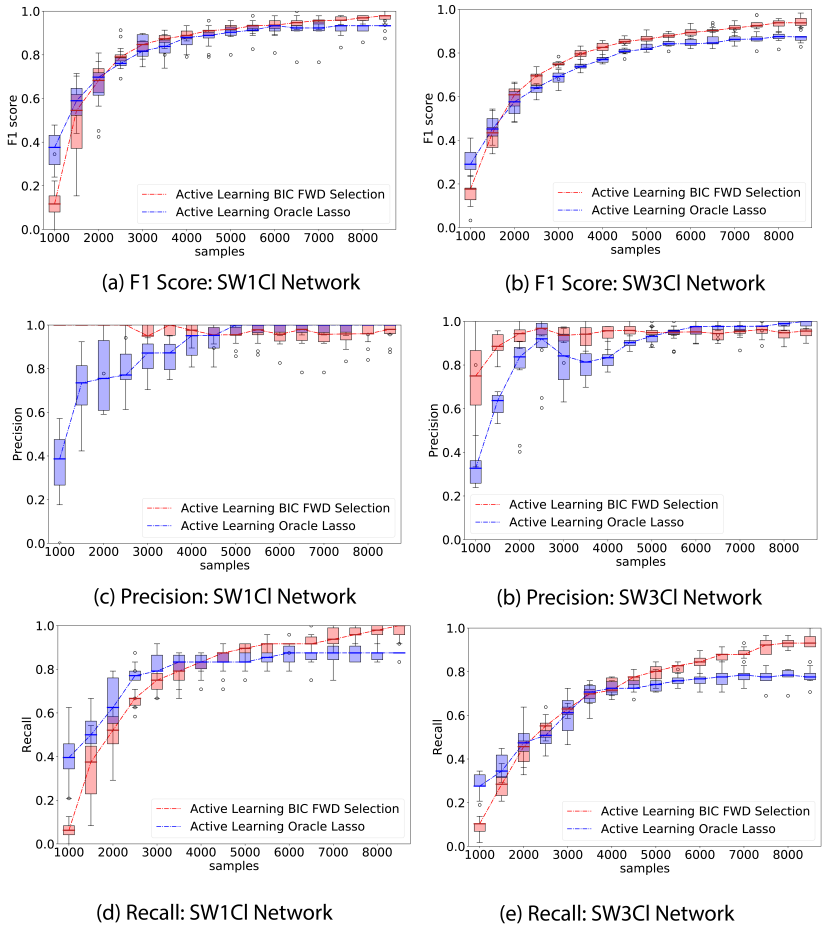


Fig 6. Whisker plot of performance indicators as a function of number of samples; elastic-forward BIC selection is shown in red, oracle lasso in blue. The whisker plot is obtained from 10 independent trials. External stimuli are drawn from the recommended Active Learning distribution; network SW1CL has a total of 24 non-zero parameters out 864 potential regressors, while network SW3CL has a total of 58 non-zero parameters out of 4536 potential regressors. The elastic-forward BIC selection still outperforms the oracle lasso for larger sample sizes. Both methods show a performance gain on the samples drawn from the Active Learning distribution when compared to their uniform sampling counterparts

Fig. 7 shows a visual representation of the edge discovering process over network SW1CL using active learning versus random sampling. Fig. 8 shows the difference between ground truth and the active learning and random sampling estimates as a function of interventions. Here we can clearly see that H edges are quickly recovered using active learning, while W edges show a slower improvement that is network dependent.

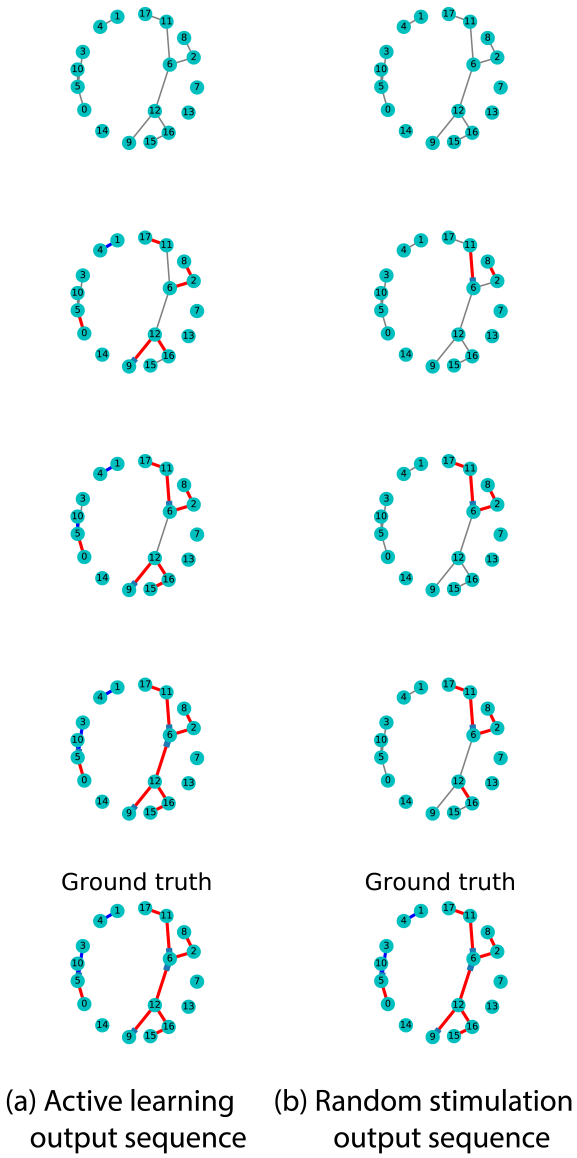


Fig 7. Comparison of the edge discovering process for network SW1CL using active learning versus random stimulation. Rows show inferred connections over a simulated cluster as a function of samples. Red and blue edges show correctly detected excitatory and inhibitory connections respectively, while grey edges show connections as not yet detected. Left column shows detected edges as a function of time when the active learning stimulation policy is used, right column shows the same cluster with a completely random stimulation policy. Rightmost cluster shows the ground truth. This example shows a clear advantage in edge recovery when using active learning compared to random stimulation.

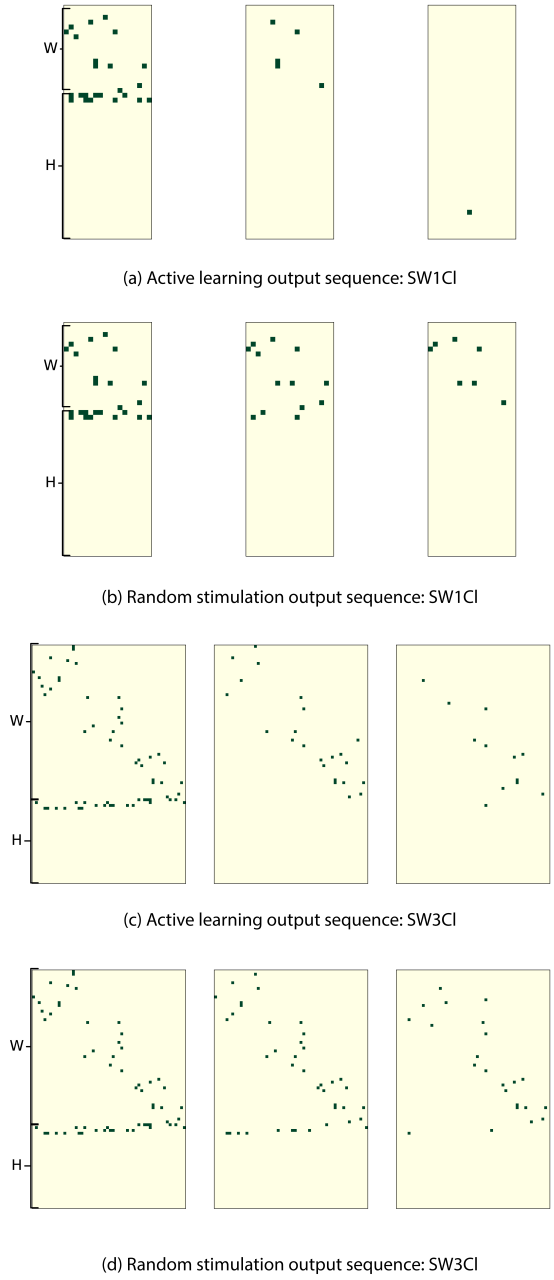


Fig 8. Comparison of the edge discovering process for networks SW1CL and SW3CL using active learning versus random stimulation. Rows show misclassified edges in the adjacency matrices W and H as a function of samples. Rows 1 and 3 show misclassified edges as a function of time when the active learning stimulation policy is used, while rows 2 and 4 show the same network probed with a random stimulation policy. The misclassified edge matrix under active learning quickly becomes sparse as the number of misclassifications goes to zero, random stimulation produces the same results but in a longer time frame.

Real data

We worked with two datasets: lt3-000-002 and lt3-000-003, hereafter called datasets 1 and 2, containing a population of 57 and 63 neurons respectively. The presented visual stimuli consisted of sinusoidal gratings defined using Hartley basis functions of the form:

$$H(k_x, k_y) = A \times \text{cas}([k_x x + k_y y]\frac{2\pi}{M}), \tag{26}$$

where the function $cas(x)$ is $cos(x) + sin(x)$, $A = \pm 1$, (x, y) are the pixel coordinates in the monitor, M is the image size, and k_x, $k_y = \{-12, ..., 12\}$ are the frequency components. Parameters A, k_x, k_y were uniformly sampled from all possible values, and each stimuli persisted for 4 frames.

To prevent excessive data fragmentation, the stimuli basis (A, k_x, k_y) was encoded into an (r, ϕ) pair using Eq. (27) with the r and ϕ parameters discretized into 7 values each,

$$\begin{aligned} r &= \sqrt{k_x^2 + k_y^2}, \\ \phi &= \text{atan}(k_y, k_x). \end{aligned} \tag{27}$$

For both spike and stimuli regressors, we used $\mathbf{1}_{[t-2,t-7]}(t)$ as the influence function as defined for Eq. (3) ($D_s^l = D_c^l = 7$ and $D_s^u = D_c^u = 2$).

$7,000$ samples of each dataset were used for training, and 2000 samples were reserved for model validation. Models were computed under two conditions, the first model used all available regressors (full model), while the second model was restricted to self regression coefficients and direct stimuli to neuron connectivity (AR model).

We first show the predictive power of both models when evaluated over the validation samples, that is to say, we evaluate the log likelihood (Eq. (6)) of both models over the test samples.

We then utilize the model that we obtained from all samples in the dataset as a template for simulations ("ground truth"), and compare the simulated performance between the proposed active learning Algorithm 2 and random sampling.

We go on to show that observed neurons tend to respond more to low frequency stimuli. We then show the recovered adjacency matrix (networks) for both datasets, and spiking trains time series for neurons belonging to the largest cliques in the network.

Recovered models

We now show the recovered connectivity matrices W and stimuli response matrices H for both datasets for the full model and the AR model.

