Variational online learning of neural dynamics

Yuan Zhao
Department of Neurobiology and Behavior
Stony Brook University
Stony Brook, NY 11794
yuan.zhao@stonybrook.edu

Il Memming Park
Department of Neurobiology and Behavior
Department of Applied Mathematics and Statistics
Institute for Advanced Computational Science
Stony Brook University
Stony Brook, NY 11794
memming.park@stonybrook.edu

Abstract

New technologies for recording the activity of large neural populations during complex behavior provide exciting opportunities for investigating the neural computations that underlie perception, cognition, and decision-making. Nonlinear state space models provide an interpretable signal processing framework by combining an intuitive dynamical system with a probabilistic observation model, which can provide insights into neural dynamics, neural computation, and development of neural prosthetics and treatment through feedback control. It brings the challenge of learning both latent neural state and the underlying dynamical system because neither is known for neural systems *a priori*. We developed a flexible online learning framework for latent nonlinear state dynamics and filtered latent states. Using the stochastic gradient variational Bayes approach, our method jointly optimizes the parameters of the nonlinear dynamical system, the observation model, and the black-box recognition model. Unlike previous approaches, our framework can incorporate non-trivial distributions of observation noise and has constant time and space complexity. These features make our approach amenable to real-time applications and the potential to automate analysis and experimental design in ways that testably track and modify behavior using stimuli designed to influence learning.

1 Introduction

Discovering interpretable structure from a streaming high-dimensional time series has many applications in science and engineering. Since the invention of the celebrated Kalman filter, state space models have been successful in providing a succinct, hence more interpretable, description of the underlying dynamics that explains the observed time series as trajectories in a low-dimensional state space. Taking a step further, state space models equipped with nonlinear dynamics provide an opportunity to describe the latent "laws" of the system that is generating the seemingly entangled time series [1, 2, 3]. Specifically, we are concerned with the problem of identifying a continuous nonlinear dynamics in the state space $\mathbf{x}(t) \in \mathbb{R}^d$ that captures the spatiotemporal structure of a noisy observation $\mathbf{y}(t)$:

$$\dot{\mathbf{x}} = F_\theta(\mathbf{x}(t), \mathbf{u}(t)) \qquad \text{(state dynamics)} \tag{1a}$$

Preprint. Work in progress.

$$\mathbf{y}(t) \sim P(\mathbf{y}(t) \mid G_\theta(\mathbf{x}(t), \mathbf{u}(t))) \qquad \text{(observation model)} \tag{1b}$$

where F and G are continuous functions that may depend on parameter θ, $\mathbf{u}(t)$ is control input, and P denotes a probability distribution that captures the noise in the observation; e.g., Gaussian distribution for field potentials or Poisson distribution for spike counts.

In practice, the continuous-time state dynamics is more conveniently formulated in discrete time as,

$$\mathbf{x}_{t+1} = f_\theta(\mathbf{x}_t, \mathbf{u}_t) + \boldsymbol{\epsilon}_t \qquad \text{(discrete time state dynamics)} \tag{2}$$

where $\boldsymbol{\epsilon}_t$ is intended to capture the unobserved (latent) perturbations of the state $\mathbf{x}_t$. Such (spatially) continuous state space models are natural in many applications where the changes are slow and the underlying system follows physical laws and constraints (e.g., object tracking), or where learning the laws are of great interest (e.g. in neuroscience and robotics) [4, 5, 6, 7, 8]. Specifically, in the context of neuroscience, the state vector $\mathbf{x}_t$ represents the instantaneous state of the neural population, while f captures the time evolution of the population state. Further interpretation of f can provide understanding as to how neural computation is implemented [9, 5, 10].

If the nonlinear state space model is fully specified, Bayesian inference methods can be employed to estimate the current state [11, 12]. Conventionally, the estimation of latent states using only the past observation is referred to as filtering – inference of the filtering distribution, $p(\mathbf{x}_t \mid \mathbf{y}_{\leq t})$. If both past and future observations are used, then the quantity of interest is usually the smoothing distribution, $p(\mathbf{x}_{\leq t} \mid \mathbf{y}_{\leq t})$). We are also interested in predicting the distribution over future states, $p(\mathbf{x}_{t:t+s} \mid \mathbf{y}_{\leq t})$ and observations, $p(\mathbf{y}_{t+1:t+s} \mid \mathbf{y}_{\leq t})$ for $s > 0$. However, in many applications, the challenge is in learning the parameters θ of the state space model (a.k.a. the system identification problem). We aim to provide a method for simultaneously learning both the latent trajectory $\mathbf{x}_t$ and the latent (nonlinear) dynamical and observational system θ, known as the *joint estimation problem* [13].

