NEURAL NETWORK POISSON MODELS FOR BEHAVIOURAL AND NEURAL SPIKE TRAIN DATA

Moein Khajehnejad*
Monash University

Forough Habibollahi*
The University of Melbourne

Richard Nock
Google Research

Ehsan Arabzadeh
Australian National University

Peter Dayan
MPI for Biological Cybernetics and the University of Tübingen

Amir Dezfouli
Data61, CSIRO

ABSTRACT

One of the most important and challenging application areas for complex machine learning methods is to predict, characterize and model rich, multi-dimensional, neural data. Recent advances in neural recording techniques have made it possible to monitor the activities of a large number of neurons across different brain regions as animals perform behavioural tasks. This poses the critical challenge of establishing links between neural activity at a microscopic scale, which might for instance represent sensory input, and at a macroscopic scale, which then generates behaviour. Predominant modeling methods apply rather disjoint techniques to these scales; by contrast, we suggest an end-to-end model which exploits recent developments of flexible, but tractable, neural network point-process models to characterize dependencies between stimuli, actions, and neural data. We apply this model to a public dataset collected using Neuropixel probes in mice performing a visually-guided behavioural task as well as a synthetic dataset produced from a hierarchical network model with reciprocally connected sensory and integration circuits intended to characterize animal behaviour in a fixed-duration motion discrimination task. We show that our model outperforms previous approaches and contributes novel insights into the relationships between neural activities and behaviour.

1 Introduction

Recent developments in neural recording techniques such as Neuropixel probes allow the activities of large numbers of neurons across the brain to be monitored as animals perform behavioural tasks [Jun et al. 2017]. This allows us to study how the brain represents past and present sensory inputs across areas, how these representations evolve over time and ultimately lead to behaviour.

Very coarsely, supervised, reinforcement learning and unsupervised methods have been applied to examine the relationships between activity and behaviour [Paninski, Pillow, and Lewi 2007; Mante et al. 2013; Ganguli and Sompolinsky 2012; Kass, Eden, and Brown 2014; Sussillo 2014; Richards et al. 2019; Schaeffer et al. 2020]. Encoding and decoding models are examples of supervised learning. Encoding models use linear or non-linear methods to predict the activity of individual neurons based on task-related variables such as stimuli, actions, rewards and the like. Decoding models use linear or non-linear methods to predict the values of these task-related variables from the conjoint activities of multiple neurons within or between areas. Both types of model have been hugely influential from the earliest days of the application of computational methods to understand neural representation and processing [Dayan and Abbott 2001; Rieke et al. 1999; Kass, Eden, and Brown 2014; Meyer et al. 2017]. However, simple reflections of the computational constraints of the task are often insufficient to capture complex neural representations of the sensory inputs and actions that are distributed across different brain regions [Steinmetz et al. 2019] and can also evolve over time in 'null' neural

*Authors contributed equally to this manuscript.

modes that have no behavioural consequence. Moreover, neural responses are variable and a population's response will often differ from trial to trial and over time, even under the same experimental conditions [Shadlen and Newsome 1998]. Most traditional approaches do not generalize to these more naturalistic conditions where trials with identical stimuli do not exhibit identical behavioural schemes [Goris, Movshon, and Simoncelli 2014; Churchland et al. 2010]. Hence, they can not account for the temporal irregularities in behaviour and neural recordings between trials.

Reinforcement learning (or in some cases, supervised learning) methods have more recently been used to learn potentially complex feedforward and recurrent neural network (RNN) models that are themselves capable of performing the same behavioural task as the subjects, based on assumptions about sensory noise and processing architecture [Barak 2017; Sussillo 2014; Richards et al. 2019; Schaeffer et al. 2020; Yamins et al. 2014]. These have been highly revealing about neural coding. However, they are also ill-suited to capture the myriad complexities of null modes, or the particular sub-optimalities expressed by individual subjects, reflecting their particular incompetence, training history and more.

Unsupervised methods have also been applied – often ways of mapping very high dimensional population activity into lower dimensional spaces [Paninski et al. 2010; Cunningham and Yu 2014; Whiteway and Butts 2019; Yu et al. 2006]; with the structure in these spaces, and perhaps the dynamical evolution of the states in these spaces, subsequently being related to task variables. These methods are typically useful since the number of dimensions of task input and/or output variability is often rather modest, implying that much of the high dimensional space that could potentially be occupied is either empty or at least not relevant for behaviour. However, they typically have to use intrinsic metrics such as variance to specify which low dimensional projections should be considered – and this again begs the question as to what is important.

Of course, there are methods that combine various of these approaches [Kobak et al. 2016; Kriegeskorte and Kievit 2013], and continual innovations. Along these lines, [Dezfouli et al. 2018] recently suggested a combined approach, with an RNN being trained to tie fMRI BOLD activity across the brain directly with ongoing behaviour. fMRI data allowed for a form of model inversion, pinning down the RNN state and so implying how behaviour would be realized neurally. However, this approach is licensed by the invertibility that is at least plausible because of the high dimensionality of fMRI. It is not currently guaranteed to be available on a trial-by-trial basis in neural recordings.

Here, motivated by previous approaches and recent neural network point-process models [Omi, Aihara, et al. 2019], we suggest a novel neural network Poisson process model which: (i) flexibly learns the connections between environmental stimuli and neural representations, and between neural representations and behavioural responses; (ii) jointly fits both behavioural and neural data; (iii) handles variabilities between response times in different trials of an experiments by a temporal rescaling mechanism, and (iv) derives spike count statistics disentangled from chosen temporal bin sizes. The framework allows efficient training of the model without making assumptions about the functional form of the relationship between input stimuli and neural and behavioural processes. We apply the method to two neural/behavioural datasets concerning visual discrimination tasks: one collected using Neuropixel probes [Steinmetz et al. 2019] from mice, and the other the output of a hierarchical network model with reciprocally connected sensory and integration circuits that was designed to model behaviour in a motion-based task [Wimmer et al. 2015]. We show that our method is able in both cases to link behavioural data with their underlying neural processes and input stimuli; the synthetic dataset allows us to compare our results against ground truth.

2 The model

Data description. We model canonical visual discrimination experiments. In our case, on each trial, subjects are presented with a stimulus and have to choose an option (or keep still; NoGo). We consider two datasets in this setting. The first is the visual discrimination experiment of [Steinmetz et al. 2019] (Figure 1a). On each trial, mice are presented with a stimulus (visual contrast on the left or right side) and have to make a simple response by turning a wheel left or right or keeping it still. The second dataset is synthetic and based on the work of [Wimmer et al. 2015] (Figure 1b). A hierarchical spiking neural network model is built to capture the essence of evidence integration and decision-making of monkeys in a standard two-alternative forced-choice motion discrimination task [Gold and Shadlen 2007; Britten et al. 1996].

Formalisation. The total number of trials in an experiment is denoted by $|\mathcal{N}|$; the stimuli on trial $n \in \{1 \dots |\mathcal{N}|\}$ are (generically) denoted by vector $\mathbf{x}_n$. If a response was made on trial n (at a time relative to stimulus onset we call r_n), we denote it by $a_n \in \mathcal{A}$ (where, here, $\mathcal{A} = \{\text{LEFT}, \text{RIGHT}\}$. After an observation window of $W = 400$ms (for the Steinmetz dataset; same as the window size chosen in [Steinmetz et al. 2019]) and $W = 2000$ms (for the synthetic data) expires, then we consider the subject to have chosen NoGo.

Different neurons in different areas and even different animals (in the Steinmetz dataset) may be recorded and contribute in separate trials. In total, $|\mathcal{S}_{u,n}|$ spikes are observed from unit u at times $\{s^i_{u,n}\}_{i=1\ldots|\mathcal{S}_{u,n}|}$, relative to stimulus onset in the corresponding trials.

In the following, we first discuss how we model the neural data; and then how we couple this model to predict behaviour. Figure 2 provides an overview of the designed framework. For more details on the model architecture, please refer to Supplementary Materials.

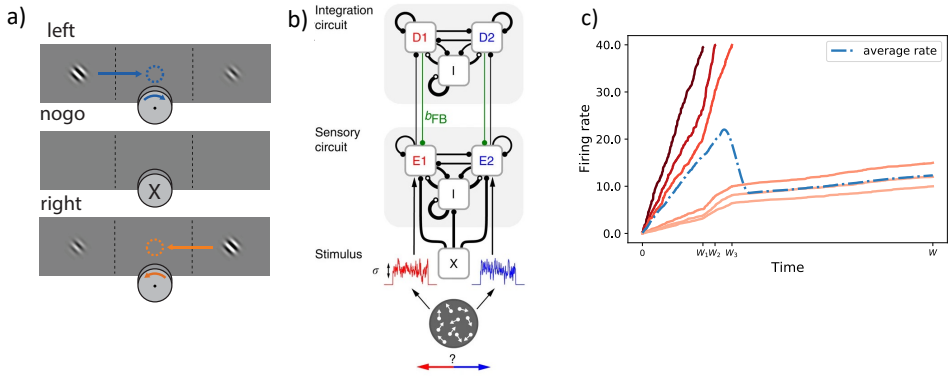


Figure 1: a) Steinmetz' visual discrimination task [Steinmetz et al. 2019]. On each trial, visual stimuli of different contrasts were potentially presented on the left and right sides. If both sides had zero contrast, the mouse earned reward by NoGo. If both had equal, non-zero, contrast, it was rewarded at random. Otherwise, it was rewarded for reporting ($a \in \{\text{LEFT}, \text{RIGHT}\}$) which contrast was larger. The figure is adapted from [Steinmetz et al. 2019]. **b)** Synthetic network model illustrating the sensory (E1; E2) and integrator (D1; D2) circuits enjoying feed-forward and top-down feedback connections as well as lateral excitatory and inhibitory (population I) recurrent connections within each circuit. The figure is adapted from [Wimmer et al. 2015]. **c)** Illustrative example illustrating the occurrence of systematic bias in firing rate estimation when aggregating trials with different end times. Red curves show firing rates for six sample trials. The dashed line shows the average of firing rates based on the unfinished trials at each point in time. Looking at the blue curve, one might conclude that neural activities increase and then decrease over times, which is not true for any individual trial.

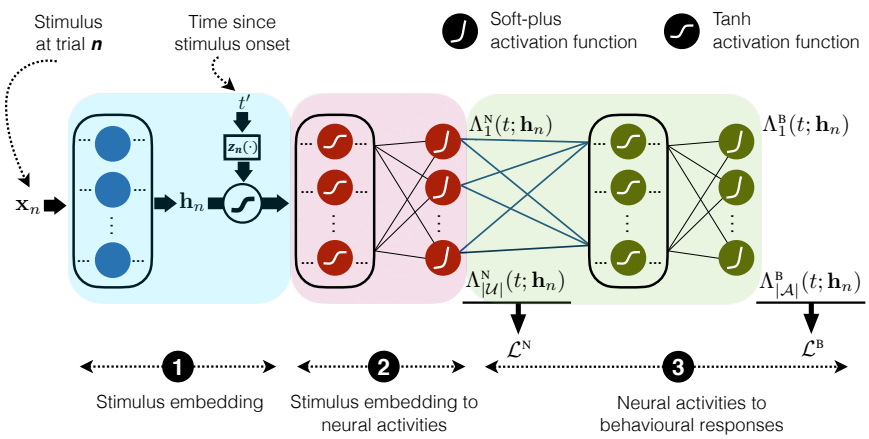


Figure 2: Network architecture. ① Embedding of input stimulus into vector $\mathbf{h}_n$. ② Transformation of embedding $\mathbf{h}_n$ into neural activity of each region for each time point t since the stimulus onset. The monotonic transformation function , $z_n(t') = t\cdot$, is applied to the input spike time series in this step. Neural activities are characterised by the rescaled *cumulative* intensity function of spike train for each region, denoted by $\Lambda^{\mathrm{N}}_u(t;\mathbf{h}_n)$ for regions $u \in \mathcal{U}$. The intensity function $\lambda^{\mathrm{N}}_u(t;\mathbf{h}_n)$ is obtained by differentiating the cumulative intensity function $\Lambda^{\mathrm{N}}_u(t;\mathbf{h}_n)$ with respect to t. Component ② structurally ensures that $\Lambda^{\mathrm{N}}_u(t;\mathbf{h}_n)$ increases with time, so $\lambda^{\mathrm{N}}_u(t;\mathbf{h}_n) \geq 0$. ③ Neural activities are mapped to behavioural responses which are represented by the rescaled cumulative intensity function $\Lambda^{\mathrm{B}}_a(t;\mathbf{h}_n)$ for making each action $a \in \mathcal{A}$ at each time t since stimulus onset (t). The data likelihood is computed over both recorded spike trains and behavioural responses. This yields a neural log-likelihood function ($\mathcal{L}^{\mathrm{N}}$) and a behavioural log-likelihood function ($\mathcal{L}^{\mathrm{B}}$).