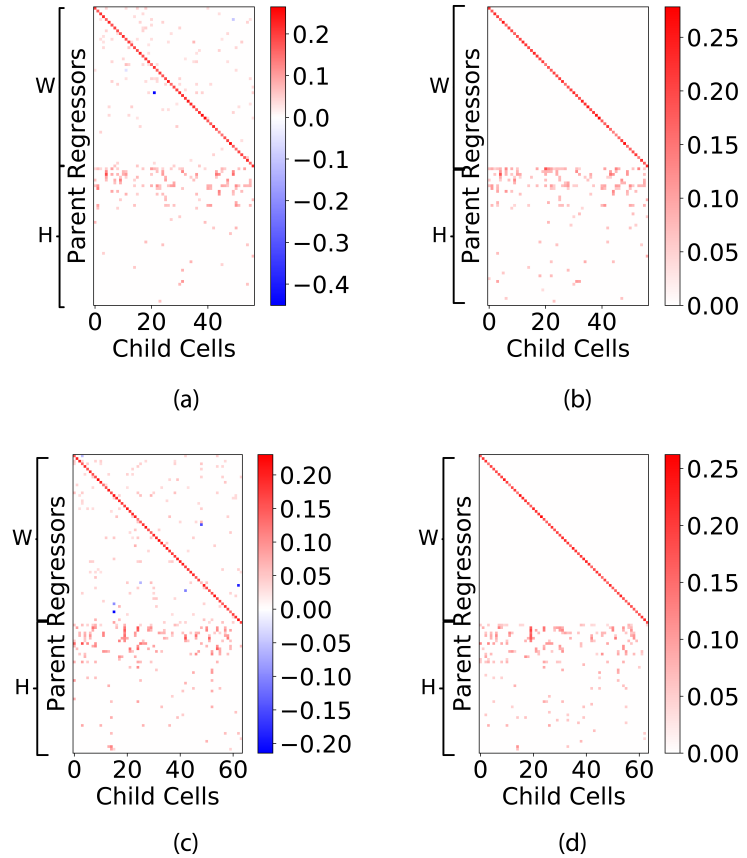


Fig 9. Recovered adjacency matrices for datasets 1 and 2. Top and bottom rows show the recovered adjacency matrices for datasets 1 and 2 respectively. Columns from left to right show the full model and the AR model respectively. Inhibitory connections are shown in blue, and excitatory connections are shown in red. We can observe that self regression coefficients are always added to the model. We also observe that the overall sparsity of the recovered network is consistent across datasets , and that the first few rows of the H matrices show a heavy concentration of excitatory connections. These rows correspond to low spatial frequency (r) values in the Hartley basis functions (Eq.(26)).

The recovered inter-neuron connectivity matrix W for both datasets was 93% sparse. Both datasets also show a large number of stimuli connections in the H matrix corresponding to low spatial frequency stimulation values (r), and consistently selected the self regression coefficient as an important regressor. This was consistent for both regression models.

Out of sample prediction power

To test the prediction power of the regressor model, we first evaluate the log likelihood of the full model and AR model over the test samples. We compute η_c and λ_c for each neuron on the test samples using the observed spike trains and visual stimuli as presented in equations (3) and (4). We then evaluate the log likelihood according to Eq. (6). The results are shown in Fig. 10.

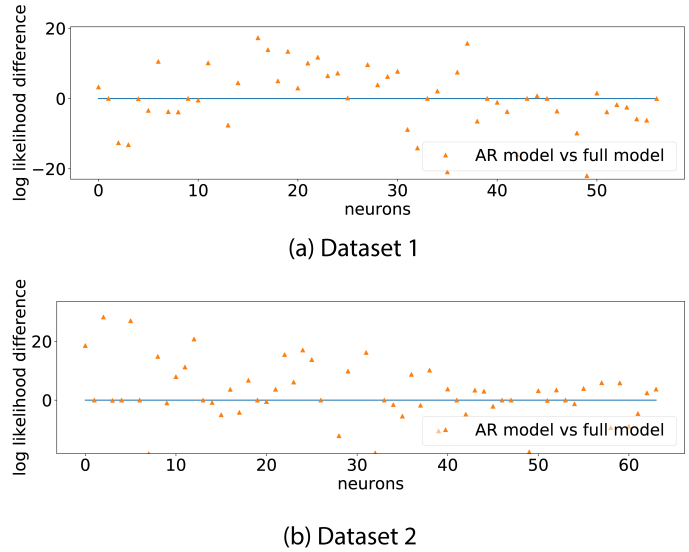


Fig 10. Log likelihood difference of full model and AR model over the test samples. The graph shows that, on average, spiking rate predictions are better on the full model than on the auto-regressive model. This grounds the idea of neuron to neuron interaction as being predictive of neuron behaviour.

Fig. 10 shows that, for many neurons, the full model generalizes well to out of sample (unobserved) data, when compared to the AR model. This shows that inter-neuron connectivity is predictive of spiking rates.

Long range prediction is also possible using the recovered model parameters $(\hat{W}, \hat{H}, \hat{b})$ and the visual stimuli sequence to be presented $(\{I_k(t)\})$. Instead of using the observed past spike trains $X_j(t)$ as regressors, we use the expected spiking rate $\lambda_j(t)$. This experiment iteratively computes the expected spiking rate for each neuron in the network using a fully observed external stimulation sequence and the past computed expected spiking rate. It is important to note that here we are computing the entire behaviour of the system given a stimuli sequence.

Formally, we define the long range spiking rate as

$$\lambda_c^{\mathrm{lr}}(t) = E[X_c \mid \{\hat{\lambda}_j^{\mathrm{lr}}(t)\}, \{\hat{I}_k(t)\}, \hat{W}, \hat{H}, \hat{b}], \tag{28}$$

where $\lambda_c^{\mathrm{lr}}(t)$ is the expected spiking rate at time t for neuron c, $\{\hat{\lambda}_j^{\mathrm{lr}}(t)\} = \{\sum_{t-D_c^u}^{t-D_c^l} \lambda_j^{\mathrm{lr}}(t)\}$ is the set of past expected spiking rates as seen through the boxcar influence function, and $\{\hat{I}_k(t)\} = \{\sum_{t-D_s^u}^{t-D_s^l} I_k(t)\}$ is the applied visual stimuli as seen through the boxcar influence function.

Here we compare the difference between the log likelihood when simulating the 2000 test samples with the stimuli that were presented versus a simulation of the system under a random equally likely stimuli selection.

The log likelihood of the sequences is computed against the observed spike trains for each neuron, experiments are repeated over 20 trials to provide error estimates. Results are shown in Fig. 11.

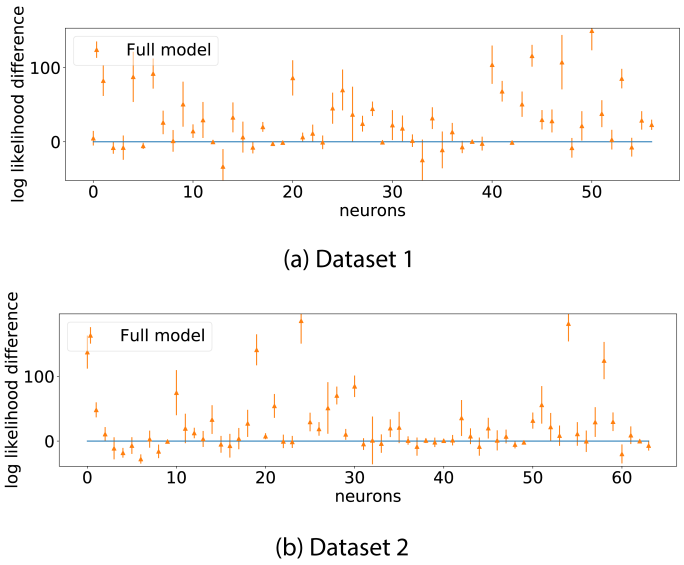


Fig 11. Difference in log likelihood of forecasted sequences, real stimulation sequence versus randomized stimulation sequences. Error bars represent one standard deviation over 20 trials.

Iterated simulations like these integrate the influence of stimuli to neuron connectivity and neuron to neuron connectivity. The large majority of neurons for both datasets showed a likelihood increase when presented with the true stimulation sequence, showing that the recovered models really capture interactions between regressors and spiking rates.

Active learning on real data

In lieu of validating the proposed active learning framework on live animals, we utilize the recovered full model networks from Fig. 9 as ground truth. For this experiment, we simulate data using the recovered inter-neuron and stimuli response matrices (W, H) as ground truth. A separate model is trained on data drawn from this new simulation. We compare the performance of the network inference algorithm when samples are drawn uniformly from all 49 possible stimuli against samples drawn from the inferred active learning distributions.

Both strategies start from the same initial $1,000$ samples, and each intervention adds an additional 500 samples. As a reminder, the ground truth network was recovered from $9,000$ samples. At the beginning of each intervention step, we compute the best network estimate so far using Algorithm 1, and show the performance in recovering the set of regressor edges PA_c using the F_1, precision and recall metrics as defined in Eq. (24). At this stage, the active learning strategy recomputes the stimuli probability distribution to apply for the following samples. These results are shown in Fig. 12. Additionally, Fig. 13 shows the stimuli probability distribution (P^{l+1}) obtained from Algorithm 1, averaged over all realizations.