Expectation maximization (EM) based methods have been widely used in practice [14, 15, 16, 17], and more recently variational autoencoder methods [18, 19, 20, 21, 22, 23] have been proposed, all of which are designed for offline analysis, and not appropriate for real-time applications. Recursive stochastic variational inference has been successful in streaming data assuming independent samples [24], however, in the presence of temporal dependence, proposed variational algorithms (e.g. [7]) remain theoretical and lack testing.

In this study, we are interested in *real-time* signal processing and state space control setting [17] where online algorithms are needed that can recursively solve the joint estimation problem on streaming observations. A popular solution to this problem exploits the fact that online state estimators for nonlinear state space models such as extended Kalman filter (EKF) or unscented Kalman filter (UKF) can be used for nonlinear regression formulated as a state space model. By augmenting the state space with the parameters, one can build an online *dual* estimator using nonlinear Kalman filtering [25, 26]. However, they involve coarse approximation of Bayesian filtering, involve many hyperparameters, do not take advantage of modern stochastic gradient optimization, and are not easily applicable to arbitrary observation likelihoods. There are also closely related online version of EM-type algorithms [4] that share similar concerns.

In hopes of lifting these concerns, we derive an *online* black-box variational inference framework, referred to as **variational joint filtering (VJF)**, applicable to a wide range of nonlinear state dynamics (dynamic models) and observation models, that is, the computational demand of the algorithm is constant per time step. Our approach aims at

1. **Online adaptive learning**: Our target application scenarios are streaming data. This allows the inference during an experiment or as part of a neural prosthetics. If the system changes, the inference will catch up with the altered system parameters.
2. **Joint estimation**: The proposed method is supposed to simultaneously learn the latent states $p(\mathbf{x}_t \mid \mathbf{y}_{\leq t})$, state dynamics $f(\mathbf{x}_t, \mathbf{u}_t)$ and the observation model $G(\mathbf{x}, \mathbf{u})$. No offline training is necessary to learn the system parameters.
3. **Interpretability**: Under the framework of state space modeling, rather than interpret the system via model parameters, we employ the language of dynamical systems and capture the characteristics of the system qualitatively via fixed point, limit cycle, strange attractor, bifurcation and so on which are key components of theories of neural dynamics and computation.

We focus on low-dimensional latent dynamics that often underlie many neuroscientific experiments and allow for producing interpretable visualizations of complex collective network dynamics in this study.

2 Variational Principle for Online Joint Estimation

The crux of recursive Bayesian filtering is updating the posterior over the latent state one step at a time:

$$p(\mathbf{x}_t \mid \mathbf{y}_{\leq t}) = \underbrace{p(\mathbf{y}_t \mid \mathbf{x}_t)}_{\text{likelihood}} \underbrace{p(\mathbf{x}_t \mid \mathbf{y}_{<t})}_{\text{prior at time } t} / \underbrace{p(\mathbf{y}_t \mid \mathbf{y}_{<t})}_{\text{marginal likelihood}} \tag{3}$$

where the input $\mathbf{u}_t$ and parameters θ are omitted for brevity. Unfortunately, the exact calculations of Eq. (3) are not tractable in general, especially for nonlinear dynamic models and/or non-conjugate distributions. We thus turn to approximate inference and develop a recursive variational Bayesian filter by deriving an evidence lower bound for the marginal likelihood as the objective function. Let $q(\mathbf{x}_t)$ denote an arbitrary probability measure which will eventually approximate the filtering density $p(\mathbf{x}_t \mid \mathbf{y}_{\leq t})$. From Eq. (3), we can rearrange the marginal log-likelihood as