2.1 Spike train models

We make the simplification that the spikes of each neuron u in trial n can be modelled as the output of an inhomogeneous Poisson process [Daley and Vere-Jones 2006] with a latent intensity function $\lambda^{\text{N}}_{u,n}(\tau; \mathbf{h}_n)$ (the superscript N indicates that the intensity function is for the Neural data). This can be interpreted as the instantaneous probability of observing a spike at time τ, where $\mathbf{h}_n$ is a function of the stimulus $\mathbf{x}_n$. The Poisson process assumption is that successive spike times are independent, given $\lambda^{\text{N}}_{u,n}(\tau; \mathbf{h}_n)$. Note that we seek to capture signal correlations but not noise correlations, and so $\mathbf{h}_n$ does not depend on the spikes observed during a trial.

2.1.1 Time rescaling of spike trains

We consider spikes from neuron u until either a response a_n was made at time r_n, or to the end of time window W, whichever comes first, i.e., up to $W_n = \min(W, r_n)$. The reason for restricting the observation period to r_n is because the aim is to model the neural processes that lead to behavioural responses, rather than what happens post-response. The joint probability density of observing spike trains from neuron u in trial n is then,

$$f^{\text{N}}_{u,n}(s^1_{u,n} \dots s^{|\mathcal{S}^{u,n}|}_{u,n}) = \prod_{i=1}^{|\mathcal{S}_{u,n}|} \lambda^{\text{N}}_{u,n}(s^i_{u,n}; \mathbf{h}_n) \exp\left(-\int_0^{W_n} \lambda^{\text{N}}_{u,n}(\tau'; \mathbf{h}_n) d\tau'\right). \tag{1}$$

Intuitively, the term $\lambda^{\text{N}}_{u,n}(s^i_{u,n}; \mathbf{h}_n)$ represents the probability density of observing a spike at time $s^i_{u.n}$ and the exponential term represents the probability of not observing spikes at other times in the observation period. We aim to estimate a single function $\lambda^{\text{N}}_{u,n}(\tau; \mathbf{h}_n)$ to model the neural activities across all trials. However, note that the duration of trials can be different (based on response times) and only trials that ended *after* τ can contribute to the estimation of $\lambda^{\text{N}}_{u,n}(\tau; \mathbf{h}_n)$, which means that this quantity is implicitly conditioned on $r_n > \tau$. This property makes the interpretation of $\lambda^{\text{N}}_{u,n}(\tau; \mathbf{h}_n)$ rather inconvenient since it will no longer represent how neural activities evolve over time, but is confounded by the distribution of response times (see Fig 1c). To address this issue, it is tempting to merely condition $\lambda^{\text{N}}_{u,n}(\tau; \mathbf{h}_n)$ on response times (in addition to $\mathbf{h}_n$) to get a picture of the spike trains that lead to each specific response time (rather than all the response times after τ). This, however, only partially addresses the issue. To address it more fully, we aim to map all trials with different duration to the same time span. To achieve this goal, we propose the following theorem and proposition:

Theorem 1. *Let $0 < s'^1 < s'^2 <, \dots, < s'^j \leq W_n \leq W$ be a realization from an inhomogeneous Poisson point process, n, with an intensity function $\lambda_n(t')$ satisfying $0 < \lambda_n(t')$ for all $t' \in (0, W_n]$. Define a one-to-one monotonic transformation function, where:*

$$z_n : [0, W_n] \to [0, W], \textit{and } z_n(0) = 0, z_n(W_n) = W$$

Assume $0 < s^1 < s^2 <, \dots, < s^j \leq W$ where $\forall k \in \{1, \dots, j\}$; $s^k = z_n(s'^k)$. Then s^k are a realization from a second inhomogeneous Poisson point process with $\lambda(t) = \lambda_n(t')$ where $t = z_n(t')$.

Proposition 1. *For a linear transformation function $z_n(.)$, as defined in* **Theorem 1.**, *the cumulative intensity function of the original and second point process realizations are related as: $\Lambda(t) = \frac{1}{\partial_t(z^{-1})} \cdot \Lambda_n(t')$.*

Note that **Theorem 1.** is a special case under the Mapping Theorem [Grimmett and Stirzaker 2020]. Please see Supplementary Materials for proofs.

2.1.2 Parametrising the intensity function

Here, we define $\lambda^{\text{N}}_u(t; \mathbf{h}_n)$ to represent canonical neural activities (prior to the response) defined over $t \in [0, W]$. Based on the above theorem, the neural activities for a certain trial with duration W_n can be obtained by applying the time rescaling on the original spike time series using a monotonic function $z_n : [0, W_n] \to [0, W]$, $t = z_n(t')$,

$$\lambda^{\text{N}}_{u,n}(t'; \mathbf{h}_n) = \lambda^{\text{N}}_u(z_n(t'); \mathbf{h}_n), \text{ with } z_n(0) = 0, z_n(W_n) = W, \ t' \in [0, W_n]. \tag{2}$$

The dependence of the intensity function on the embedding $\mathbf{h}_n$ plays a crucial role in determining how neural activities are shaped by the stimulus and has to be characterized in a flexible manner. To achieve this, one option is to use a multi-layer feed-forward network which takes t and $\mathbf{h}_n$ as inputs, and outputs $\lambda^{\text{N}}_u(t; \mathbf{h}_n) \geq 0$. Unfortunately, it is then intractable to calculate the integral in equation 1. An elegant solution to this problem is to parameterize the cumulative

intensity function $\Lambda_u^{\text{N}}(t; \mathbf{h}_n)$ instead of $\lambda_u^{\text{N}}(t; \mathbf{h}_n)$ [Omi, Aihara, et al. 2019]. This is

$$\Lambda_u^{\text{N}}(t; \mathbf{h}_n) = \int_0^t \lambda_u^{\text{N}}(\tau; \mathbf{h}_n) d\tau, \tag{3}$$

and can be (automatically) differentiated to produce the intensity function:

$$\lambda_u^{\text{N}}(t; \mathbf{h}_n) = \frac{\partial \Lambda_u^{\text{N}}(t; \mathbf{h}_n)}{\partial t}. \tag{4}$$

We represent $\Lambda_u^{\text{N}}(t; \mathbf{h}_n)$ using a feed-forward network. Note that the gradient of the cumulative intensity function w.r.t to t must always be non-negative (see Section Model structure). In this work, we chose time to be rescaled uniformly using $z_n(t') = t'W/W_n$. This makes a substantive assumption that the activities in slow and fast trials are stretched versions of each other. Its advantage is to reduce the task of learning trial-specific intensities to learning the canonical intensity function $\lambda_u^{\text{N}}(t; \mathbf{h}_n)$. Based on Theorem 1. and Proposition 1., trial specific intensity and cumulative intensity functions (defined on $t' \in [0, W_n]$) are then given by,

$$\lambda_{u,n}^{\text{N}}(t'; \mathbf{h}_n) = \lambda_u^{\text{N}}(z_n(t'); \mathbf{h}_n), \; \Lambda_{u,n}^{\text{N}}(t'; \mathbf{h}_n) = \frac{W_n}{W} \cdot \Lambda_u^{\text{N}}(z_n(t'); \mathbf{h}_n). \tag{5}$$

The relatively simple form of transformed cumulative intensities is a consequence of the uniform time rescaling (see Supplementary Materials for proof). Using these two related functions, the data log-likelihood implied by equation 1 is

$$\mathcal{L}_{u.n}^{\text{N}} = \log f_{u.n}^{\text{N}}(s_{u.n}^1 \dots s_{u.n}^{|\mathcal{S}^{u.n}|}) = \sum_{i=1}^{|\mathcal{S}_{u.n}|} \left[\log \frac{\partial \Lambda_u^{\text{N}}(t = z_n(s_{u.n}^i); \mathbf{h}_n)}{\partial t} \right] - \frac{W_n}{W} \cdot \Lambda_u^{\text{N}}(W; \mathbf{h}_n), \tag{6}$$

which retains the required flexibility, while obviating the calculation of the intractable integral. The total data likelihood for all the trials and neurons is then

$$\mathcal{L}^{\text{N}} = \sum\nolimits_{t=1}^{|\mathcal{N}|} \sum\nolimits_{u \in \mathcal{U}_n} \mathcal{L}_{u.n}^{\text{N}}.$$

In principle, each recorded neuron in the experiment could be assigned a separate intensity function. However, given the experimental methodology of only recording some neurons on some trials in the Steinmetz dataset, the problem of missing data would be radically acute, and the model would be uninterpretable. The computational cost would also be prohibitive. Instead, we make the simplifying assumptions that all the neurons in each of the 42 brain regions identified by [Steinmetz et al. 2019] and (as is true by design) the 4 regions in the synthetic model share common intensity functions. As such, we consider $|\mathcal{U}_{\text{Stein}}| = 42$ and $|\mathcal{U}_{\text{synth}}| = 4$.

2.2 Behavioural response models

Having provided a way of characterizing the neural response to the embedding $\mathbf{h}_n$, we next need to model the link between neural representations and behaviour. We assume that the probability of making a behavioural response at each point in time depends on the activity of the neurons at that time. In turn, these are driven by the stimulus $\mathbf{h}_n$ on that trial. That is, the behavioural responses are indirectly affected by the stimulus via neural activities. However, rather than model this dependence explicitly, which is hard given the punctate nature of the response, we approximate it implicitly, via smooth intensity functions that in turn depend on Λ^{N}.

The intensity function for an action a is denoted by $\lambda_a^{\text{B}}(t; \mathbf{h}_n)$, which specifies the instantaneous probability of taking the action at time t relative to stimulus onset. The superscript B indicates that the intensity function is for the Behavioural data. A key simplification is to allow for the theoretical possibility that the animal performs the same action more than once on a trial; or performs both actions. However, that actions are actually sparse implies that this approximation is not too costly. We write the canonical behavioural cumulative intensity function as a function of the canonical neural cumulative intensity functions.

$$\Lambda_a^{\text{B}}(t; \mathbf{h}_n) = \Phi_a(\Lambda_1^{\text{N}}(t; \mathbf{h}_n), \dots, \Lambda_{|\mathcal{U}|}^{\text{N}}(t; \mathbf{h}_n)). \tag{7}$$

Function $\Phi_a(.)$ can be realised using a deep feed-forward network which can represent arbitrary dependencies between neural activities and behavioural responses.