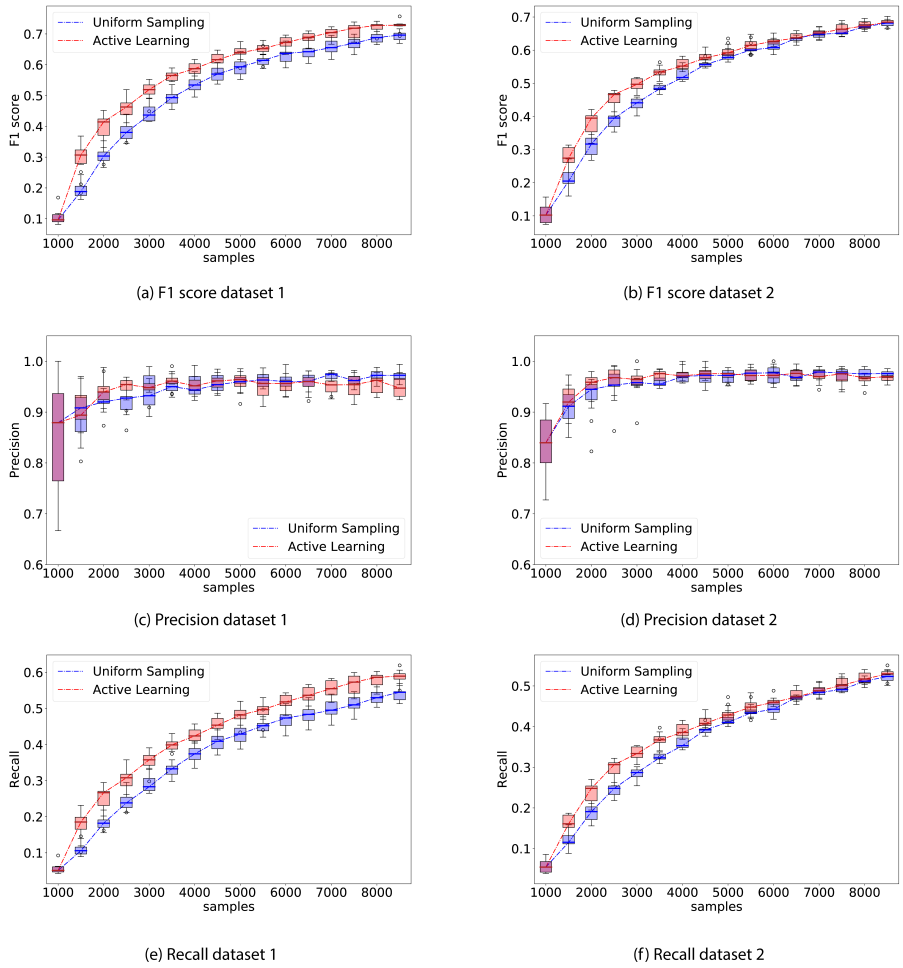
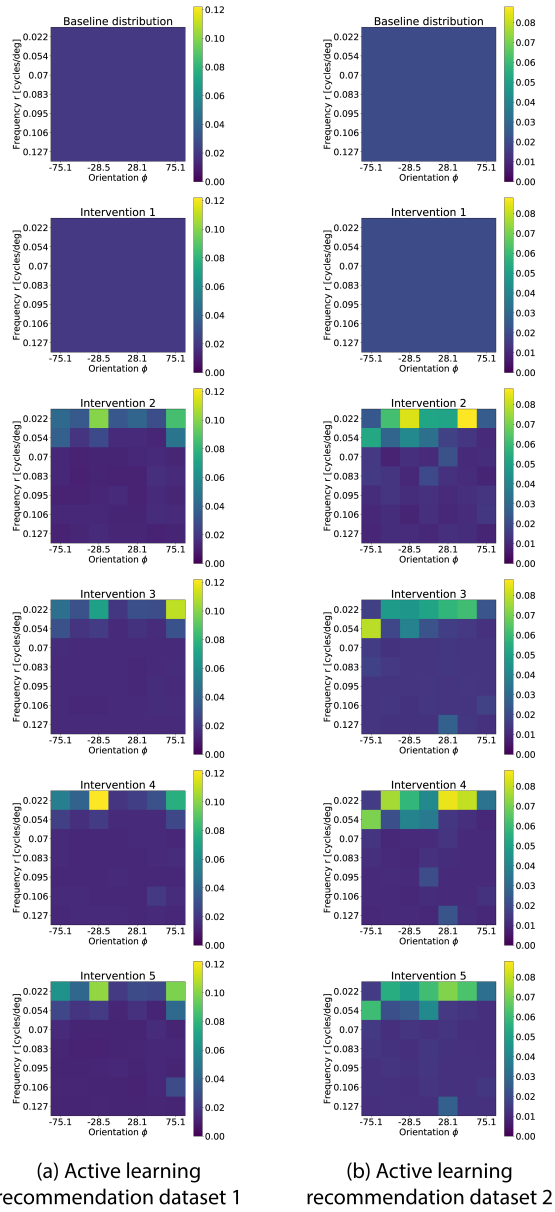


Fig 12. Comparison of performance between the proposed active learning (AL) method versus uniformly sampling from all stimuli. The experiment consisted of 500 sample interventions, with an initial $1,000$ sample observation. Whisker plot is obtained from 10 independent trials. Left column shows F_1, precision, and recall performance on network recovered from dataset 1 and right column shows F_1, precision, and recall performance on network dataset 2. On average, computing the Active learning intervention step took $108s \pm 60s$ on dataset 1, and $86s \pm 46s$ on dataset 2.



(a) Active learning recommendation dataset 1

(b) Active learning recommendation dataset 2

Fig 13. Distribution of recommended stimuli (P^{l+1}) averaged across all realizations as a function of the number of interventions. Initially, the distribution is uniform (top). The experiment consisted of 500 sample interventions, with an initial $1,000$ sample observation. Left column shows the stimuli probability distribution history for dataset 1, while right column shows the distribution for dataset 2.

The F_1 score of the active learning experiment is consistently better than random stimuli selection. While the performance gain is not large for these networks, the result spreads are tight and consistent; there is thereby no reason for not using the active learning strategy over random stimulation. We also observe that the AL algorithm preferentially presents low frequency stimuli, even though no explicit variable in the AL algorithm distinguishes between low and high frequency stimuli. In the following section we will show this is a reasonable result, since we found that neurons in these datasets tend to respond considerably more to low frequency stimuli.

Network analysis

Now that we showed the performance of the proposed methods we will present some observations over the recovered biological models.

We first analyze the obtained input response matrices H for both datasets. Fig. 14 shows the percentage of neurons in each dataset that directly respond to each stimulation pattern. We can conclude that low frequency stimulation patterns have an out-sized proportion of directly responding neurons when compared to higher frequency patterns.

For each dataset, we then examine the adjacency matrices W and extract two of the largest cliques in the network. Fig. 15 shows these cliques, and the spike trains of all neurons in the clique.

We additionally count the occurrence rate of motif triplets in the recovered networks, and do a simple hypothesis test to check if these motifs could have arisen from a small-world network topology, Fig. 16 show these results. The high p-values observed in Fig. 16 show that the recovered network motifs are at least compatible with the small-world topology hypothesis. We also compared the distribution of number of children and parents per cell between the inferred networks and a small-world network ensemble. In Fig. 17 we can see that the inferred networks are compatible with a small-world topology in terms of number children and parents per neuron. Finally, we count the percentage of cells involved in multiple cliques, and further show the percentage of cells involved in multiple cliques of a set size. The percentage counts also appear to be within the expected counts of a small-world network, this is shown in Fig. 18. This is in accordance with other observations like the ones in [36–38].

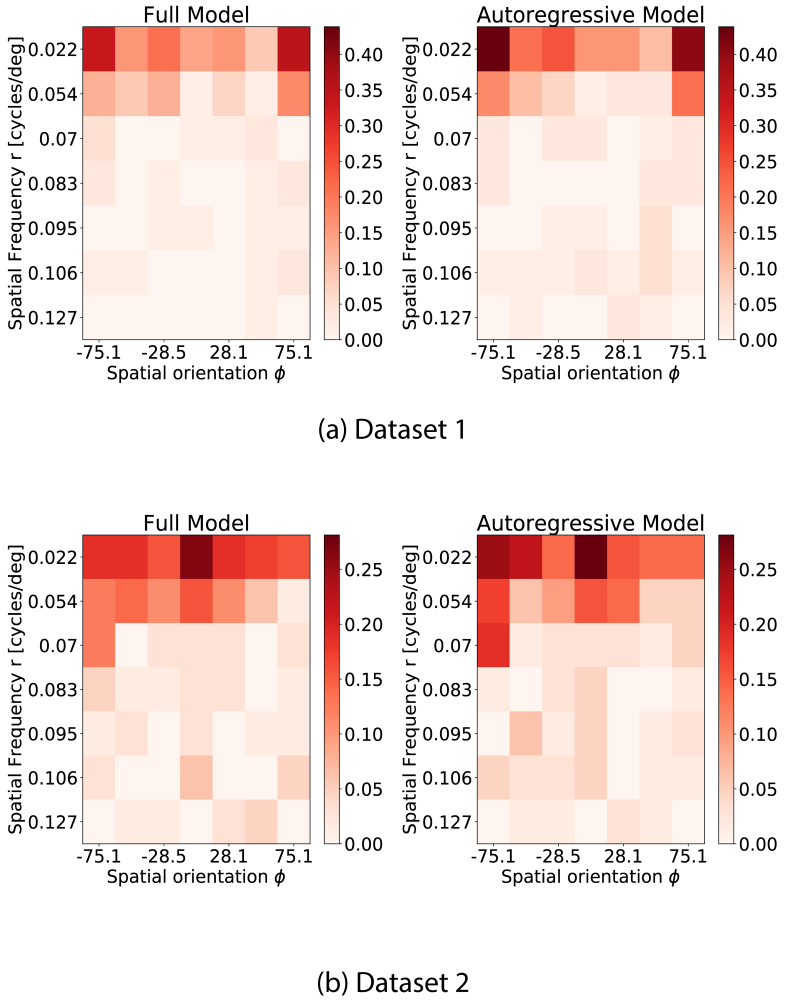


Fig 14. Percentage of neurons that have an excitatory or inhibitory response to each possible visual stimuli. Visual stimuli are represented in matrix form, where rows represent spatial frequency r and columns spatial orientation ϕ. The value for each entry in the matrix is the percentage of neurons in the datasets that show a response to that visual stimuli. This visualization shows that both datasets show a large number of directly responding neurons for low spatial frequency visual stimuli.

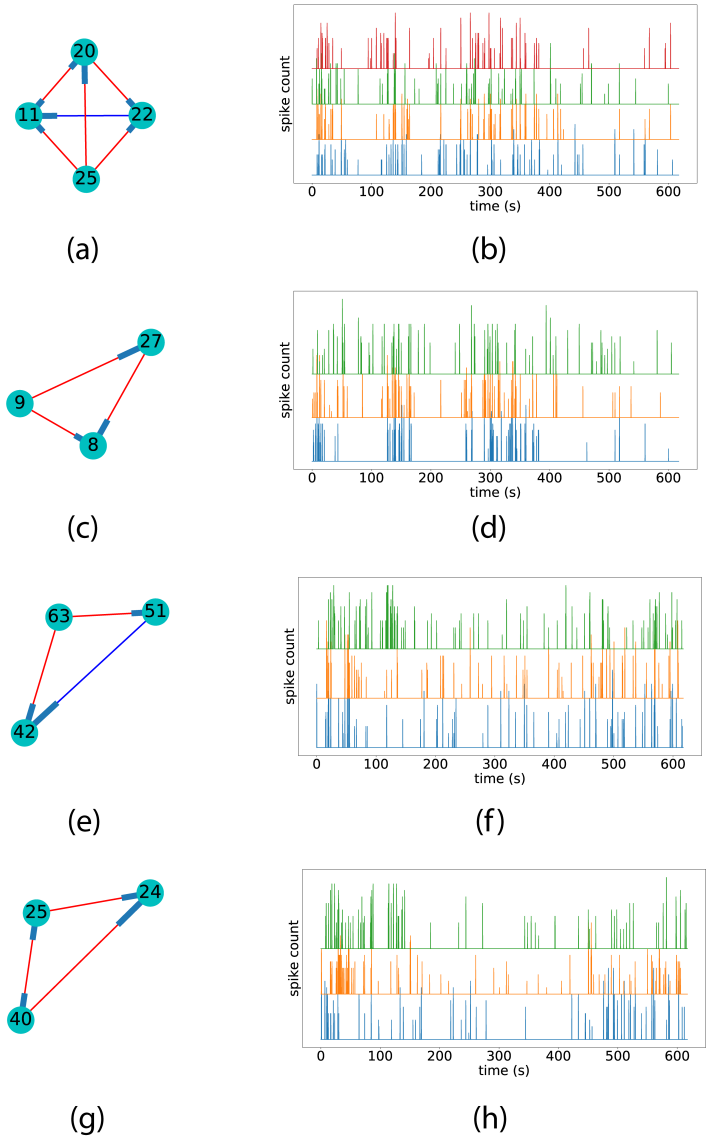


Fig 15. Figures a), c), e), and g) show the excitatory (red) and inhibitory (blue) edges detected for neuron cliques in datasets 2 (a) and c)) and 3 (e) and g)). Figures b), d), f), and h) show the spike time series of the neurons in the clique. The nodes are numbered according to the corresponding neuron index. We can visually see that spike trains from neurons in the selected cliques show similar spiking behaviour throughout the experiment.

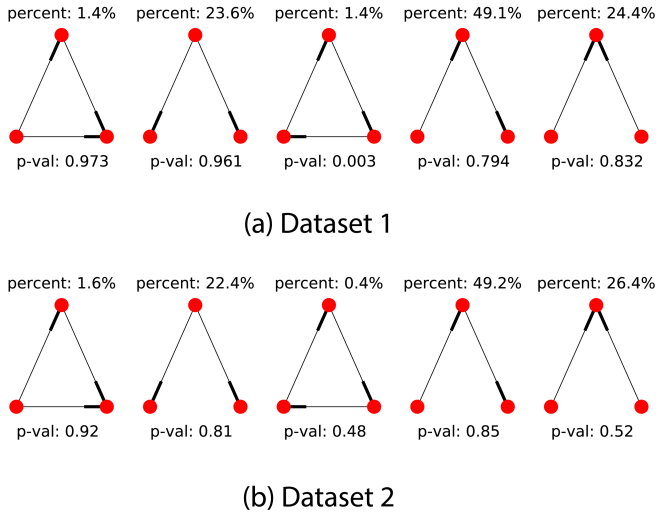


Fig 16. We count the occurrences of motif triplets for both datasets (we ignore edge weight and sign) by enumerating all neuron triplet combinations in the recovered networks and checking for graph isomorphism against all 5 motif triplet types. Top and bottom rows show results for datasets 1 and 2 respectively. We compare the obtained motif counts against a base model of small-world network topology and show the obtained p-values. These p-values are obtained by computation of the mean and standard deviation of each motif type in a small-world network with the same node count and edge density. The relatively large p-values obtained show that the small-world model is a good fit for the recovered network topology.

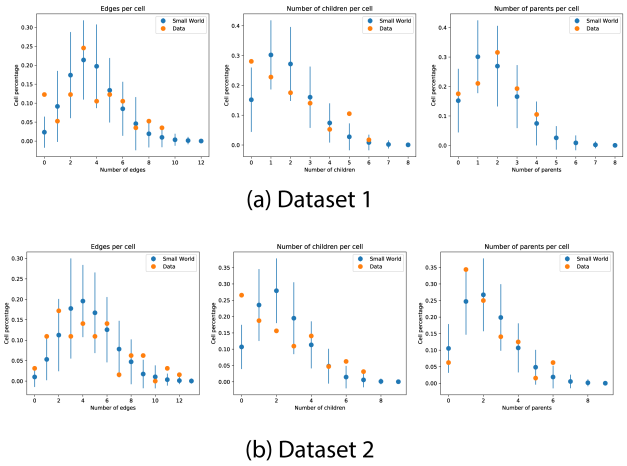
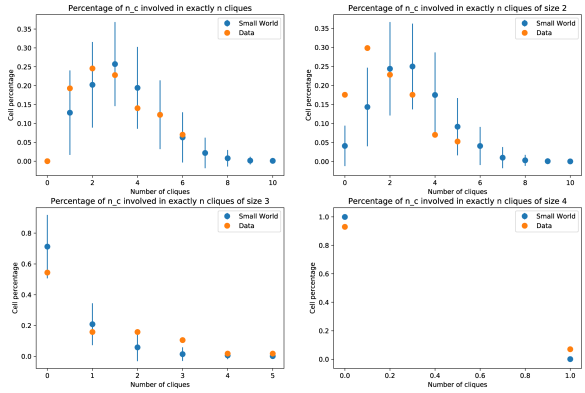
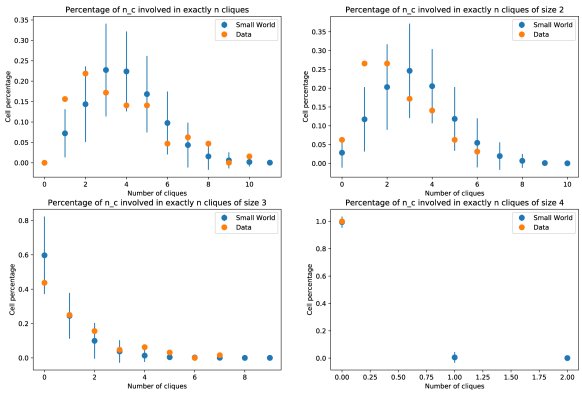


Fig 17. We compare the edge density distribution of the recovered inter neuron connectivity matrices in both datasets. Edge counts shown from left to right are all edges (number of neurons connected to node, either as parent or child), outbound edges (number of child nodes), and inbound nodes (number of parent nodes). The edge counts are compared against a base model of small-world network topology, error bars denote two standard deviations obtained from simulation of small-world networks with the same number of nodes and connectivity degree.



(a) Dataset 1



(b) Dataset 2

Fig 18. We count the percentage of cells participating in multiple cliques. The counts are compared against a base model of small-world network topology, error bars denote two standard deviations obtained from simulation of small-world networks with the same number of nodes and connectivity degree.

Discussion

There is a large body of work done in reverse-engineering neural circuits from data across different modalities [40, 41]. Some of these methods learn a directed graph, which provides a compact and interpretable way of encoding Granger causal [42] relations between neurons and covariates. Several works have been published on the use of Gaussian Bayesian Networks to learn connectivity strengths from data [43, 44]. For cases where the framework is not applicable, as in the case of spike train time series, Generalized Linear Models (GLMs) have been successfully used [24, 45–51].

Parameters for these models are most commonly estimated using Maximum Likelihood Estimation or Maximum a Posteriori estimation [24, 46–51].The parameters obtained from the GLM can be interpreted as a directed graph capturing dependencies

between variables of interest, or nodes (neuron spiking rates and visual stimuli). The presence of an edge in this graph represents a directed influence from one node to another, and the weight of the edge represents the magnitude of that influence. The graph can be made sparse with the use of subset selection, where only a limited number of edges are assigned non-zero weights (no influence). Subset selection can be performed using deviance tests [52] or by the use of priors [50].

In parallel, there is a corresponding push for actively estimating the best stimuli subset for network inference, with variants based on mutual information and Gaussian approximations of MAP parameters [24, 46, 48, 49, 53–62]

Methods like the one proposed in [53] use of mutual information for intervention selection, but rely on a specific Gaussian Bayesian Network framework, making them unsuitable for count data. Methods like [63] generalize D-Optimal factorial design for GLMs and to multi-level regressor covariates, but require full control over the regressor covariates.

The proposed variable selection algorithm is focused on subset selection of parent regressors. It is posed as an optimization problem where the objective is to find a set of regressors that minimize the BIC score, subject to a confidence interval restriction. Using BIC as the score to optimize fosters prediction improvement while penalizing model complexity. The use of p-value as a restriction criteria ensures that the regressors in the model are highly significant. A local minimal set of regressors is constructively obtained following a simple rule-set.

For the purposes of this work, the delay time window (boxcar length) was set directly to incorporate a delay of 5 time bins ($\sim 0.3s$). In a more general setting, the delay parameter could be set by fitting a model to data with various values of this parameter, and choosing the one with the highest likelihood.

The variable selection method was tested on simulated data and compared against oracle Lasso. It performed worse than oracle lasso for very small sample sizes, but otherwise proved to be better on the F_1 and recall metrics. On settings where no ground truth is available, Lasso would require some other sub-optimal method for parameter tuning. The p-value restriction parameter present in the elastic-forward model selection algorithm is easily interpretable as the desireable confidence level on the regressor parameters, this makes the parameter easy to set beforehand for any experiment

Note that the time complexity of the forward model selection scales as the product of the number of available regressors and true edges in the network. For the worst case scenario (fully connected network), this means the forward model selection strategy scales quadratically with the number of regressors. For sparse networks, this computational cost can be brought down significantly by preselecting regressors based on the approximate p-value computation shown in Eq. (47).

The proposed active learning algorithm follows a simple design philosophy, it looks for promising edges not added into the model so far, and increases the appearance frequency of stimuli that drive up the spiking rate of the parent nodes of these edges. To achieve this, it defines a score for each possible stimulus based on the previous learned model, it takes into account the spiking rate difference (impact) of presenting one stimulus more frequently than the others on every parent node in the system, and weighs it by the potential log-likelihood improvement of adding every edge associated with this parent node to the model. The log-likelihood value of an edge is tightly related with the BIC score the elastic-forward model selection attempts to optimize.

Active learning proved to be faster than random stimulation in recovering edges on the simulated networks, this was especially true for edges whose spiking rate could be greatly affected by changing the stimuli distribution (first order connections). On the simulations from networks recovered from real data, the performance of active learning was consistently better than random stimulation, and the measured metrics had a

tighter spread. There is therefore no reason not to use active learning during data acquisition.

We measured three basic properties of the recovered inter-neuron connectivity networks from on real data: edge distribution, motif type distribution, and number of cliques per neuron. Counts were compared to simulations of small-world networks with an identical number of nodes and connectivity degree. All three properties measured fell well within the expected values for these types of networks, pointing at a small-world-like structure in the recovered inter-neuron connectivity networks.

All codes used for this work are available at `https://github.com/MartinBertran/ActiveLearningCortical`

Conclusion

In this paper we propose a simple framework for actively learning network connectivity for GLMs by selecting external forcing actions. The algorithm has the advantage of making relatively few assumptions on the exact distribution of the model, and the amount of control the experimenter has over the regressor covariates.

The use of a greedy regressor selector using BIC and Wald testing allows for an easy identification of edges that seem beneficial for the model, but do not yet have a sufficient number of interaction samples to be included in the model. By utilizing external triggers to these interactions, the algorithm prioritizes interventions that provide information over uncertain edges.

The greedy regressor selector outperforms the oracle Lasso in identifying the proper regressor subsets in simulations for non-small sample sizes, even when accounting for the oracle selection of the l_1 prior.

Even on the recovered real datasets, the use of the active learning algorithm proved to be beneficial as well. The algorithm is very quick at recovering directly connected edges, making its application in conjunction with optogenetics an interesting proposition.

Finally, we note the method is not restricted to one modality or domain, it can be applied in any situation where there is a family of possible actions available to probe the activity of a network, in both biological and artificial systems.

We note that for all measured properties, the recovered network structure on the real datasets was consistent with a small-world topology.