Then, differentiating:

$$\lambda_a^{\text{B}}(t; \mathbf{h}_n) = \sum\nolimits_{u=1}^{|\mathcal{U}|} \lambda_u^{\text{N}}(t; \mathbf{h}_n) \frac{\partial \Phi_a(.)}{\partial \Lambda_u^{\text{N}}}, \tag{8}$$

which implies that the behavioural response probability at each point in time is indirectly dependent on the stimulus through the spike rate of different neural activities, as desired. Function $\Phi(.)$ is also designed to be increasing in t to ensure that $\lambda_a^{\text{B}}(t;\mathbf{h}_n) > 0$ (see Section Model structure).
These response rates are presented as a function of canonical neural intensity functions. Similar to 5 for each trial n we have,

$$\lambda_{a,n}^{\text{B}}(t';\mathbf{h}_n) = \lambda_a^{\text{B}}(z_n(t');\mathbf{h}_n),\ \Lambda_{a,n}^{\text{B}}(t';\mathbf{h}_n) = \frac{W_n}{W}\cdot\Lambda_a^{\text{B}}(z_n(t');\mathbf{h}_n). \tag{9}$$

The subjects can either act or not on a trial; the latter is determined by a censoring window W. Write $\mathcal{N}_a$ as the set of trials on which action a was taken before W, at reaction time r_n. The simplified joint probability distribution of the behavioural observations is then:

$$f_a^{\text{B}}(\{r_n\}_{n\in\mathcal{N}_a}) = \left(\prod\nolimits_{n\in\mathcal{N}_a}\lambda_{a,n}^{\text{B}}(r_n;\mathbf{h}_n)\right)\prod\nolimits_{n=1}^{|\mathcal{N}|}\exp\left(-\int_0^{W_n}\lambda_{a,n}^{\text{B}}(\tau';\mathbf{h}_n)d\tau'\right), \tag{10}$$

and, taking logs, the log-likelihood for those observations is,

$$\mathcal{L}_a^{\text{B}} = \sum\nolimits_{n\in\mathcal{N}_a}\log\frac{\partial\Lambda_a^{\text{B}}(t=z_n(r_n);\mathbf{h}_n)}{\partial t} - \sum\nolimits_{n=1}^{|\mathcal{N}|}\frac{W_n}{W}\cdot\Lambda_a^{\text{B}}(W;\mathbf{h}_n). \tag{11}$$

Over the whole experiment and actions, the behavioural likelihood can be defined as,

$$\mathcal{L}^{\text{B}} = \sum\nolimits_{a\in\mathcal{A}}\mathcal{L}_a^{\text{B}}, \tag{12}$$

in which $\mathcal{A}$ is the set of available actions.

2.3 Model structure

We implement the model using the neural network architecture shown in Figure 2. This has three components. The first maps the stimulus $\mathbf{x}_n$ that was presented through a series of fully connected layers to realize an input embedding denoted by $\mathbf{h}_n$. The second component takes the embedding $\mathbf{h}_n$ and t and outputs the modelled activity of each neural region u at time t in the form of cumulative intensity functions for $\Lambda_u^{\text{N}}(t;\mathbf{h}_n)$. This component is designed such that the outputs of the network, i.e., $\Lambda_u^{\text{N}}(t;\mathbf{h}_n)$s, are monotonic functions of t to ensure that their gradients with respect to t (which are neural intensity functions) are always positive. To achieve this, following ideas from [Sill 1998; Chilinski and Silva 2018; Omi, Aihara, et al. 2019], the weights of the network are constrained to be positive and tanh activation functions are used in the middle layers and soft-plus in the output layers.
The third component of the model takes the neural cumulative intensity functions and maps them to the behavioural cumulative intensity functions (function Φ in equation 7). We used the same method as for the second component to ensure that the gradient of $\Lambda_a^{\text{B}}(t;\mathbf{h}_n)$ with respect to t is positive.
For training the model, the neural loss function $\mathcal{L}^{\text{N}}$ is used to train all the weights from stimulus to neural cumulative intensity functions (blue and red rectangles in Figure 2). Given these trained neural cumulative intensity functions, then the weights connecting neural outputs to behavioural outputs are trained using $\mathcal{L}^{\text{B}}$. The gradients were obtained using automatic differentiation in Tensorflow [Abadi et al. 2015]. See Supplementary Materials for details.

3 Results

3.1 Data structure

Synthetic dataset. For the synthetic data, we use data generated from the model introduced in [Wimmer et al. 2015]. Activities from two direction-selective sensory regions (E1 and E2; e.g. V5/MT) as well as two integrator regions (D1 and D2; e.g. LIP, FEF) are modeled and then observed. Each region has 240 neurons and the whole experiment consists of 1800 trials (1200 trials for training and 600 trials for testing). Left and right sensory regions prefer leftwards and rightwards motion respectively; time-varying activity in these regions inspired by stimuli with coherence levels varying from completely obscure: 0%, to rather definite: 50% or 80% (which are encoded using one-hot encoding), are accumulated by populations in the integration region. The latter has attractor dynamics; a response is realized when the activity state is sufficiently close to one of the two attractors. A choice is modelled as being made when the average activity of a (trial-)random subset of neurons in D1 or D2 over a window of 50ms reaches 40Hz. This corresponds to

strong evidence in favour of motion in the corresponding direction which can be RIGHT (for D1) or LEFT (for D2).

Steinmetz dataset. We use the data reported in [Steinmetz et al. 2019][2]. The experiments consist of 38 sessions of a visual discrimination task (Figure 1a). Activities from 42 brain regions from the left hemisphere of the brain were recorded (not all regions were recorded in all the sessions). Overall the data from 30,000 neurons were recorded and the whole experiment consisted of 10011 trials. We used the data from 12 sessions for testing and the rest for training the model.
On each trial, the animals were presented with stimuli on the left and right and were required to turn a wheel LEFT, RIGHT or keep it fixed (NoGo), based on the contrast input (four possible levels of contrast on each side: 0, 25, 50, 100%; 0% on both sides requires NoGo). We encoded stimulus contrast using one-hot encoding based on which side had a higher contrast, or whether they had equal contrasts ($\mathbf{x}_n$ of dimension 3). The reaction time r_n corresponds to the beginning of the wheel turn if this happens before the end of the response window.

3.2 Training process

All the weights in the model were trained using the Adam optimiser [Kingma and Ba 2014]. Stimulus and integrator components (blue and red rectangles in Figure 2) each were composed of three fully connected hidden layers with 20 neurons in each layer. The third layer was then connected to the output layer consisting of one neuron per region in the dataset. The neural component was followed by two fully connected behavioural layers for each action (with 10 neurons). 'softplus' and 'tanh' activation functions were used to ensure positive of intensity functions. See Supplementary Materials for more details about the model architecture and training process.

3.3 Experiments

We show statistics of the quality of the fit of the model later. However, we first illustrate the neural and behavioural properties of the model by freezing the weights and performing simulations with different values of W_n for sample regions of both synthetic and Steinmetz datasets. Results showing the performance of the model on all the available test regions of both datasets are presented in Supplementary Materials.
The solid lines in the upper panels of Figure 3 illustrate the neural responses for the synthetic dataset for $0.3 \leq W_n \leq 0.6$ when the stimulus had highest coherence level (0.8) and moved RIGHT. The chosen interval includes more than 90% of all trials in the coherence level of 0.8. These results are compared with the empirical activity derived from the data (dashed lines) for both integrator regions D1 and D2. Note the change in y-scale between the plots, given the strong left stimulus. These values are closely related to the learned intensities and capture the observed variability in response times between different trials given the time rescaling of input spike trains.
Figure 3 lower panels show the mean rate of behavioural responses for each interval of learned neural intensities for the complete set of trials. As expected, the highest rate of behavioural action is observed for neural intensities close to 40 spikes/sec for D1. By contrast, region D2 shows very low behavioural activity rates due to the non-favorable direction of stimulus in these trials. Note that the shown empirical firing rates are averaged across all responses, while the intensities correspond to different response times, which helps explain the differences between the two measures. For example, in very fast responses, there is an initial burst in the firing rates, which is captured by the initial sharp rise in the intensities (see Figure 3), but this is invisible in the averaged firing rates.
The upper panels of Figure 4 show modelled and actual neural activities in the Steinmetz dataset for two example brain regions with high contrast levels of stimulus on RIGHT: one the subiculum (SUB), which [Steinmetz et al. 2019] reported as containing neurons that consistently fired before wheel turns regardless of their direction (arguably a surprising feature of this dataset, given the relationship of the subiculum with areas such as the hippocampus and entorhinal cortex rather than motor regions), and the other, visual (VISp; primary visual area), which is reported to have the highest portion of visual encoding neurons . The activities were all recorded in the left hemisphere, and therefore, the right stimulus/action are contralateral to the recording sites. Comparing the lower and upper panels of Figure 4 show, we generally see more activity on the left side in VISp for stimuli with high contrast levels on the RIGHT which is consistent with previous analyses [Steinmetz et al. 2019]. It is clear that the region shows lower neural activity levels for ipsilateral stimuli. The subiculum (SUB), is reported in [Steinmetz et al. 2019] as containing neurons that consistently fired before wheel turns regardless of their direction. We can see in Figure 4 lower panels that this indeed is captured by our model for the SUB region where it reaches similar firing rates no matter the direction of stimulus and motion are. The relative

[2]https://github.com/nsteinme/steinmetz-et-al-2019/wiki/data-files

timing of the activities in the areas are also consistent with expectations. The solid lines in the panels show estimated firing rates, which are consistent with the data (dashed lines) in particular for VISp which had a peak in firing rate early after stimulus was presented (see below) and closely related to the presented visual stimuli in the task. Note that in our framework, estimation of the neural activities (firing rates) does not rely on selecting a temporal bin size. This is unlike most previous state-of-the-art works [Liu and Lengyel 2021] where the output firing rates are substantially affected by the choice of bin size.
Next, examining the behavioural predictions of our model, the neural activities presented in Figure 4 show that sooner responses (with high probabilities) are strongly related to the peak intensities in VISp. In agreement with these findings, middle right panel in Figure 4 shows higher probability for the occurrence of reactions when the neural activity in VISp peaks. SUB is also coupled to behaviour in the middle panel – in fact for both directions of movement.

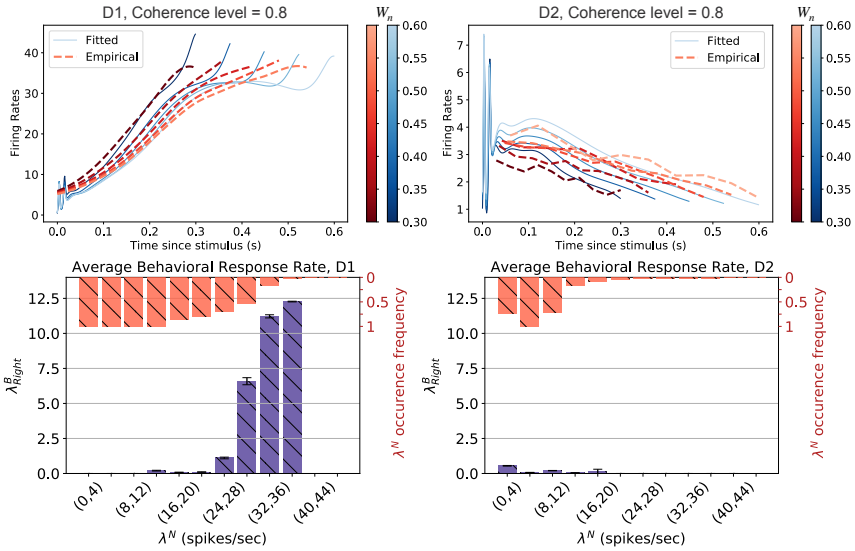


Figure 3: Synthetic Data (Upper panels). The activity rate of neural population in the two integrator regions of the synthetic dataset. The dashed lines show the empirically derived firing rates compared to the solid lines which are estimated using the proposed model. Each line plot corresponds to the average activity rate of trials with W_n in a specific interval illustrated by the colorbar (see section F in Supplementary Materials). **(Lower panels).** Illustrating the average response rate for each interval of neural activity rates. The plots correspond to the trials with highest coherence level. The plots show the average between trials with a RIGHT reaction. Error bars show the standard error. Purple bars show the average behavioural response rates. Orange bars indicate the proportion of trials with the occurrence of each neural response interval.

3.4 Baseline Methods

Finally, Table 1 shows a comparison of the performance of our proposed framework to those of recent prominent baseline point process estimators. We compare the negative log likelihoods (NLL) for the neural activity on sample regions of Steinmetz dataset as well as the 4 regions of the synthetic dataset. For details on the utilized settings for implementing the baseline methods, please see the Supplementary Materials.