Appendix

Derivation of the observed Fisher information

Given a neuron c and the MLE estimators $\hat{\theta} = \{\hat{W}_c, \hat{H}_c, \hat{b}_c\}$, the element k, j of the observed Fisher information matrix $\hat{\mathcal{I}}_c(\hat{\theta})$ for the given model can be expressed as

$$[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,j} = -\sum_{t=1}^{m}[-\frac{y_c(t)}{\lambda_c(t)^2}[\frac{\partial \lambda_c(t)}{\partial \eta_c}]^2 R_i(t)R_j(t) + [\frac{y_c(t)}{\lambda_c(t)} - 1]\frac{\partial^2 \lambda_c(t)}{\partial \eta_c^2} R_i(t)R_j(t)], \quad (29)$$

where R is the concatenation of all the considered regressors (neurons and stimuli),

$$R(t) = [\hat{X}(t), \hat{I}(t), 1] \in \mathbf{R}^{m \times [n_s + n_c + 1]}, \quad (30)$$

and the first and second derivatives of $\lambda_c(t)$ are,

$$\frac{\partial \lambda_c(t)}{\partial \eta_c} = \frac{e^{\kappa\eta_c(t)}}{1+e^{\kappa\eta_c(t)}}, \tag{31}$$

$$\frac{\partial^2 \lambda_c(t)}{\partial \eta_c^2} = \frac{\kappa e^{\kappa\eta_c(t)}}{(1+e^{\kappa\eta_c(t)})^2}. \tag{32}$$

Replacing Eq. (31) and Eq. (32) in Eq. (29) we get:

$$\begin{aligned}[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,j} &= -\sum_{t=1}^{m}[-\frac{y_c(t)}{\lambda_c(t)^2}[\frac{e^{\kappa\eta_c(t)}}{1+e^{\kappa\eta_c(t)}}]^2 R_i(t)R_j(t) + [\frac{y_c(t)}{\lambda_c(t)} - 1]\frac{\kappa e^{\kappa\eta_c(t)}}{(1+e^{\kappa\eta_c(t)})^2} R_i(t)R_j(t)], \\ &= \sum_{t=1}^{m}\big[\frac{y_c(t)}{\lambda_c(t)^2}e^{\kappa\eta_c(t)} + \kappa(1-\frac{y_c(t)}{\lambda_c(t)})\big]\frac{e^{\kappa\eta_c(t)}}{(1+e^{\kappa\eta_c(t)})^2}R_i(t)R_j(t).\end{aligned} \tag{33}$$

Approximate Wald test dependence on model parameter κ

We now relate the Fisher information matrix to the κ parameter and other quantities that do not depend on the Maximum Likelihood estimate of the model. To achieve this, we make several approximations. We start by expressing Eq. (33)

$$[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,j} = \sum_{t/R_i(t)R_j(t)\neq 0} g(y_c(t), R_i(t), R_j(t), \eta_c(t)), \tag{34}$$

with

$$g(y_c(t), R_i(t), R_j(t), \eta_c(t)) = \big[\frac{y_c(t)}{\lambda_c(t)^2}e^{\kappa\eta_c(t)} + \kappa(1-\frac{y_c(t)}{\lambda_c(t)})\big]\frac{e^{\kappa\eta_c(t)}}{(1+e^{\kappa\eta_c(t)})^2}R_i(t)R_j(t). \tag{35}$$

The first simplification we make is approximating the summation in Eq. (34) by its expected value. If we define $M_{i,j}$ as the number of samples where $R_iR_j \neq 0$, we get

$$[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,j} = \sum_{t/R_i(t)R_j(t)\neq 0} g(y_c(t), R_i(t), R_j(t), \eta_c(t)) \approx M_{i,j}E[g|R_iR_j \neq 0], \tag{36}$$

We further approximate this expression by moving the expectation inside the function,

$$[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,j} \approx M_{i,j}E[g|R_iR_j \neq 0] \approx M_{i,j}g(y_c^{ij}, R_i^{ij}, R_j^{ij}, \eta_c^{ij}), \tag{37}$$

where

- $y_c^{ij} = E[y_c(t)|R_i(t)R_j(t) \neq 0]$,
- $R_i^{ij} = E[R_i(t)|R_i(t)R_j(t) \neq 0]$,
- $R_j^{ij} = E[R_j(t)|R_i(t)R_j(t) \neq 0]$,

- $\eta_c^{ij} = E[\eta_c(t)|R_i(t)R_j(t) \neq 0]$.

Going back to the full expression we get

$$M_{i,j}g(y_c^{ij}, R_i^{ij}, R_j^{ij}, \eta_c^{ij}) = M_{i,j}\big[\frac{y_c^{ij}}{(\lambda_c^{ij})^2}e^{\kappa\eta_c^{ij}} + \kappa(1 - \frac{y_c^{ij}}{\lambda_c^{ij}})\big]\frac{e^{\kappa\eta_c^{ij}}}{(1+e^{\kappa\eta_c^{ij}})^2}R_i^{ij}R_j^{ij}. \quad (38)$$

Notice that $y_c^{ij} = \lambda_c^{ij}$ for a well fitted model, where $\lambda_c^{ij} = \frac{log(1+e^{\kappa\eta_c^{ij}})}{\kappa}$
Then we have that:

$$[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,j} \approx M_{i,j}[\frac{e^{\kappa\eta_c^{ij}}}{\lambda_c^{ij}}]\frac{e^{\kappa\eta_c^{ij}}}{(1+e^{\kappa\eta_c^{ij}})^2}R_i^{ij}R_j^{ij}. \quad (39)$$

From the definition of $\lambda_c(\eta_c)$ (Eq. (4)) we can express $e^{\kappa\eta_c^{ij}} = e^{\kappa\lambda_c^{ij}} - 1$ and obtain

$$[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,j} \approx M_{i,j}\frac{(e^{\kappa\lambda_c^{ij}} - 1)^2}{\lambda_c^{ij} \times (e^{\kappa\lambda_c^{ij}})^2}R_i^{ij}R_j^{ij}. \quad (40)$$

We relate this Fisher information matrix approximation to the z-score of any given regressor. To avoid computing the inverse of the Fisher information matrix, we take the following approximation:

$$[\hat{\mathcal{I}_c^{-1}}(\hat{\theta})]_{i,i} \approx \frac{1}{[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,i}}, \quad (41)$$

where

$$[\hat{\mathcal{I}}_c(\hat{\theta})]_{i,i} \approx M_i\frac{(e^{\kappa\lambda_c^i} - 1)^2}{\lambda_c^i e^{2\kappa\lambda_c^i}}(R^i)_i^2. \quad (42)$$

By replacing this approximation into Eq. (9), the Z-score of regressor i on neuron c can be very roughly approximated to :

$$\begin{aligned} Z_{ic}^2 &\approx w_{ic}^2 \times [\hat{\mathcal{I}}_c(\hat{\theta})]_{i,i} && (43)\\ &= M_i\frac{(e^{\kappa\lambda_c^i}-1)^2}{\lambda_c^i e^{2\kappa\lambda_c^i}}(w_{ic}R_i^i)^2. && (44)\end{aligned}$$

Our final task is relating the term $w_{ic}R_i^i$ to quantities that do not depend on the Maximum Likelihood estimate. To do this, we again note that for any well fitted model we can write:

$$\begin{aligned} E[y_c|R_i \neq 0] &= \lambda_c^i \\ &= \frac{log(1+e^{\kappa\eta_c^i})}{\kappa}, \\ E[y_c|R_i = 0] &= \lambda_c^{-i} \\ &= \frac{log(1+e^{\kappa\eta_c^{-i}})}{\kappa}. \end{aligned} \quad (45)$$

We conveniently express $\eta_c^i = w_{ic}E[R_i|R_i \neq 0] + \eta_c^{\setminus\{i\}}$ where $\eta_c^{\setminus\{i\}}$ is the contribution of all regressors beside regressor i. Similarly, we express $\eta_c^{-i} = \eta_c^{\setminus\{i\}}$, where we further assumed that the total contribution of all other regressors to the η_c parameter is mostly independent on the state of regressor R_i.

Working both equations, we obtain:

$$w_{ic}R_i^i = \frac{1}{\kappa}log(\frac{e^{\kappa\eta_c^i}}{e^{\kappa\eta_c^{-i}}}) = \frac{1}{\kappa}log(\frac{e^{\kappa\lambda_c^i}-1}{e^{\kappa\lambda_c^{-i}}-1}). \tag{46}$$

We finally conclude that, after several approximations, the Z-score of a parameter can be roughly approximated with an expression that only depends on the rates of our data and the parameter κ of our model,

$$\begin{aligned} Z_{ic}^2 &\approx w_{ic}^2 \times [\hat{\mathcal{I}}_c(\hat{\theta})]_{i,i} \\ &\approx M_i \frac{(e^{\kappa\lambda_c^i}-1)^2}{\lambda_c^i e^{2\kappa\lambda_c^i}}(\frac{1}{\kappa}log(\frac{e^{\kappa\lambda_c^i}-1}{e^{\kappa\lambda_c^{-i}}-1}))^2 \\ &\approx \frac{M_i}{\kappa^2}\frac{(e^{\kappa\lambda_c^i}-1)^2}{\lambda_c^i e^{2\kappa\lambda_c^i}}(log(\frac{e^{\kappa\lambda_c^i}-1}{e^{\kappa\lambda_c^{-i}}-1}))^2 \end{aligned} \tag{47}$$

Using this approximation, we can gain some intuition on how the Z-score will behave under certain conditions. In particular, we see that the Z-score depends approximately linearly on the number of observed interactions, it also depends on what is essentially a ratio between $E[y_c|R_i \neq 0]$ and $E[y_c|R_i = 0]$. The dependence on the κ parameter is not easy to discern at a glance, but we can observe how the Z-score behaves for various combinations of spiking rate parameters and κ values.

Fig. 19 shows how this approximate Z-score function behaves for several κ as a function of the fractional rate $\frac{\lambda_c^i}{\lambda_c^{-i}}$ and λ_c^{-i}. Rates and base spiking rates where selected to be in accordance to the rates measured in our real datasets.

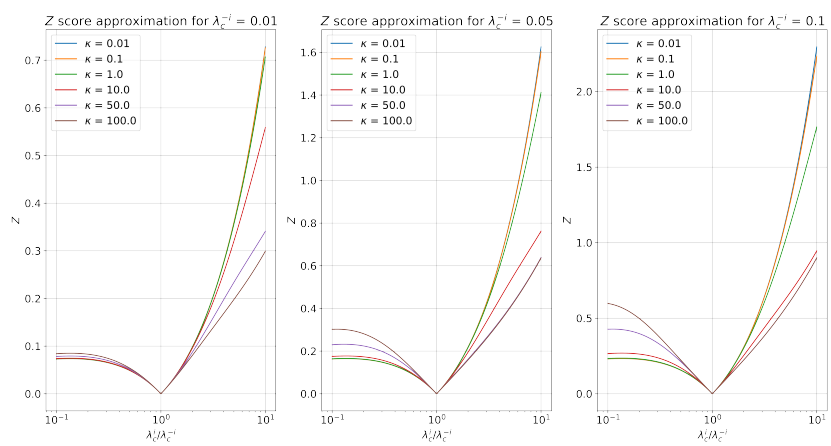


Fig 19. Single sample ($M = 1$) Z-score approximation according to Eq. (46) for several κ parameters as a function of the ratio $\frac{\lambda_c^i}{\lambda_c^{-i}}$ and λ_c^{-i} ($E(y_c|R_i = 0)$). The λ_c^{-i} values were chosen to be representative of the observed rates in our real datasets.

Numerical issues arise from picking very large values of κ. We decided to choose $\kappa = 10$ for all experimental results since it seemed to provide a nice middle ground

between dampening the Z-score for low spiking rate inhibitions and not giving disproportionately large Z-scores for small excitatory rates.

Fisher information: Model selection and the under-sampled regime

We analyze the behaviour of ill-conditioned but non-singular Fisher matrices, where there is a potentially large difference between the largest and smallest eigenvalues present in the Fisher information matrix, and show how the Wald test present in the model selection strategy copes with this issue. We use this analysis as we take the limit of the smallest eigenvalue going to 0 to illustrate what this test does for singular matrices.

Overall, we provide a link between the smallest eigenvalue of the Fisher information matrix, and the largest observed parameter variance we obtain from the diagonal elements of the inverse Fisher information matrix. From this we finally conclude that regressor subsets that produce nearly singular matrices are consistently rejected from our model selection process.

We first note that, for any non-singular matrix A, if λ is an eigenvalue of A, then λ^{-1} is an eigenvalue of A^{-1}. So the inverse of a matrix with a very small eigenvalue has a correspondingly large eigenvalue.

We also note that the Fisher information matrix is positive (semi) definite, we can thus sort its eigenvalues in descending order and write $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_r \geq 0$, where r is the number of rows (regressors) of the matrix. The corresponding eigenvalues of the inverse Fisher matrix are ordered: $\lambda_r^{-1} \geq ... \geq \lambda_1^{-1} \geq 0$

Since the inverse Fisher information is symmetric, we can use the Schur-Horn theorem to see that its diagonal elements are majorized by its eigenvalues. By definition of majorization, this means that:

$$x_1 + ... + x_i \leq \lambda_r^{-1} + ... + \lambda_{r-i+1}^{-1}; \quad \forall i = 1, ..., r \tag{48}$$

where x_i is the i-th largest entry in the diagonal elements in the inverse Fisher matrix, the equality holds exactly for $i = r$. These values are the fisher variances of the regressors in Eq. (9), sorted by descending variance.

From this, we can easily see that the largest observed fisher variance is at least

$$\begin{aligned} x_1 &\geq \frac{1}{r} \sum_{i=1}^{r} \lambda_i^{-1} \\ &\geq \frac{1}{r} \sum_{i=1}^{r-1} \lambda_i^{-1} + \frac{1}{r\lambda_r} \end{aligned} \tag{49}$$

From this we can conclude that as the smallest value of the Fisher information matrix goes to 0, at least one regressor's Fisher variance grows arbitrarily large. Relating this back into Eq. (9) and Eq. (12) we can see that for any reasonable p-value threshold, a parameter subset PA_c yielding an ill-conditioned Fisher information matrix will be rejected.

Regressor selection: fraction parameter ν

We evaluate the effect of varying the sample fraction ν used for the random subset selection step of the elastic-forward model selection method. The ν parameter is tested in the 0.5 to 0.9 ranges. This is compared against the baseline oracle lasso method and model selection where no random subset selection is performed (no-subset).

All methods are evaluated on samples drawn from simulated network SW1CL using the F_1, precision, and recall performance metrics. Results are shown in Fig. 20.

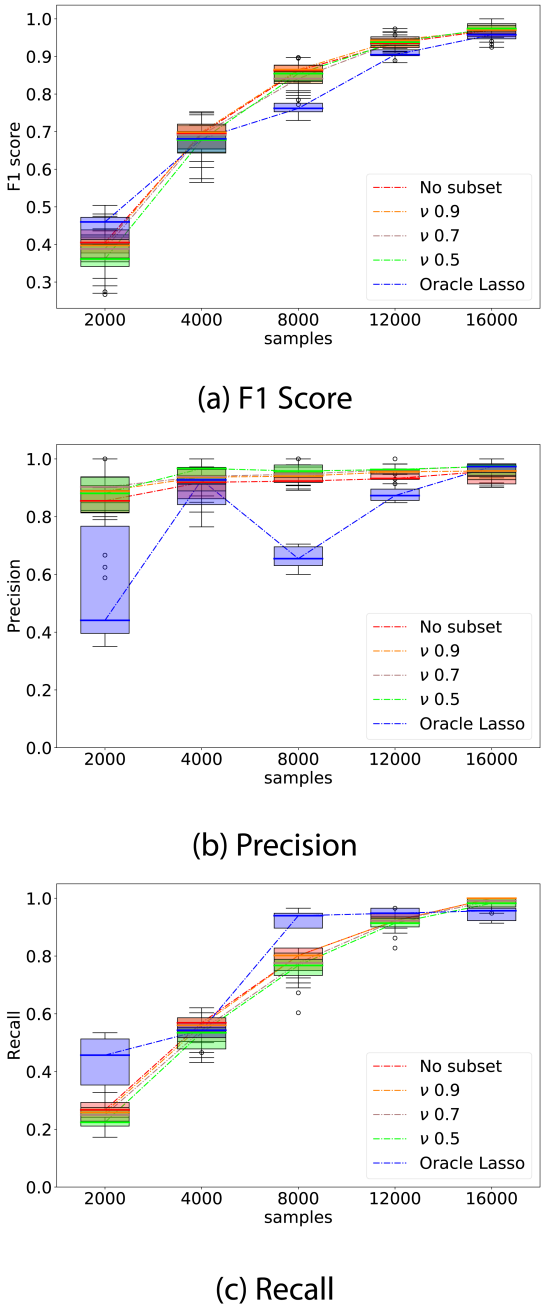


Fig 20. Whisker plot of performance indicators as a function of number of samples; elastic-forward BIC selection is compared using several ν parameters against both oracle Lasso and the special case where the whole dataset is used at once, without splitting into random subsets (no-subset). The plots show relatively little difference between the various ν parameters, but the use of the ν parameter is better performing overall to both oracle lasso and no-subset model selection

We can observe from Fig. 20 that all ν parameters perform similarly for all but the smallest sample sizes. Additionally, the use of no-subset model selection had a large performance decrease for small sample sizes when compared to all ν parameters, but similar performance on larger sample sizes.

Poisson model selection on non-Poisson models

A natural question that may arise is if the Poisson GLM model can still capture directed interactions between neurons when the data does not come from a Poisson distribution.

To that effect, we repeated the simulated experiments using the same connectivity matrices defined for networks SW1CL and SW3CL, but this time, the spiking activity was obtained using a Leaky Integrate and Fire model [64].

The equations of the model can be summarized as

$$\begin{aligned} \dot{v_c}(t) &= -v_c(t) + b(t) + \sum_{u=1}^{n_c} W_{u,c} \times s_u(t) \circledast h(t) + \sum_{v=1}^{n_s} H_{v,c} \times I_v(t) \circledast h(t), \\ h(t) &= a^2 \times e^{-a(t-t_D)} \times (t - t_D) \times \mathbf{1}_{t-t_D \geq 0}, \\ \text{If} v_c(t) \geq 1 &= \begin{cases} v_c(t) = 0, \\ s_c(t) = 1, \end{cases} \end{aligned} \tag{50}$$

where v_c is the membrane potential of neuron c, and s_c its corresponding spike train.

W and H are the inter-neuron and stimuli-neuron connectivity matrices respectively. The parameter $h(t)$ represent the influence kernel, and depends on the synaptic density (a) and kernel delay (t_D). $b(t)$ is the direct current parameter.

For our simulations we sampled $b(t)$ from a uniform distribution ($b(t) \sim U[0, b]$), where b was chosen such that the average spiking rate of each neuron is the same as in the original Poisson GLM simulations. The W and H connectivity matrices were linearly scaled with respect to the original SW1CL network to preserve the conditional spiking rates. For these simulations, we set $a = 1.5$ and $t_D = 2$.

Fig. 21 compares the performance of the Active Learning algorithm and Uniform sampling when applied to the non-Poisson, Leaky Integrate and Fire network.

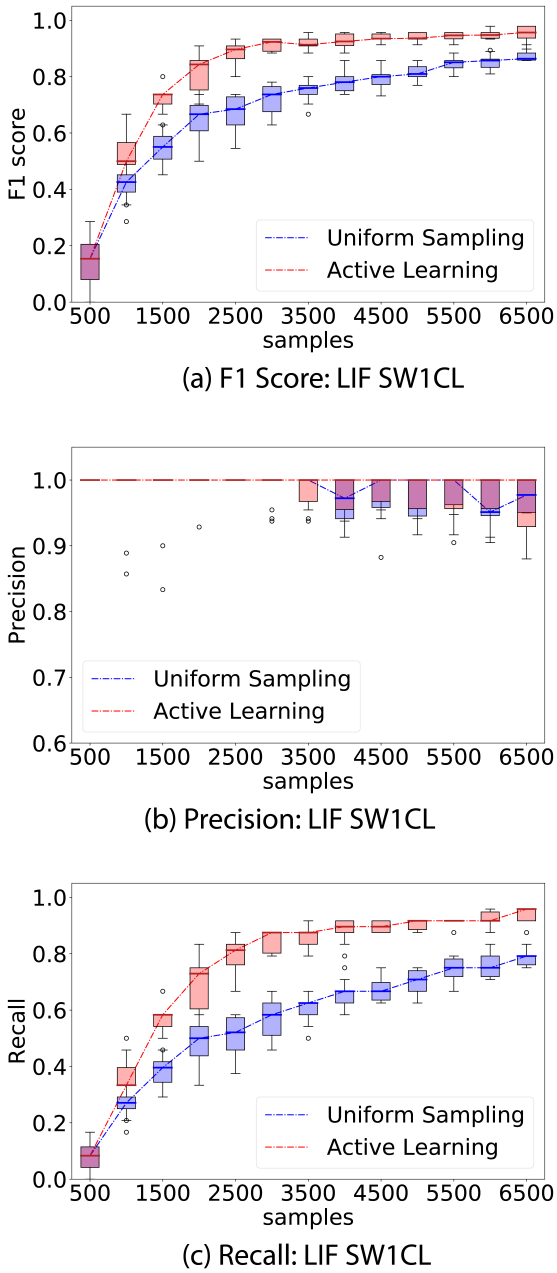


Fig 21. Comparison of performance between the proposed active learning method versus uniformly sampling from all stimuli. Spiking trains were simulated using a Leaky Integrate and Fire model, but the recovered networks were done using the proposed Poisson GLM. The experiment consisted of 500 sample interventions, with an initial 500 sample observation. Whisker plots are obtained from 10 independent trials.

From Fig. 21, we see that, at least for this tested configuration, the Poisson GLM is still able to recover the correct directed connections even when the originating data model does not follow a Poisson distribution.