- **GLM Model [Truccolo et al. 2005; Paninski 2004]**: Generalized-linear models (GLM) also known as Poisson regression, are used to model the intensity of the input data as a linear combination of time-dependent covariates. Here, the total spike counts for all trials are calculated and concatenated for count windows of 5ms as inputs to the GLM model.
- **NHPoisson Model [Cebrian 2015]**: NHPoisson is a method for the modelling non homogeneous Poisson processes in time estimating maximum likelihood. The model is based on formulating the intensity as a function of time-dependent covariates.
- **Universal Count Model [Liu and Lengyel 2021]**: This model builds on sparse Gaussian processes (GP) to capture arbitrary spike count distributions flexibly relying on both observed and latent covariates. It uses scalable variational inference and can jointly infer the covariate-to-spike count distribution mappings and latent trajectories. We also examine a second variant of this model which replaces the GP-based approaches with an artificial neural network (ANN) mapping. We denote the two variations by U-GP and U-ANN respectively.

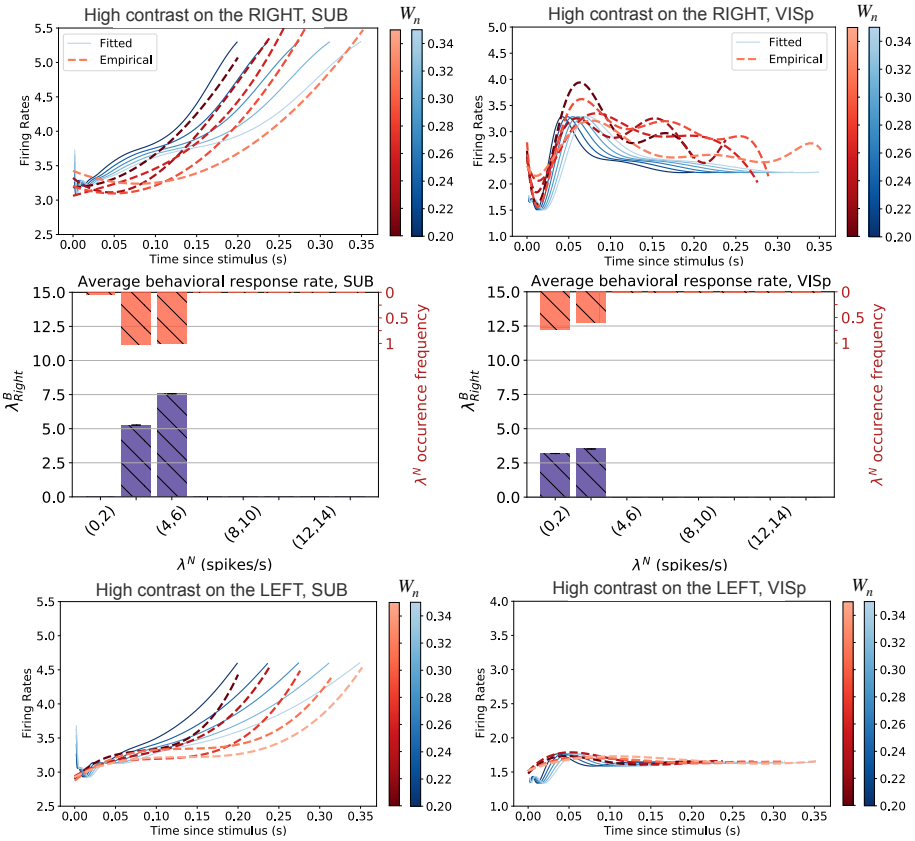


Figure 4: Steinmetz (Upper panels). The activity rate of neural population in the two sample regions of the Steinmetz dataset for the highest contrast level on RIGHT. The dashed lines show the empirically derived firing rates compared to the solid lines which are estimated using the proposed model. Each line plot corresponds to the average activity rate of trials with W_n in a specific interval illustrated by the colorbar (see section F in Supplementary Materials). **(Middle panels).** Illustrating the average response rate for each interval of neural activity rates. The plots correspond to the trials with high contrast on RIGHT. Error bars show the standard error. Purple bars show the average behavioural response rates. Orange bars indicate the proportion of trials with the occurrence of each neural response interval. **(Lower panels).** Neural activities for contralateral and ipsilateral stimuli. The VISp shows a direction selective activity pattern, preferring the contralateral stimuli on the RIGHT. However this is not the case for SUB which is in agreement with reports in [Steinmetz et al. 2019] where this region is known to contain firing neurons before the motion initiation regardless of the direction of stimulus and movement.

- **Poisson Gaussian-Process Latent Variable Model (P-GPL) [Wu et al. 2017]**: In this model, Poisson spiking observations are accompanied by two underlying Gaussian processes: One governing a temporal latent variable, the other governing a set of nonlinear tuning curves. The model learns using a decoupled Laplace approximation which is a fast approximate inference method. The same set of temporal covariates as above are also utilized in the implementation of this method.

It is important to mention that the performance of all the baseline methods depend on the length of the selected time bin for spike count calculation; a constraining dependency causing non-robust results that our proposed method overcomes. We selected the time bin length for optimal performance by performing a heuristic search in a rational range.
Note that the average spike counts per region in the Steinmetz dataset is roughly 60 times less than in the synthetic one. This is due to the difference between the lengths of the experiment and the numbers of neurons per region in the Steinmetz and synthetic dataset. Thereby, the NLL measures differ by two orders of magnitude.

Finally, Figure 5 shows the comparison of the performance of the proposed model with the baseline methods in estimating neural activities summed over all the 37 regions in the test data of Steinmetz dataset (including the 4 regions reported in Table 1).

Table 1: Comparison of the Negative Log-Likelihood (NLL) measure for the neural intensity function estimations in example regions of the synthetic and Steinmetz datasets.

Methods	Synthetic Dataset				Steinmetz Dataset			
	D1	D2	E1	E2	VISp	SUB	VISam	SNr
NN-Poisson	**-5285.118**	**-2703.695**	**-14708.950**	**-8570.335**	**3.451**	**-2.522**	**0.022**	**-19.217**
GLM	**-324.343**	**-146.556**	**-439.662**	**-347.569**	**87.114**	**17.219**	**2.216**	**-5.782**
NHPoisson	**-1330.065**	**-1066.044**	**-1354.954**	**-1341.548**	**33.817**	**-0.012**	**2.915**	**-11.871**
U-GP	**-3532.831**	**-2317.245**	**-10334.716**	**-5289.174**	**12.337**	**1.336**	**1.066**	**-17.484**
U-ANN	**-3417.905**	**-2010.752**	**-4981.384**	**-2863.996**	**12.679**	**1.354**	**1.108**	**-14.603**
P-GPL	**-2631.857**	**-2196.593**	**-4774.682**	**-3729.311**	**12.375**	**-1.312**	**1.584**	**-15.173**

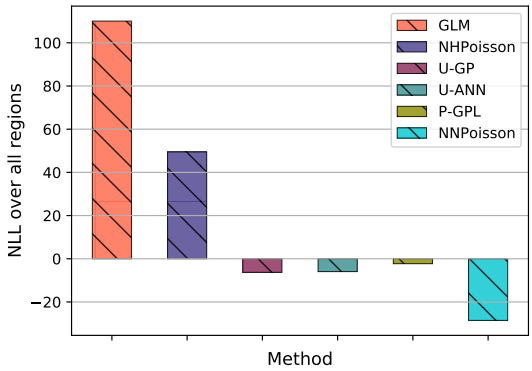


Figure 5: Total NLL of the estimated neural intensity function on Steinmetz test set. Illustrating the sum of NLL values evaluated over all the 37 test regions in the Steinmetz dataset using different baselines compared to the performance of the proposed method.

4 Discussion

We presented a novel framework for linking neural spike trains to sensory inputs and behaviour. The framework extended previous works on fMRI data [Dezfouli et al. 2018] by using a flexible Point process framework. The model was able to learn a suitable encoding of the stimulus and provided a joint explanation for both behavioural and neural data that could be used to recover correlational links between neural and behavioural activities. Unlike previous efforts, the learning process of the proposed model is independent of the selection of a time bin for spike count calculations obtaining higher robustness. The current method represents the dependency of neural activities on stimulus and trial duration, but not on previous neural activities – thus capturing signal rather than noise correlations (although the latter are an obvious target for future work). There are many additional directions for future work: capturing richer aspects of behaviour that are known to couple to neural activity [Balleine and O'Doherty 2010]; integrating and/or substituting spiking activity with calcium imaging; like auto-regressive linear-nonlinear-Poisson (LNP) models [Chichilnisky 2001]; differentiating more finely the activity in different regions (including neurons with opposite stimulus coding). Nevertheless, we suggest that our method casts brain and behaviour interactions in a compelling new light.

References

Abadi, Martín et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org.

Balleine, Bernard W and John P O'Doherty (2010). "Human and rodent homologies in action control: corticostriatal determinants of goal-directed and habitual action". In: *Neuropsychopharmacology* 35.1, pp. 48–69.

Barak, Omri (2017). "Recurrent neural networks as versatile tools of neuroscience research". In: *Current opinion in neurobiology* 46, pp. 1–6.

Britten, Kenneth H et al. (1996). "A relationship between behavioral choice and the visual responses of neurons in macaque MT". In: *Visual neuroscience* 13.1, pp. 87–100.

Cebrian, AC (2015). "NHPoisson: Modelling and Validation of Non Homogeneous Poisson Processes". In: *R package version* 3.

Chichilnisky, EJ (2001). "A simple white noise analysis of neuronal light responses". In: *Network: Computation in Neural Systems* 12.2, pp. 199–213.

Chilinski, Pawel and Ricardo Silva (2018). "Neural likelihoods via cumulative distribution functions". In: *arXiv preprint arXiv:1811.00974*.

Churchland, Mark M et al. (2010). "Stimulus onset quenches neural variability: a widespread cortical phenomenon". In: *Nature neuroscience* 13.3, pp. 369–378.

Cunningham, John P and M Byron Yu (2014). "Dimensionality reduction for large-scale neural recordings". In: *Nature neuroscience* 17.11, p. 1500.

Daley, D.J. and D. Vere-Jones (2006). *An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods*. Probability and Its Applications. Springer New York. ISBN: 9780387215648.

Dayan, Peter and Laurence F Abbott (2001). "Theoretical neuroscience: computational and mathematical modeling of neural systems". In.

Dezfouli, Amir et al. (2018). "Integrated accounts of behavioral and neuroimaging data using flexible recurrent neural network models". In: *Advances in Neural Information Processing Systems*, pp. 4228–4237.

Ganguli, Surya and Haim Sompolinsky (2012). "Compressed sensing, sparsity, and dimensionality in neuronal information processing and data analysis". In: *Annual review of neuroscience* 35, pp. 485–508.

Gold, Joshua I and Michael N Shadlen (2007). "The neural basis of decision making". In: *Annual review of neuroscience* 30.

Goris, Robbe LT, J Anthony Movshon, and Eero P Simoncelli (2014). "Partitioning neuronal variability". In: *Nature neuroscience* 17.6, pp. 858–865.

Grimmett, Geoffrey and David Stirzaker (2020). *Probability and random processes*. Oxford university press.

Jun, James J et al. (2017). "Fully integrated silicon probes for high-density recording of neural activity". In: *Nature* 551.7679, pp. 232–236.

Kass, Robert E, Uri T Eden, and Emery N Brown (2014). *Analysis of neural data*. Vol. 491. Springer.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Kobak, Dmitry et al. (2016). "Demixed principal component analysis of neural population data". In: *Elife* 5, e10989.

Kriegeskorte, Nikolaus and Rogier A Kievit (2013). "Representational geometry: integrating cognition, computation, and the brain". In: *Trends in cognitive sciences* 17.8, pp. 401–412.

Liu, David and Máté Lengyel (2021). "A universal probabilistic spike count model reveals ongoing modulation of neural variability". In: *Advances in Neural Information Processing Systems* 34.

Mante, Valerio et al. (2013). "Context-dependent computation by recurrent dynamics in prefrontal cortex". In: *nature* 503.7474, pp. 78–84.

Meyer, Arne F et al. (2017). "Models of neuronal stimulus-response functions: elaboration, estimation, and evaluation". In: *Frontiers in systems neuroscience* 10, p. 109.

Omi, Takahiro, Kazuyuki Aihara, et al. (2019). "Fully Neural Network based Model for General Temporal Point Processes". In: *Advances in Neural Information Processing Systems*, pp. 2120–2129.