Table of defined variables

764

Variable symbol	Variable Definition
m	number of acquired samples
n_c	number of neurons
n_r	number of regressors
$X = \{X_i\}_{i=1,...,n_c}\ X_i \in \mathbf{R}^m$	spike time series of target neurons
$R = \{R_i\}_{i=1,...,n_r}\ R_i \in \mathbf{R}^m$	regressor time series
b_c	bias of target neuron c
w_{ic}	influence weight of regressor i on neuron c
PA_c	set of parent regressors of neuron c
n_s	number of external stimuli
C	set of n_c neurons
S	set of n_s external stimuli
$I = \{I_i\}_{i=1,...,n_s}\ I_i \in \mathbf{R}^m$	external stimuli time series
$\hat{X} = \{\hat{X}_i\}_{i=1,...,n_c}$	past neuron activity
$\hat{I} = \{\hat{I}_i\}_{i=1,...,n_s}$	past stimuli activity
$D_s^u,\ D_s^l$	upper and lower bounds of boxcar influence function for stimuli
$D_c^u,\ D_c^l$	upper and lower bounds of boxcar influence function for neurons
C_c'	set of parent neurons of neuron c
S_c'	set of parent stimuli of neuron c
$W_c \in \mathbf{R}^{\lvert C_c' \rvert}$	edge weights between parent neurons of c and neuron c
$H_c \in \mathbf{R}^{\lvert S_c' \rvert}$	edge weights between parent stimuli of c and neuron c
$W = \{w_{ij} : i, j = 1, ..., n_c\}$	adjacency matrix
$H = \{h_{ij} : i = 1, ..., n_s,\ j = 1, ..., n_c\}$	stimuli response matrix
λ_c	instantaneous Poisson spiking rate of neuron c
κ	spiking rate non-linearity calibration constant
$\{\hat{X}\}_{C_c'} = \{\hat{X}_j\}_{j \in C_c'}$	past neuron activity of parent neurons of c
$\{\hat{I}\}_{S_c'} = \{\hat{I}_j\}_{j \in S_c'}$	past stimuli activity of parent stimuli of c
$\hat{\theta} = \{\hat{W}_c, \hat{H}_c, \hat{b}_c\}$	MLE estimate of model parameters of neuron c
$\hat{W},\ \hat{H}$	estimators of adjacency matrix W and stimuli response matrix H
$\widehat{PA}_c$	estimated parent set of neuron c
γ	p-value restriction
l	active learning iteration counter
m_l	number of samples acquired up to active learning iteration l
$\hat{W}^l,\ \hat{H}^l$	estimators of adjacency matrix W and stimuli response matrix H up to iteration l
P^{l+1}	probability distribution vector for presenting each stimuli S at intervention $l+1$
$\hat{P}_s$	surrogate stimuli probability distribution vector, where stimuli s is presented more frequently
$\hat{P}$	uniform stimuli probability distribution vector
$\{S\}_{P=\hat{P}_s}$	sequence of stimuli drawn from distribution $\hat{P}_s$
$\widehat{ERC}_{s,c}$	expected rate change of neuron c when using stimuli probability distribution $\hat{P}_s$
$\hat{D}_{c,c_i}$	Deviance statistic in neuron c between model with and without regressor neuron c_i
$\widehat{ERC}_{s,s_i}$	expected rate change of stimuli s when using stimuli probability distribution $\hat{P}_s$
$\hat{D}_{c,s_i}$	Deviance statistic in neuron c between model with and without regressor stimuli s_i
$\widehat{SC}_{s,W}[l+1]$	score of stimuli s associated with inter neuron matrix W
$\widehat{SC}_{s,H}[l+1]$	score of stimuli s associated with stimuli response matrix H
$\widehat{SC}_s[l+1]$	full score of stimuli s
β	active learning smoothing constant

Table 1. Summary of defined variables

Algorithms

765

Algorithm 1 Elastic-forward selection

Require: $X_c,\ \hat{R}_r,\ k,\ \gamma$

▷ X_c sequence of observations of neuron c
▷ $\hat{R}_r$ is the full set of regressor observations
▷ k maximum number of regressors to add per step
▷ γ minimum allowed regressor p-value

$r' = \{\}$ ▷ Initialize set of active regressors to empty set (only bias is included).
repeat
  $\hat{\theta}_{r'};\ \hat{L_{r'}};\ \hat{I}(\hat{\theta}_{r'});\ pvals_{r'};\ \mathrm{BIC}_{r'} = \mathrm{EvaluateRegressors}(X_c;\ \hat{R}_r;\ r')$

  $r^{\dagger}, \{j_i\} = \mathrm{Forward\,Model\,Proposal}(X_c,\ \hat{R}_r,\ r',\ \gamma,\ k,\ \mathrm{BIC}_{r'})$
  ▷ Both sub routines are explained in algorithms 3 and 4
  if $\{j_i\} = \emptyset$ **then**
  ▷ No suitable candidate found
    return $(\hat{\theta}_{r'};\ \hat{L_{r'}};\ \hat{I}(\hat{\theta}_{r'});\ \mathrm{pvals}_{r'};\ \mathrm{BIC}_{r'})$
  end if
  success = False
  $n = |\{j_i\}|$
  $r_{\mathrm{best}} = r'$
  $\mathrm{BIC}_{\mathrm{best}} = \mathrm{BIC}_{r'}$
  while not sucess **do**

    $r^{\ddagger} = r' + \{j_i\}[:n]$
    $\hat{\theta}_{r^{\ddagger}};\ \hat{L_{r^{\ddagger}}};\ \hat{I}(\hat{\theta}_{r^{\ddagger}});\ pvals_{r^{\ddagger}};\ \mathrm{BIC}_{r^{\ddagger}} = \mathrm{EvaluateRegressors}(X_c;\ \hat{R}_r;\ r^{\ddagger})$
    ▷ Update best set found so far using Algorithm 3
    if $max(\mathrm{pvals}_{r^{\ddagger}}) \leq \gamma$ and $\mathrm{BIC}_{r^{\ddagger}} \leq \mathrm{BIC}_{\mathrm{best}}$ **then**
      $r_{\mathrm{best}} = r^{\ddagger}$
      $\mathrm{BIC}_{\mathrm{best}} = \mathrm{BIC}_{r^{\ddagger}}$
    end if
    ▷ Already found best set in the descending sequence, update and exit loop
    if $\mathrm{BIC}_{r^{\ddagger}} \geq \mathrm{BIC}_{\mathrm{best}}$ and $r_{\mathrm{best}} \neq r'$ **then**
      success = True
      $r' = r_{\mathrm{best}}$
    end if
    $n = n - 1$
  end while
until True

Algorithm 2 Iterative Active Learning

Require: $X,\ I,\ k,\ \gamma,\ \beta$

▷ X Initial observations for all neurons
▷ I Initial applied stimuli
▷ k maximum number of regressors to add per step
▷ γ minimum allowed regressor p-value
▷ β active learning smoothing parameter

for $l \in$ interventions **do**
  $\hat{R}_r \leftarrow$ BuildRegressors(X, I) ▷ Build regressor sequences from X and I as defined for Eq. (3)
  for $c \in$ neurons **do**
    ▷ Perform model selection using Algorithm 4
    $(\hat{\theta}_{r'}[c], \hat{L_{r'}}[c], \hat{I}(\hat{\theta}_{r'})[c], pvals_{r'}[c], \mathrm{BIC}_{r'}[c]) = \mathrm{ElasticForwardSelection}(\mathrm{X_c}, \hat{\mathrm{R}}_\mathrm{r}, \mathrm{k}, \gamma)$

    ▷ Get log-likelihood difference of considering all regressors that are not parent edges of neuron c
    $R_{nonparent(c)} = \{r{:}\hat{\theta}_{r'}[c] = 0\}$

    for $r \in R_{nonparent(c)}$ **do**
      $L_{r,c} \leftarrow$ Compute Loglikelihood difference ▷ Use Eq. (15) for neuron regressor and Eq. (18) for stimuli regressor
    end for
  end for

▷ Compute the impact and score of each stimuli according to previous observations
  for $s \in stimuli$ **do**
    for $c \in neurons$ **do**
      $IM_{s,c} \leftarrow$ Compute impact of stimuli s on neuron c ▷ Use Eq. (14)
    end for

    for $s_i \in stimuli$ **do**
      $IM_{s,s_i} \leftarrow$ Compute impact of stimuli s on stimuli s_i ▷ Use Eq. (17)
    end for
    $SC_s \leftarrow$ Compute score of stimuli s ▷ Use Eqs. (16,19,20)
  end for

  $P^{l+1} \leftarrow$ Compute Stimuli Probability Vector ▷ Use Eqs. (21,23)
  $(X, I) \leftarrow$ AcquireSamples ▷ Acquire samples of X and I drawing I with probability P^{l+1}

end for

Algorithm 3 Sub routine: Evaluate Regressors

Require: $X_c,\ \hat{R}_r,\ r^\dagger$

▷ X_c sequence of observations of neuron c

▷ $\hat{R}_r$ is the full set of regressor observations ($\hat{R}_r = [\hat{X}, \hat{I}, 1]$)

▷ $r^\dagger$ subset of regressors to consider

$\hat{\theta}_{r^\dagger} = \text{argmax}_{\theta_{r^\dagger}} \mathcal{L}(X_c; \hat{R}_{r^\dagger}; \theta_{r^\dagger})$

$\hat{L}_{r^\dagger} = \mathcal{L}(X_c; \hat{R}_{r^\dagger}; \hat{\theta}_{r^\dagger})$

$\hat{I}(\hat{\theta}_{r^\dagger}) = -E[\frac{\partial^2 \hat{\mathcal{L}}}{\partial\theta_a \partial\theta_b}]$

$\text{pvals}_{r^\dagger} = chi2sf(\hat{\theta}_{r^\dagger}/diag([\hat{I}(\hat{\theta}_{r^\dagger})]^{-1}))$ ▷ Compute $\mathcal{X}^2_{df=1}$ survival function for each parameter in $r^\dagger$

$\text{BIC}_{r^\dagger} = -2\hat{L}_{r^\dagger} + \ln(n) \times \left|r^\dagger\right|$

return $\hat{\theta}_{r^\dagger};\ \hat{L}_{r^\dagger};\ \hat{I}(\hat{\theta}_{r^\dagger});\ \text{pvals}_{r^\dagger};\ \text{BIC}_{r^\dagger}$

Algorithm 4 Sub routine: Forward Model Proposal

Require: $X_c,\ \hat{R}_r,\ r,\ \gamma,\ k,\ \text{baseBIC}$

▷ Draw n_{splits} sets from data and regressor sequence randomly, each set will contain $p\%$ of the total sample count

$\{X_{c,i}, \hat{R}_{r,i}\}_{i=1,\ldots,n_{\text{splits}}} = \text{Random Samples}(X, \hat{R}_r, p\%, n_{\text{splits}})$

▷ Add regressors individually and compute indicators

for $j \notin r$ **do**

$r^\dagger = r + \{j\}$

for $i = 1, ..., n_{\text{splits}}$ **do**

$_;\ _;\ _;\ \text{pvals}^\dagger_{\text{split}}[i];\ \text{BIC}^\dagger_{\text{split}}[i] = \text{EvaluateRegressors}(X_{c,i};\ \hat{R}_{r,i};\ r^\dagger)$

end for ▷ Evaluate BIC score and p-values on the full dataset as well. Max function provides an extra safeguard against bad regressors

$_;\ _;\ _;\ \text{pvals}^\dagger_{\text{full}};\ \text{BIC}^\dagger_{\text{full}} = \text{EvaluateRegressors}(X_c;\ \hat{R}_r;\ r^\dagger)$

$\text{pval}_{\text{score}}[j] = max(median(\text{pvals}^\dagger_{\text{split}}), \text{pvals}^\dagger_{\text{full}})$

$\text{BIC}_{\text{score}}[j] = max(median(\text{BIC}^\dagger_{\text{split}}), \text{BIC}^\dagger_{\text{full}})$

end for

▷ Sort candidates by best BIC in set, and select at most k of them, subject to γ restriction

$\{j_i\} = \text{argsort}(\text{BIC}_{\text{score}}[\text{BIC}_{\text{score}} < \text{baseBIC} \wedge \text{p} - \text{value}_{\text{score}} < \gamma])[0:k]$

return $r^\dagger = r + \{j_i\},\ \{j_i\}$

Acknowledgments

We thank Prof. Gregory Randall for important discussions during the early stages of this work. This work has been supported by NIH, NSF, and DoD.