Paninski, Liam (2004). "Maximum likelihood estimation of cascade point-process neural encoding models". In: *Network: Computation in Neural Systems* 15.4, pp. 243–262.

Paninski, Liam, Jonathan Pillow, and Jeremy Lewi (2007). "Statistical models for neural encoding, decoding, and optimal stimulus design". In: *Progress in brain research* 165, pp. 493–507.

Paninski, Liam et al. (2010). "A new look at state-space models for neural data". In: *Journal of computational neuroscience* 29.1-2, pp. 107–126.

Richards, Blake A et al. (2019). "A deep learning framework for neuroscience". In: *Nature neuroscience* 22.11, pp. 1761–1770.

Rieke, Fred et al. (1999). *Spikes: exploring the neural code*. MIT Press.

Ross, Sheldon M et al. (1996). *Stochastic processes*. Vol. 2. Wiley New York.

Schaeffer, Rylan et al. (2020). "Reverse-engineering recurrent neural network solutions to a hierarchical inference task for mice". In: *Advances in Neural Information Processing Systems* 33.
Shadlen, Michael N and William T Newsome (1998). "The variable discharge of cortical neurons: implications for connectivity, computation, and information coding". In: *Journal of neuroscience* 18.10, pp. 3870–3896.
Sill, Joseph (1998). "Monotonic networks". In: *Advances in neural information processing systems*, pp. 661–667.
Steinmetz, Nicholas A et al. (2019). "Distributed coding of choice, action and engagement across the mouse brain". In: *Nature* 576.7786, pp. 266–273.
Sussillo, David (2014). "Neural circuits as computational dynamical systems". In: *Current opinion in neurobiology* 25, pp. 156–163.
Truccolo, Wilson et al. (2005). "A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects". In: *Journal of neurophysiology* 93.2, pp. 1074–1089.
Whiteway, Matthew R and Daniel A Butts (2019). "The quest for interpretable models of neural population activity". In: *Current opinion in neurobiology* 58, pp. 86–93.
Wimmer, Klaus et al. (2015). "Sensory integration dynamics in a hierarchical network explains choice probabilities in cortical area MT". In: *Nature communications* 6.1, pp. 1–13.
Wu, Anqi et al. (2017). "Gaussian process based nonlinear latent structure discovery in multivariate spike train data". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 3499–3508.
Yamins, Daniel LK et al. (2014). "Performance-optimized hierarchical models predict neural responses in higher visual cortex". In: *Proceedings of the national academy of sciences* 111.23, pp. 8619–8624.
Yu, M Byron et al. (2006). "Extracting dynamical structure embedded in neural activity". In: *Advances in neural information processing systems*, pp. 1545–1552.

Neural Network Poisson Models for Behavioural and Neural Spike Train Data

Supplementary Materials

Moein Khajehnejad*, Forough Habibollahi*, Richard Nock, Ehsan Arabzadeh, Peter Dayan, Amir Dezfouli

A Proof of Theorem 1.

Following the Mapping Theorem of an inhomogeneous Poisson process [Grimmett and Stirzaker 2020], and given a one-to-one monotonic transformation function, $z_n(t') = t$, between the original inhomogeneous Poisson point process $0 < s'^1 < s'^2 <, \ldots, < s'^j \leq W_n \leq W$ and $0 < s^1 < s^2 <, \ldots, < s^j \leq W$, let $0 < s^1 < s^2 <, \ldots, < s^j \leq W$ represent a set of event (spike) times from a second inhomogeneous Poisson point process. Let $g(t)$ represent the corresponding event time probability density. Given $g(t)$ is a density function (measurable and non-negative function), following the push-forward probability density, we get:

$$g(t) = f((z^{-1})(t)) \cdot \partial_t \, (z^{-1}), \tag{13}$$

where z^{-1} is the inverse of the transformation, $z_n(.)$, and $f(t')$ is the event time probability density function of the original Poisson point process.

Now, for $t \in (0, W]$, let $N(t)$ be the sample path of the associated counting process. The sample path is a right continuous function that jumps 1 at the event times and is constant otherwise. Then, we compute the probability that a spike s^k occurs in $[t, t+\Delta t)$ where $k = N(t)+1$. Note that events $\{N(t+\Delta t) - N(t) = 1\}$ and $\{s^k < t+\Delta t \mid s^k > t\}$ are equivalent. Thereby,

$$P(N(t + \Delta t) - N(t) = 1) = P(s^k < t + \Delta t \mid s^k > t). \tag{14}$$

From the definition of conditional probability:

$$P(s^k < t + \Delta t \mid s^k > t) = \frac{P(t < s^{\,k} < t + \Delta t)}{P(s^k > t)}. \tag{15}$$

Therefore, we get:

$$P(N(t + \Delta t) - N(t) = 1) = \frac{P(t < s^{\,k} < t + \Delta t)}{P(s^k > t)} = \frac{\int_t^{t+\Delta t} g(u)du}{1 - \int_{s_{N(t)}}^{t} g(u)du}. \tag{16}$$

Using equation 13, we obtain:

$$P(N(t + \Delta t) - N(t) = 1) = \frac{\displaystyle\int_t^{t+\Delta t} \partial_u \, (z_n^{-1}) \cdot f(z_n^{-1}(u))du}{1 - \displaystyle\int_{s_{N(t)}}^{t} \partial_u \, (z_n^{-1}) \cdot f(z_n^{-1}(u))du}. \tag{17}$$

We also have $u' = z_n^{-1}(u)$ and hence:

$$\frac{du}{du'} = \frac{du}{dz_n^{-1}(u)} = \frac{1}{\frac{dz_n^{-1}(u)}{du}} = \frac{1}{\partial_u \, (z_n^{-1})}. \tag{18}$$

Thereby, inserting the above in equation 17 and by a change of variable u to u', noting that $z_n^{-1}(t + \Delta t) := (t + \Delta t)'$, we get:

$$P(N(t + \Delta t) - N(t) = 1) = \frac{\displaystyle\int_{t'}^{(t+\Delta t)'} \partial_u \, (z_n^{-1}) \cdot f(u') \cdot \frac{1}{\partial_u \, (z_n^{-1})} du'}{1 - \displaystyle\int_{s'_{N(t')}}^{t'} \partial_u \, (z_n^{-1}) \cdot f(u') \cdot \frac{1}{\partial_u \, (z_n^{-1})} du'} = P(N(t + \Delta t)' - N(t') = 1), \tag{19}$$

* Authors contributed equally to this manuscript.

where $N(t')$ is the sample path of the associated counting process for $t' \in (0, W_n]$.
The intensity function of a point process, $\lambda(t)$, can be written as:

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{P(N(t+\Delta t) - N(t) = 1)}{\Delta t}. \tag{20}$$

hence, since we proved $P(N(t+\Delta t) - N(t) = 1) = P(N(t+\Delta t)' - N(t') = 1)$, we get:

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{P(N(t+\Delta t) - N(t) = 1)}{\Delta t} = \lim_{\Delta t \to 0} \frac{P(N(t+\Delta t)' - N(t') = 1)}{\Delta t} = \lambda_n(t'). \tag{21}$$

Finally, the intensity function of the new count process $\lambda(t)$ equals $\lambda_n(t')$ of the original Poisson point process, thereby satisfying the four properties of a Poisson point process [Ross et al. 1996] and we have now established our result. □
Figure 6 is schematically illustrating this result in case of a linear transformation function.

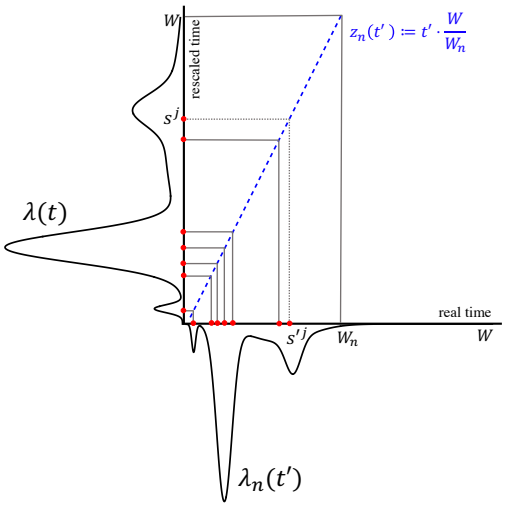


Figure 6: Illustrating the transformation of the Poisson point process in real time $\lambda_n(t')$, to the rescaled time domain which results a point process with intensity function $\lambda(t)$. The transformation function $z_n(t') = t' \cdot \frac{W}{W_n}$ performs a one-to-one monotonic mapping of the events in real time (s'^j) to the events on the rescaled time axis (s^j).

B Proof of Proposition 1.

The cumulative intensity function of a point process with intensity function $\lambda_n(t')$ is given by:

$$\Lambda_n(t') = \int_0^{t'} \lambda_n(\tau') d\tau', \tag{22}$$

for $t' \in [0, W_n]$, which can be (automatically) differentiated to produce the intensity function,

$$\lambda_n(t') = \frac{\partial \Lambda_n(t')}{\partial t'}. \tag{23}$$

Given a monotonic transformation function $z_n(t') = t$, and since $\lambda_n(t') = \lambda(t)$ (see A), we get:

$$\begin{aligned} \Lambda(t) &= \int_0^t \lambda(\tau) d\tau = \int_0^t \lambda_n(\tau') d\tau \\ &= \int_0^{z_n^{-1}(t)} \lambda_n(\tau') \cdot \frac{1}{\partial_\tau (z_n^{-1})} d\tau'. \end{aligned} \tag{24}$$

Now, following integration by parts,

$$\Lambda(t) = \Lambda_n(t') \cdot \frac{1}{\partial_t (z_n^{-1})} - \int_0^{t'} \Lambda_n(\tau') \cdot \frac{d}{d\tau'} \left(\frac{1}{\partial_\tau (z_n^{-1})} \right) d\tau'. \tag{25}$$

Inserting the special case of a linear transformation function where $\partial_\tau(z_n^{-1})$ is a constant, the above equation reduces to:

$$\Lambda(t) = \frac{1}{\partial_t(z_n^{-1})} \cdot \Lambda_n(t'). \tag{26}$$

In this work, we rescaled time using function $z_n(t') = t'W/W_n$ where $z_n : [0, W_n] \to [0, W]$ and we chose to stretch the firing rates. Thereby, trial specific cumulative intensity functions are simply defined as:

$$\Lambda(t) = \frac{W}{W_n} \cdot \Lambda_n(t'). \tag{27}$$

C Model architecture and training

In this section we explain the details of the model architecture.

C.1 Steinmetz dataset

The input stimulus $\mathbf{x}_n$ is passed through two dense layers with 20 units (SOFTPLUS activation). The output is then passed through a layer with $|\mathbf{h}_n| = 50$ units using a linear activation function to create the stimulus embedding, $\mathbf{h}_n$. There is no constraint on the weights in these layers.

As mentioned in the main text, we assumed the time rescaling function is linear. The elapsed time $t' \in [0, W_n]$ (since the stimulus) is first scaled by a factor of W/W_n to obtain $t \in [0, W]$ and is then passed through a linear layer (with the same number of units as $|\mathbf{h}_n| = 50$) with the weights constrained to be non-negative. The resulting output is added to $\mathbf{h}_n$, with the sum then being passed through a TANH activation function.
The output of this TANH activation is passed through a dense layer with 20 units and a further TANH activation function (first neural layer) and then through another dense layer with 42 units and SOFTPLUS activation function (second neural layer). The weights of these layers are all constrained to be non-negative. The outputs of this layer correspond to $\Lambda_u^{\text{N}}(t; \mathbf{h}_n)$, referred to as neural outputs, and are multiplied by W_n/W in the network readout to obtain $\Lambda_{u,n}^{\text{N}}(t'; \mathbf{h}_n)$ for $u = 1 \ldots 42$ which is then passed to the loss function.

For modelling behavioural data, the neural output ($\Lambda_u^{\text{N}}(t; \mathbf{h}_n)$) is passed through a dense layer with 10 units (TANH activation) and then through another dense layer with 1 unit (SOFTPLUS activation) to produce $\Lambda_{\text{RIGHT}}^{\text{B}}(t; \mathbf{h}_n)$. The weights are all constrained to be non-negative.