References

1. Simon HA. The architecture of complexity. In: Facets of systems science. Springer; 1991. p. 457–476.
2. Saleem AB, Ayaz A, Jeffery KJ, Harris KD, Carandini M. Integration of visual motion and locomotion in mouse visual cortex. Nature Neuroscience. 2013;16(12):1864–1869.
3. Ayaz A, Saleem AB, Schölvinck ML, Carandini M. Locomotion controls spatial integration in mouse visual cortex. Current Biology. 2013;23(10):890–894.
4. Lee AM, Hoy JL, Bonci A, Wilbrecht L, Stryker MP, Niell CM. Identification of a brainstem circuit regulating visual cortical state in parallel with locomotion. Neuron. 2014;83(2):455–466.
5. Ringach DL, Sapiro G, Shapley R. A subspace reverse-correlation technique for the study of visual neurons. Vision Research. 1997;37(17):2455–2464.
6. Nauhaus I, Nielsen KJ, Callaway EM. Nonlinearity of two-photon Ca2+ imaging yields distorted measurements of tuning for V1 neuronal populations. Journal of Neurophysiology. 2011;107(3):923–936.
7. Bonin V, Histed MH, Yurgenson S, Reid RC. Local diversity and fine-scale organization of receptive fields in mouse visual cortex. Journal of Neuroscience. 2011;31(50):18506–18521.
8. Jones JP, Palmer LA. An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. Journal of Neurophysiology. 1987;58(6):1233–1258.
9. Packer AM, Peterka DS, Hirtz JJ, Prakash R, Deisseroth K, Yuste R. Two-photon optogenetics of dendritic spines and neural circuits. Nature Methods. 2012;9(12):1202–1205.
10. Packer AM, Russell LE, Dalgleish HW, Häusser M. Simultaneous all-optical manipulation and recording of neural circuit activity with cellular resolution in vivo. Nature Methods. 2015;12(2):140–146.
11. Nykamp DQ. A stimulus-dependent connectivity analysis of neuronal networks. Journal of mathematical biology. 2009;59(2):147.
12. Nelder JA, Baker RJ. Generalized Linear Models. Wiley Online Library; 1972.
13. Truccolo W, Eden UT, Fellows MR, Donoghue JP, Brown EN. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. Journal of neurophysiology. 2005;93(2):1074–1089.
14. Ringach DL, Mineault PJ, Tring E, Olivas ND, Garcia-Junco-Clemente P, Trachtenberg JT. Spatial clustering of tuning in mouse primary visual cortex. Nature Communications. 2016;7.
15. Malone BJ, Ringach DL. Dynamics of tuning in the Fourier domain. Journal of Neurophysiology. 2008;100(1):239–248.
16. Mineault PJ, Tring E, Trachtenberg JT, Ringach DL. Enhanced spatial resolution during locomotion and heightened attention in mouse primary visual cortex. Journal of Neuroscience. 2016;36(24):6382–6392.

17. Berens P, Freeman J, Deneux T, Chenkov N, McColgan T, Speiser A, et al. Community-based benchmarking improves spike inference from two-photon calcium imaging data. bioRxiv. 2017; p. 177956.

18. Schwarz G. Estimating the dimension of a model. The Annals of Statistics. 1978;6(2):461–464.

19. Harrell FE, Lee KL, Califf RM, Pryor DB, Rosati RA. Regression modelling strategies for improved prognostic prediction. Statistics in Medicine. 1984;3(2):143–152.

20. Le Cam L. Asymptotic methods in statistical decision theory. Springer Science & Business Media; 2012.

21. Lehmann EL, Casella G. Theory of Point Estimation. Springer Science & Business Media; 2006.

22. Cramér H. Mathematical Methods of Statistics (PMS-9). vol. 9. Princeton University Press; 2016.

23. Rao CR. Information and the Accuracy Attainable in the Estimation of Statistical Parameters. In: Breakthroughs in Statistics. Springer; 1992. p. 235–247.

24. Lewi J, Butera R, Paninski L. Real-time adaptive information-theoretic optimization of neurophysiology experiments. In: Advances in Neural Information Processing Systems; 2007. p. 857–864.

25. John GH, Kohavi R, Pfleger K. Irrelevant features and the subset selection problem. In: Machine learning: Proceedings of the Eleventh International Conference; 1994. p. 121–129.

26. Miller A. Subset Selection in Regression. CRC Press; 2002.

27. Symonds MR, Moussalli A. A brief guide to model selection, multimodel inference and model averaging in behavioural ecology using Akaike's information criterion. Behavioral Ecology and Sociobiology. 2011;65(1):13–21.

28. Posada D, Buckley TR, Thorne J. Model Selection and Model Averaging in Phylogenetics: Advantages of Akaike Information Criterion and Bayesian Approaches Over Likelihood Ratio Tests. Systematic Biology. 2004;53(5):793–808. doi:10.1080/10635150490522304.

29. Burnham KP, Anderson DR. Multimodel inference: understanding AIC and BIC in model selection. Sociological Methods & Research. 2004;33(2):261–304.

30. Candes E, Tao T. The Dantzig selector: Statistical estimation when p is much larger than n. The Annals of Statistics. 2007; p. 2313–2351.

31. George EI, McCulloch RE. Variable selection via Gibbs sampling. Journal of the American Statistical Association. 1993;88(423):881–889.

32. Akaike H. Information theory and an extension of the maximum likelihood principle. In: Selected Papers of Hirotugu Akaike. Springer; 1998. p. 199–213.

33. Tibshirani R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society Series B (Methodological). 1996; p. 267–288.

34. McCullagh P. Generalized linear models. European Journal of Operational Research. 1984;16(3).

35. Watts DJ, Strogatz SH. Collective dynamics of 'small-world'networks. Nature. 1998;393(6684):440–442.

36. Sporns O, Zwi JD. The small world of the cerebral cortex. Neuroinformatics. 2004;2(2):145–162.

37. Yu S, Huang D, Singer W, Nikolić D. A small world of neuronal synchrony. Cerebral Cortex. 2008;18(12):2891–2901.

38. Bassett DS, Bullmore E. Small-world brain networks. The Neuroscientist. 2006;12(6):512–523.

39. Ramkumar P. Pyglmnet; 2017. `https://github.com/glm-tools/pyglmnet`, commit = 962b6dc.

40. Schneidman E, Berry MJ, II RS, Bialek W. Weak pairwise correlations imply strongly correlated network states in a neural population. Nature. 2006;440(7087):1007.

41. Bettencourt LM, Stephens GJ, Ham MI, Gross GW. Functional structure of cortical neuronal networks grown in vitro. Physical Review E. 2007;75(2):021915.

42. Pearl J. Causality. Cambridge University Press; 2009.

43. Friedman N, Linial M, Nachman I, Pe'er D. Using Bayesian networks to analyze expression data. Journal of computational biology. 2000;7(3-4):601–620.

44. Natale JL, Hofmann D, Hernández DG, Nemenman I. Reverse-engineering biological networks from large data sets. arXiv preprint arXiv:170506370. 2017;.

45. Park IM, Meister ML, Huk AC, Pillow JW. Encoding and decoding in parietal cortex during sensorimotor decision-making. Nature Neuroscience. 2014;17(10):1395–1403.

46. Shababo B, Paige B, Pakman A, Paninski L. Bayesian inference and online experimental design for mapping neural microcircuits. In: Advances in Neural Information Processing Systems; 2013. p. 1304–1312.

47. Mishchencko Y, Vogelstein JT, Paninski L. A Bayesian approach for inferring neuronal connectivity from calcium fluorescent imaging data. The Annals of Applied Statistics. 2011; p. 1229–1261.

48. Lewi J, Butera R, Paninski L. Efficient active learning with generalized linear models. In: Artificial Intelligence and Statistics; 2007. p. 267–274.

49. Paninski L, Pillow J, Lewi J. Statistical models for neural encoding, decoding, and optimal stimulus design. Progress in Brain Research. 2007;165:493–507.

50. Pfau D, Pnevmatikakis EA, Paninski L. Robust learning of low-dimensional dynamics from large neural ensembles. In: Advances in Neural Information Processing Systems; 2013. p. 2391–2399.

51. Stevenson IH, Rebesco JM, Miller LE, Körding KP. Inferring functional connections between neurons. Current Opinion in Neurobiology. 2008;18(6):582–588.

52. Kim S, Putrino D, Ghosh S, Brown EN. A Granger causality measure for point process models of ensemble neural spiking activity. PLoS Computational Biology. 2011;7(3):e1001110.

53. Cho H, Berger B, Peng J. Reconstructing causal biological networks through active learning. PloS One. 2016;11(3):e0150611.

54. MacKay DJ. Information-based objective functions for active data selection. Neural Computation. 1992;4(4):590–604.

55. Lewi J, Butera R, Paninski L. Sequential optimal design of neurophysiology experiments. Neural Computation. 2009;21(3):619–687.

56. Bölinger D, Gollisch T. Closed-loop measurements of iso-response stimuli reveal dynamic nonlinear stimulus integration in the retina. Neuron. 2012;73(2):333–346.

57. Benda J, Gollisch T, Machens CK, Herz AV. From response to stimulus: adaptive sampling in sensory physiology. Current opinion in neurobiology. 2007;17(4):430–436.

58. Edin F, Machens CK, Schütze H, Herz AV. Searching for optimal sensory signals: iterative stimulus reconstruction in closed-loop experiments. Journal of computational neuroscience. 2004;17(1):47–56.

59. DiMattina C, Zhang K. Active data collection for efficient estimation and comparison of nonlinear neural models. Neural computation. 2011;23(9):2242–2288.

60. DiMattina C, Zhang K. Adaptive stimulus optimization for sensory systems neuroscience. Frontiers in neural circuits. 2013;7:101.

61. Pillow JW, Park M. Adaptive Bayesian methods for closed-loop neurophysiology. Closed Loop Neuroscience (AE Hady, ed), Elsevier. 2016;.

62. Park M, Pillow JW. Bayesian active learning with localized priors for fast receptive field characterization. In: Advances in neural information processing systems; 2012. p. 2348–2356.

63. Yang J, Mandal A. D-optimal factorial designs under generalized linear models. Communications in Statistics-Simulation and Computation. 2015;44(9):2264–2277.

64. Koch C, Segev I. Methods in neuronal modeling: from ions to networks. MIT press; 1998.