The neural output ($\Lambda_u^{\text{N}}(t; \mathbf{h}_n)$) is also passed through a dense layer with 10 units (TANH activation) and then through another dense layer with 1 unit (SOFTPLUS activation) to produce $\Lambda_{\text{LEFT}}^{\text{B}}(t; \mathbf{h}_n)$. The weights of all the layers are constrained to be non-negative.

Note that the path from t to neural and behavioural outputs only contains positive weights which, together with monotonic activation functions, is a sufficient condition to guarantee that the outputs (neural and behavioural) are monotonic functions of t [Sill 1998; Chilinski and Silva 2018; Omi, Aihara, et al. 2019].

$\Lambda_{\text{LEFT}}^{\text{B}}(t; \mathbf{h}_n)$ and $\Lambda_{\text{RIGHT}}^{\text{B}}(t; \mathbf{h}_n)$ were multiplied by factor of W_n/W to obtain trial specific values for $\Lambda_{\text{LEFT},n}^{\text{B}}(t'; \mathbf{h}_n)$ and $\Lambda_{\text{RIGHT},n}^{\text{B}}(t'; \mathbf{h}_n)$.

Figure 7 visualizes the details of the model structure implemented to be trained on the Steinmetz dataset.

C.2 Synthetic dataset

The same model architecture was used for this dataset as explained above, but with the following differences: (i) the size of output neural layer was set to four for the four regions involved in this dataset; (ii) the size of the embedding was set to $|\mathbf{h}_n| = 20$, which was smaller than for the Steinmetz dataset, since the synthetic dataset had a smaller number of regions.

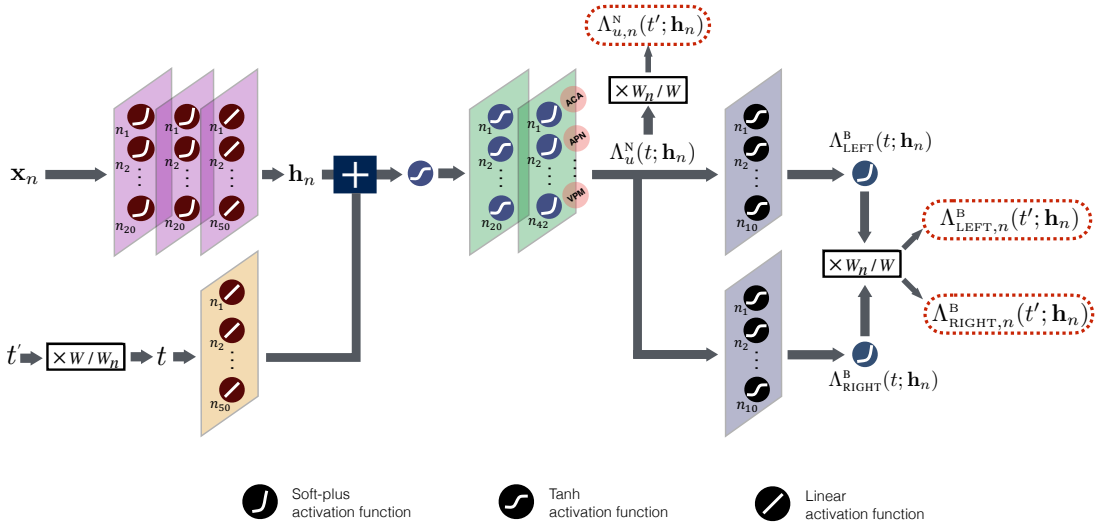


Figure 7: Model Architecture. Schematic representing the details of the implemented neural network to be trained on the Steinmetz dataset. The neural and behavioural outputs of this network are multiplied by a factor of W_n/W for the selected linear transformation function, z, to obtain the final model readouts.

C.3 Training

Parameter adaptation was performed using the Adam optimizer with learning rates of 0.01 (Steinmetz) and 0.001 (synthetic). The neural loss function was used to train all the weights up to the neural outputs, and behavioural loss was used to train all the weights connecting neural outputs to the behavioural outputs. For the Steinmetz dataset, in each iteration of training the weights were updated using neural loss for 10 steps, and then the behavioural loss was used to update the weights for 420 steps (each step is one update to the weights). The training process continued until no significant improvement on the losses (on training data) was observed. For the synthetic dataset, in each iteration the neural loss was used to update the weights for 10 steps, which was followed by training using the behavioural loss for 40 steps.

For the generation of the synthetic data we used the code provided in this link[3]. We generated 1200 trials for training and 600 trials for testing. For the Steinmetz dataset, we used 12 sessions for testing and the rest for training. Note that since in each session only a subset of regions were recorded (42 regions in total), we had 42 regions for training the model and 37 regions for testing the model. The split of sessions between training and test was chosen so as to obtain the maximum number of regions in both training and test datasets. The data was downloaded from this link[4].

Automatic differentiation in Tensorflow was used to calculate the intensities.

D Effects of spike time rescaling on model performance

To illustrate the effectiveness of our proposed model and the increase in model performance compared to the case where the time rescaling block is disabled, we reproduced both networks and evaluated them on the synthetic dataset. The two networks were identical in all other aspects. The NLL measures computed on the test data for both networks in all 4 regions as well as the scores of the NO SCALING version of the model relative to the original model are listed in Table 2; the positive score means that the proposed model (with time rescaling) is better than the case without rescaling.

[3]https://senselab.med.yale.edu/MicrocircuitDB/showModel.cshtml?model=168867&file=%2Fhierarchical_network%2Freadme.html#tabs-2

[4]https://github.com/nsteinme/steinmetz-et-al-2019/wiki/data-files

Table 2: Comparison of the Negative Log-Likelihood (NLL) measure for the neural intensity function estimations in the original model with and without time rescaling.

Network Model	Synthetic Dataset				
	D1	D2	E1	E2	Total
NN-Poisson (time rescaling)	**-5285.118**	**-2703.695**	**-14708.95**	**-8570.335**	**-31268.098**
NN-Poisson (no time rescaling)	**-5038.888**	**-2477.002**	**-14574.856**	**-8394.009**	**-30484.717**
Relative score	**0.04**	**0.08**	**0.009**	**0.02**	**0.025**

E Baseline Methods

For the comparison of our method with the available baselines, we use Generalized-linear models (GLM) [Truccolo et al. 2005; Paninski 2004] as well as a recent method for modelling non homogeneous Poisson processes (NHPoisson) [Cebrian 2015].

E.1 GLM

For comparison with a Poisson family GLM, one Poisson GLM with a continuous predictor (elapsed time), a categorical predictor with three/four levels for synthetic/Steinmetz dataset (stimuli coherence/contrast), as well as a behavioral predictor (the reaction times) is used to estimate the neural intensity function of each region. As the continuous predictor, we include a list of sinusoidal, tanh, and exponential functions of the elapsed time as covariates. The observed behavioral covariates are in the form of temporal arrays consisting of NO-ACTION:0, RIGHT reaction time, or -LEFT reaction time in each time bin. Note that the sign of the provided reaction time is indicative of the direction of the motion. The total spike counts for all trials were then calculated and concatenated for count windows of 5ms. Due to the large data size and the use of aggregated spike counts from all trials, we did not use self-coupling terms for spike events. The spike counts were then used as targets for the GLM model, leading to a specific neural intensity function. The derived intensity function was then used to calculate the negative log-likelihood (NLL) based on equation 1.

E.2 NHPoisson

We used the same procedure as above with the count windows equal to 0.5 ms for the Steinmetz and synthetic datasets to extract the regional activity rates exploiting the elapsed time, trial stimulus type, and the behavioral reaction times. The NHPoisson R package [Cebrian 2015] requires binary information about whether a given event has occurred in each time bin. Thereby, we used a smaller count window with an order of magnitude similar to inter spike intervals. This enabled us to keep the average spike events per non empty time bin equal to 0.97 and 0.99 for Steinmetz and synthetic datasets respectively. A binary vector then stored the presence or absence of spikes in each time bin and then non-zero indices were fed to the NHPoisson model. Repeatedly, in addition to elapsed time arrays and the categorical array of stimuli coherence/contrast levels for synthetic/Steinmetz dataset, temporal arrays consisting of NO-ACTION:0, RIGHT reaction time, or -LEFT reaction time in each time bin were also provided as behavioural covariates. We also examined a list of sinusoidal, tanh, and exponential functions of the elapsed time as covariates. Using the embedded Akaike information criterion (AIC) calculator in the package, the best covariate was selected and added to the model (lowest AIC score). The extracted neural intensity functions for each region were then obtained. The derived intensity functions were plugged into equation 1 to obtain NLL values for the NHPoisson model.

E.3 Universal Count Model

This universal probabilistic spike count model uses sparse Gaussian processes to derive spike count distributions. It consists of C Gaussian process (GP) priors, a basis expansion, and a linear-softmax mapping [Liu and Lengyel 2021]. Using 1 ms time bins, the spike counts of all trials associated with each neuron (in the Steinmetz dataset) or the neuronal population (in the synthetic dataset) are calculated and concatenated together. Note that for the case of the synthetic dataset, since the generated spike time sequences are not associated with specific neuron IDs and represent the population activity, a single cumulative spike count sequence was fed to the model for training. Once again, we use the elapsed time, the categorical stimulus type associated with each trial repeated for the duration of the trial, as well as the behavioural temporal array (consisting of 0 (i.e. NoGo), RIGHT reaction time, or -LEFT reaction time in each time bin) as the observed covariates for the model. To implement the original model (U-GP), we set hyperparameter

$C = 3$ as suggested in [Liu and Lengyel 2021] and choose an elementwise linear-exponential basis expansion. We use 5 fold-cross validation on the data from each region and we cross-validate over the neuron dimension by using the train set to infer the latent states in the test data, and then evaluate the cross-validated log-likelihood of the fitted model. For the case of synthetic dataset, the cross validation is across the trials. The learning rate is set to 0.001 and we choose a tuple batch size (to indicate the trial structure of the data; for details please refer to [Liu and Lengyel 2021]) equal to the number of time bins accumulated over all trials associated with each region. In this method, an inference model with log likelihood based objectives using variational inference is built upon the data and spike couplings are computed and added as well.
In the second variation of the model (U-ANN), the same setting is implemented by replacing the GP mapping with an artificial neural network (ANN) mapping. In this model, a 10-fold cross validation is utilized. The remainder of the parameters and model settings are tuned according to the suggestions in [Liu and Lengyel 2021].

E.4 Poisson Gaussian-Process Latent Variable (P-GPL)

This model uses Poisson spiking observations and two underlying Gaussian processes [Wu et al. 2017]. A fast approximate inference method called the decoupled Laplace approximation is applied to learn the model from data. A Gaussian process is first used to extract the nonlinear evolution of the latent dynamic in the form of a latent variable. A second GP then generates the log of the tuning curve as a nonlinear function of the latent variable. This curve is then mapped to a final tuning curve via an exponential link function to estimate the spike rates of each neuron ($\lambda_i(t)$ for the ith neuron) and henceforth obtain the population activity rate in each discrete time bin. Here, the spike count matrix consisting of spike counts in time bins of 1 ms for each neuron (or the whole population in the synthetic dataset) is used to construct a generative model of the latent structure underlying these data. The data from all trials for each neuron are once again concatenated. Both Sinusoid and deterministic Gaussian bump tuning curves are examined to estimate the latent processes. The deterministic Gaussian bump tuning curve was then selected to report the results due to higher performance on our data which was expected given the naturally 2D motion space which is present in the Steinmetz dataset. We first estimated the parameters for the mapping function using spike trains from all the neurons within the training dataset. Then these parameters were fixed and the latent process using spike trains from 70% of the test data were inferred (as suggested in [Wu et al. 2017]). We then report the NLL measure comparing the estimated latent process generating $\lambda_i(t)$ values and the known empirical rates from the remaining test data averaged over all neurons in each region. The rest of the parameters are chosen according to [Wu et al. 2017].

F Calculation of empirical rates

For the empirical firing rates, we divided the observation period into equally-sized bins each having 0.008s and 0.05s width for the Steintmetz and synthetic datasets respectively. Then, we calculated the total number of spikes in each period and normalized that by the total number of neurons and trials contributed to the spike set and also by the period duration (0.008s or 0.05s). Note that only spikes which were made *before* the response were included in the analysis. Moreover, given the selected stimulus type for each examined W_n in Figure 4, only trials which had end times in an interval of 37.5 ms before the W_n were included. This interval was chosen to be 60 ms for the results in Figure 3. This interval's width was adjusted so that for each region in both datasets, at least 15% of all trials with the chosen stimulus type would fall within the interval before each of the 5 selected W_ns in Figures 8 and 10.

G Additional results

G.1 Synthetic dataset

Using the procedure explained in Section F, Figure 8 shows the comparison of the model estimation to the empirically derived firing rates for all the regions in the synthetic dataset.

To evaluate the performance of the model in terms of predicting the behaviour of the neural population and the reaction times, we used all trials with rightward motion and fed them to the trained network of the model to get the estimated neural and behavioural response rates. The average of the estimated behavioural response rates corresponding to each of a set of potential intervals of the estimated neural responses were then calculated over trials. Figure 9 shows the results

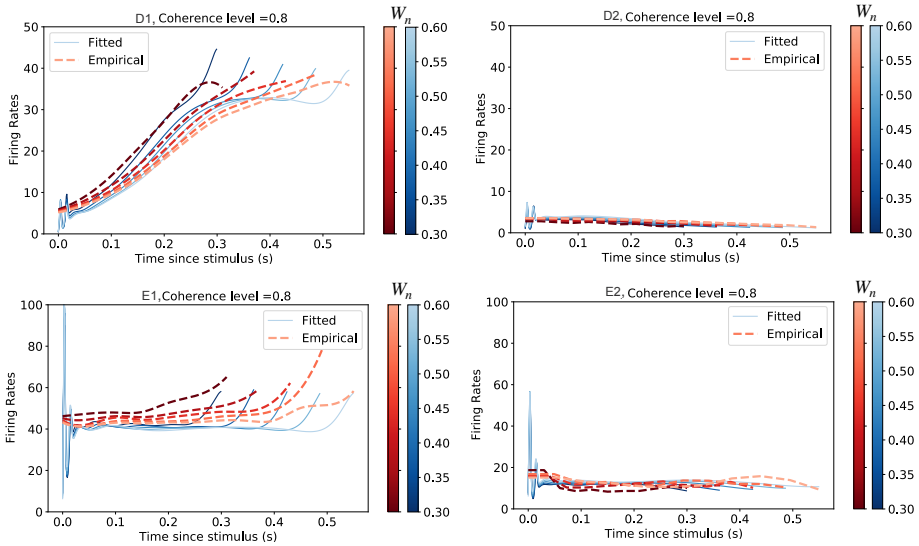


Figure 8: Synthetic Dataset. The red dashed lines are the empirically derived firing rates compared to the blue solid lines which are estimated using the proposed model. The plots correspond to the trials with high contrast on RIGHT. As expected from the dataset architecture, for trials when the RIGHT choice is made, the activity in D1 represents an increasing trend until the response is made. This is in contrast with region D2 for these trials where there is little to no activity detected in this region. Neurons in both sensory regions, E1 and E2 show constant activity throughout each trial with higher activity detected by E1 neurons which prefer rightward motion.

for all 4 regions. The purple bars show the average behavioural response rate in each occurred neural activity interval. The orange bars indicate the proportion of trials which achieve each specific neural activity range among all RIGHT trials, hence its occurrence frequency. Comparing the sensory regions in this figure, the behavioural correlate is with a low firing rate for E2 and with significantly higher firing rates in E1 which is the rightward selective sensory region.

G.2 Steinmetz dataset

With a similar procedure as explained above, Figure 10 shows the comparison of the model estimation with the empirically derived firing rates for all the regions in the Steinmetz dataset. The occasional observed underfitting for some regions may be overcome by further forms of regularization in future works. The plots highlight the performance quality of the trained network on test regions. The temporal neural pattern in each region is well captured by the model outputs for different observation windows W_n.

The behavioural performances were also evaluated similar to above for the available test regions in the Steinmetz dataset. Figure 11 illustrates the results. In each region, the highest response rates correspond to the neural activity values achieved near the end of trial in Figure 10. The purple bars show the average behavioural response rate in each neural activity interval. The orange bars show how frequently these were observed in each RIGHT trial; meaning the occurrence ratio of that specific neural response range over all RIGHT test trials.

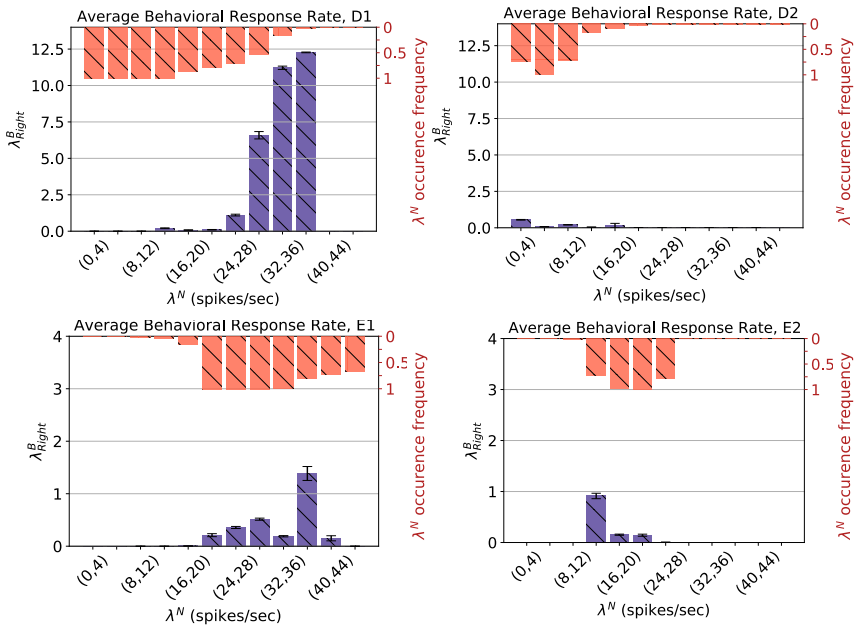
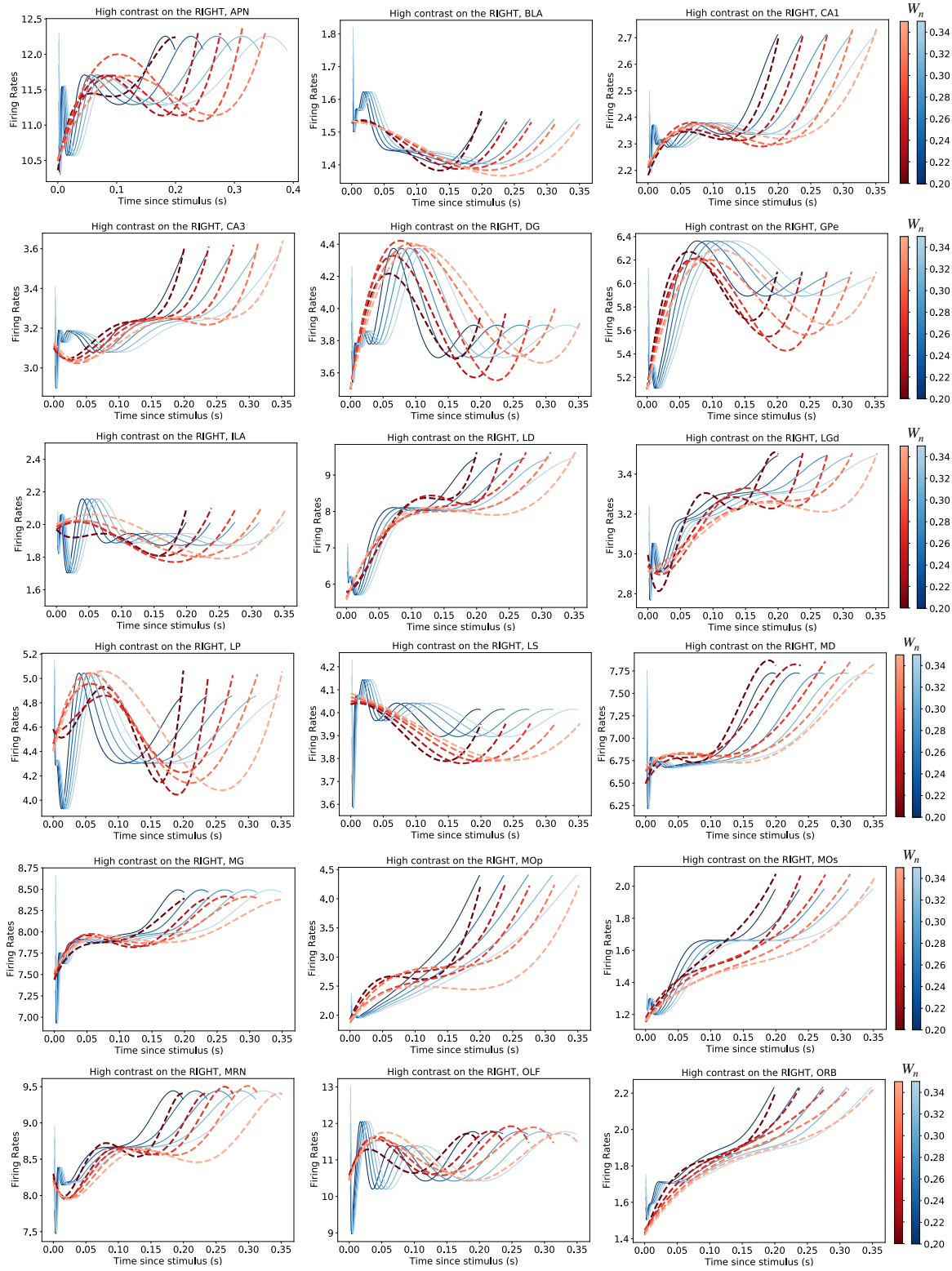


Figure 9: Synthetic Dataset Illustrating the average estimated response rate for each interval of estimated neural activity. Error bars show the standard error. Purple bars show the average behavioural response rates. Orange bars indicate the proportion of trials with the occurrence of each neural response interval. As expected in the RIGHT trials and represented by purple bars, highest behaviour rates are detected in D1 when neural activities approach 40 spikes/s in the (32,36) and (36,40) range. On the other hand, there is little to no activity estimated for D2 which is a leftward preferring integration region. Highest response rates are achieved for when E1 neural activity is in the vicinity of 40 spikes/s (in the (36,40) range) and when the neural activity in E2 is in (12,14) range. These are in agreement with results from Figure 8.


High contrast on the RIGHT, APN
High contrast on the RIGHT, BLA
High contrast on the RIGHT, CA1
High contrast on the RIGHT, CA3
High contrast on the RIGHT, DG
High contrast on the RIGHT, GPe
High contrast on the RIGHT, ILA
High contrast on the RIGHT, LD
High contrast on the RIGHT, LGd
High contrast on the RIGHT, LP
High contrast on the RIGHT, LS
High contrast on the RIGHT, MD
High contrast on the RIGHT, MG
High contrast on the RIGHT, MOp
High contrast on the RIGHT, MOs
High contrast on the RIGHT, MRN
High contrast on the RIGHT, OLF
High contrast on the RIGHT, ORB
Firing Rates
Time since stimulus (s)
W_n

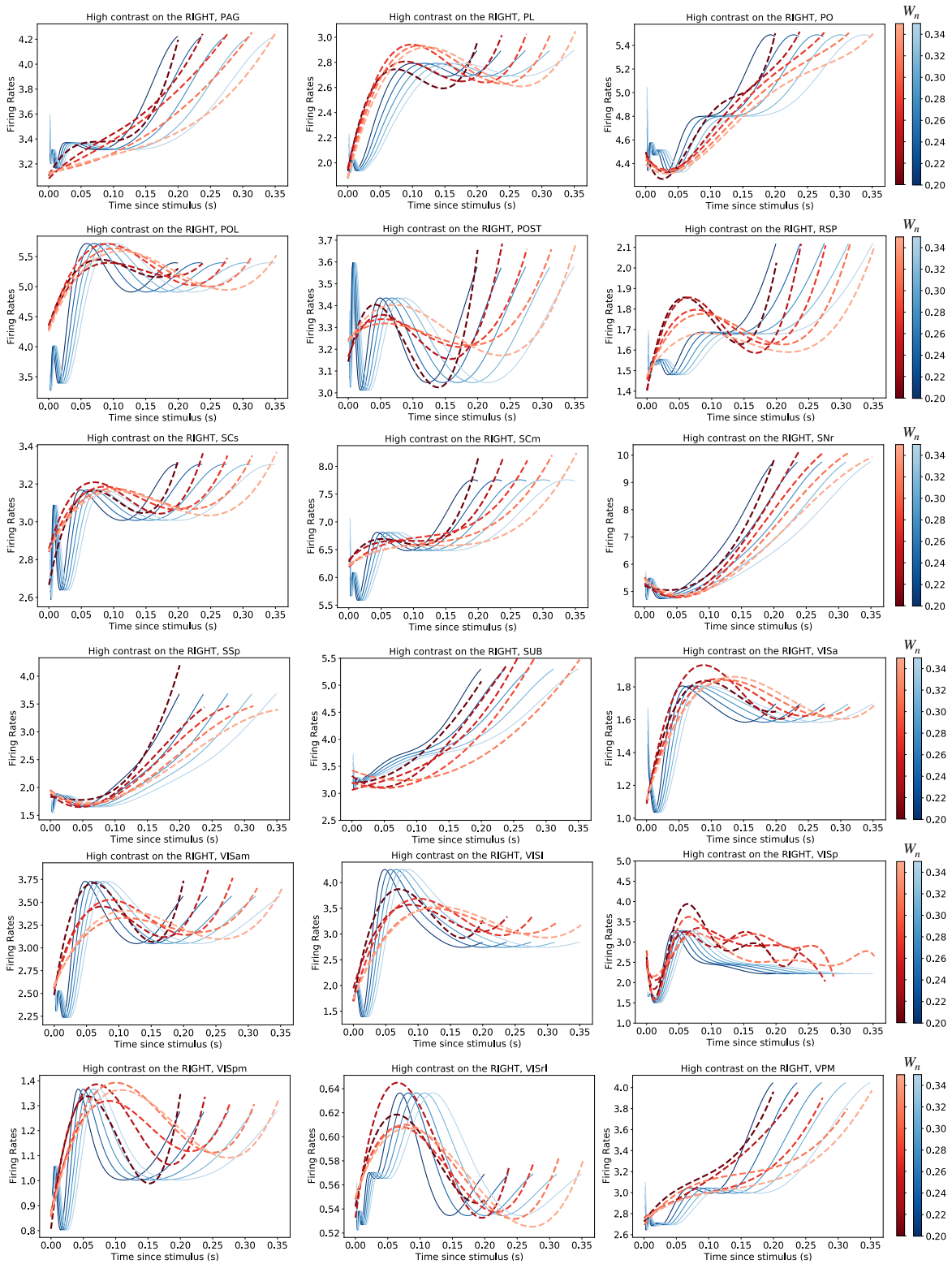
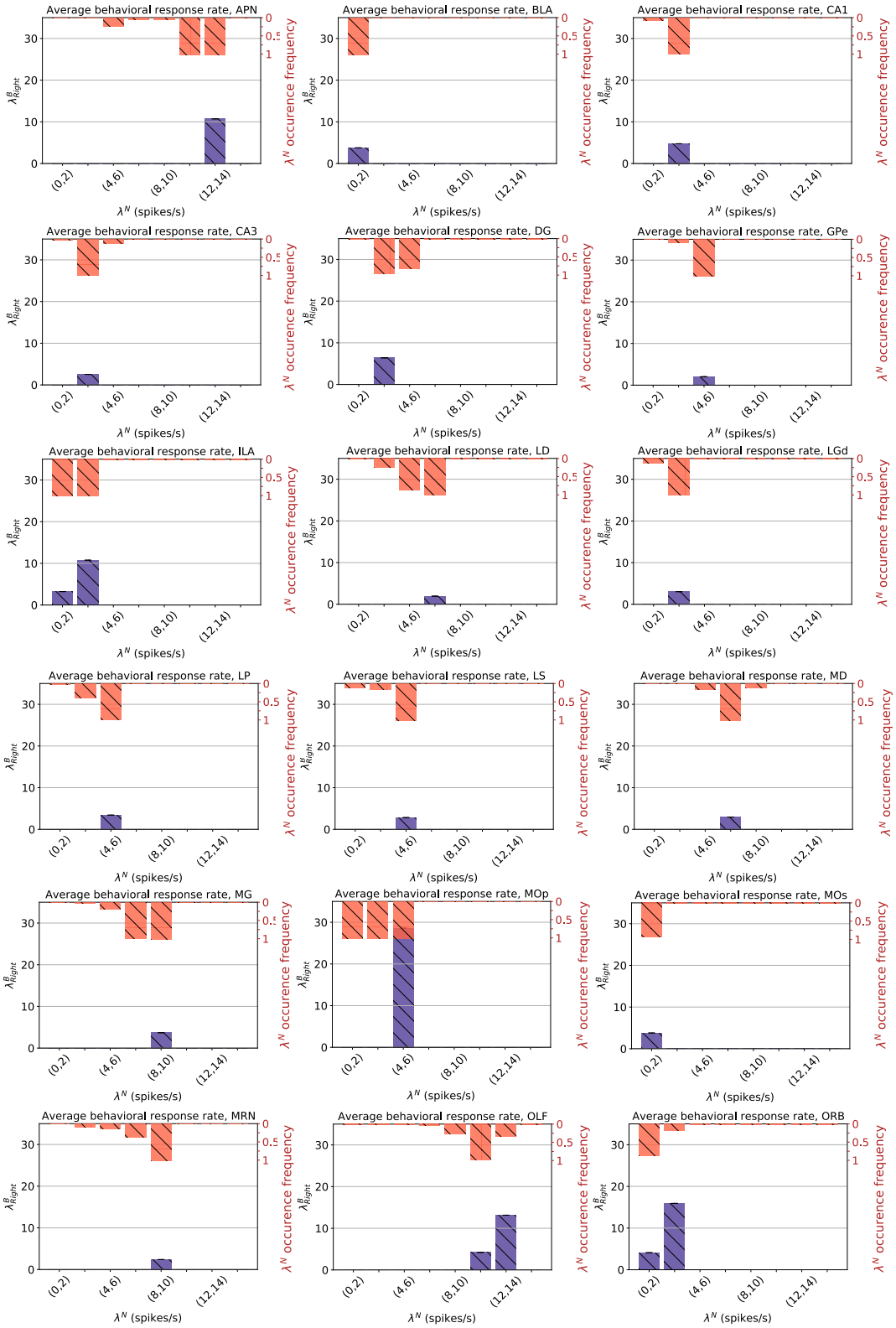


Figure 10: Steinmetz Dataset. The red dashed lines are the empirically derived firing rates compared to the blue solid lines which are estimated using the proposed model. The plots correspond to the trials with high contrast on RIGHT. The results are obtained only using trials with the RIGHT side contrast higher than LEFT.


Average behavioral response rate, APN
Average behavioral response rate, BLA
Average behavioral response rate, CA1
Average behavioral response rate, CA3
Average behavioral response rate, DG
Average behavioral response rate, GPe
Average behavioral response rate, ILA
Average behavioral response rate, LD
Average behavioral response rate, LGd
Average behavioral response rate, LP
Average behavioral response rate, LS
Average behavioral response rate, MD
Average behavioral response rate, MG
Average behavioral response rate, MOp
Average behavioral response rate, MOs
Average behavioral response rate, MRN
Average behavioral response rate, OLF
Average behavioral response rate, ORB
λ^B_{Right}
λ^N occurence frequency
λ^N (spikes/s)
(0,2)
(4,6)
(8,10)
(12,14)

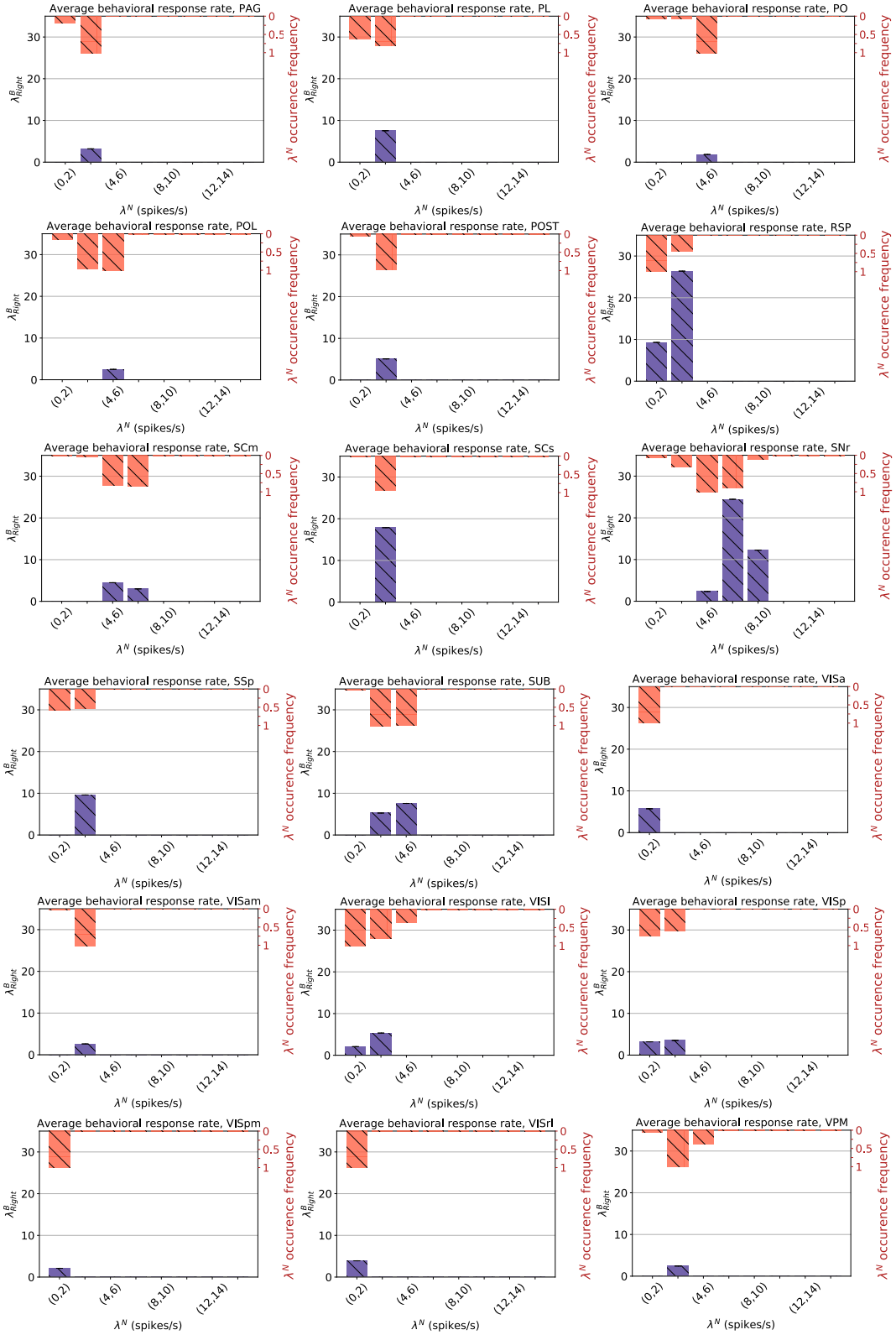


Figure 11: Steinmetz Dataset. Illustrating the average estimated response rate for each interval of estimated neural activity. The purple bars show the average behavioural response rate and orange bars represent the occurrence ratio of that specific neural response interval among all RIGHT test trials. Error bars show the standard error. The size of error bars is invisible compared to the bar sizes in most cases.