$$\begin{aligned}
&\log p(\mathbf{y}_t \mid \mathbf{y}_{<t}) \\
&= \log \frac{p(\mathbf{y}_t \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{y}_{<t})}{p(\mathbf{x}_t \mid \mathbf{y}_{\leq t})} \qquad \text{for any } \mathbf{x}_t \\
&= \mathbb{E}_{q(\mathbf{x}_t)} \left[\log \frac{p(\mathbf{y}_t \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{y}_{<t})}{p(\mathbf{x}_t \mid \mathbf{y}_{\leq t})} \right] \qquad \text{the marginal is constant to } q(\mathbf{x}_t) \\
&= \mathbb{E}_{q(\mathbf{x}_t)} \left[\log \frac{p(\mathbf{y}_t \mid \mathbf{x}_t) p(\mathbf{x}_t \mid \mathbf{y}_{<t}) q(\mathbf{x}_t)}{p(\mathbf{x}_t \mid \mathbf{y}_{\leq t}) q(\mathbf{x}_t)} \right] \\
&= \underbrace{\mathbb{E}_{q(\mathbf{x}_t)} \left[\log p(\mathbf{y}_t \mid \mathbf{x}_t) \right]}_{\text{reconstruction log-likelihood}} - \mathbb{D}_{\mathrm{KL}}(q(\mathbf{x}_t) \,\|\, p(\mathbf{x}_t \mid \mathbf{y}_{<t})) + \underbrace{\mathbb{D}_{\mathrm{KL}}(q(\mathbf{x}_t) \,\|\, p(\mathbf{x}_t \mid \mathbf{y}_{\leq t}))}_{\text{variational gap \#1}} \\
&\geq \mathbb{E}_{q(\mathbf{x}_t)} \left[\log p(\mathbf{y}_t \mid \mathbf{x}_t) \right] + \underbrace{\mathbb{H}(q(\mathbf{x}_t))}_{\text{entropy}} + \mathbb{E}_{q(\mathbf{x}_t)} \left[\log p(\mathbf{x}_t \mid \mathbf{y}_{<t}) \right] \\
&= \mathbb{E}_{q(\mathbf{x}_t)} \left[\log p(\mathbf{y}_t \mid \mathbf{x}_t) \right] + \mathbb{H}(q(\mathbf{x}_t)) + \mathbb{E}_{q(\mathbf{x}_t)} \left[\log \mathbb{E}_{p(\mathbf{x}_{t-1} \mid \mathbf{y}_{<t})} \left[p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \right] \right] \\
&= \mathbb{E}_{q(\mathbf{x}_t)} \left[\log p(\mathbf{y}_t \mid \mathbf{x}_t) \right] + \mathbb{H}(q(\mathbf{x}_t)) + \mathbb{E}_{q(\mathbf{x}_t)} \left[\mathbb{E}_{p(\mathbf{x}_{t-1} \mid \mathbf{y}_{<t})} \left[\log p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \right] \right] \\
&\quad + \underbrace{\mathbb{E}_{q(\mathbf{x}_t)} \left[\mathbb{D}_{\mathrm{KL}}(p(\mathbf{x}_{t-1} \mid \mathbf{y}_{<t}) \,\|\, p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{y}_{<t})) \right]}_{\text{variational gap \#2}} \\
&\geq \mathbb{E}_{q(\mathbf{x}_t)} \left[\log p(\mathbf{y}_t \mid \mathbf{x}_t) \right] + \mathbb{H}(q(\mathbf{x}_t)) + \mathbb{E}_{q(\mathbf{x}_t)} \mathbb{E}_{p(\mathbf{x}_{t-1} \mid \mathbf{y}_{<t})} \left[\log p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \right]
\end{aligned}$$

where $\mathbb{H}$ denotes Shannon's entropy and $\mathbb{D}_{\mathrm{KL}}$ denotes the Kullback-Leibler (KL) divergence [27]. Maximizing this lower bound would result in a variational posterior $q(\mathbf{x}_t) \approx p(\mathbf{x}_t \mid \mathbf{y}_{\leq t})$ w.r.t. $q(\mathbf{x}_t)$. Naturally we plug in the previous step's solution to the next time step, obtaining a loss function suitable for recursive estimation:

$$\mathcal{L} := \mathbb{E}_{q(\mathbf{x}_t)} \left[\log p(\mathbf{y}_t \mid \mathbf{x}_t) \right] + \mathbb{H}(q(\mathbf{x}_t)) + \underbrace{\mathbb{E}_{q(\mathbf{x}_t)} \mathbb{E}_{q(\mathbf{x}_{t-1})} \left[\log p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \right]}_{\text{dynamics log-likelihood}} \tag{4}$$

This also results in consistent $q(\mathbf{x}_t)$ for all time steps as they are in the same family of distribution.

Meanwhile, as it is aimed to jointly estimate the observation model $p(\mathbf{y}_t \mid \mathbf{x}_t)$ and state dynamics $p(\mathbf{x}_t \mid \mathbf{x}_{t-1})$, we achieve online inference by maximizing this objective $\mathcal{L}$ w.r.t. their parameters (omitted for brevity) and the variational posterior distribution $q(\mathbf{x}_t)$ simultaneously provided that $q(\mathbf{x}_{t-1})$ takes some parameterized form and has been estimated from the previous time step. Maximizing the objective $\mathcal{L}$ is equivalent to minimizing the two variational gaps: (1) the variational filtering posterior has to be close to the true filtering posterior, and (2) the filtering posterior from the previous step needs to be close to $p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{y}_{<t})$. Note that the second gap is invariant to $q(\mathbf{x}_t)$ if $p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{y}_{<t}) = p(\mathbf{x}_{t-1} \mid \mathbf{y}_{<t})$, that is, the one-step backward smoothing distribution is identical to the filtering distribution.

On the flip side, intuitively, there are three components in $\mathcal{L}$ that are jointly optimized: (1) reconstruction log-likelihood which is maximized if $q(\mathbf{x}_t)$ concentrates around the maximum likelihood estimate given only $\mathbf{y}_t$, (2) the dynamics log-likelihood which is maximized if $q(\mathbf{x}_t)$ concentrates at around the maximum of $\mathbb{E}_{q(\mathbf{x}_{t-1})}\left[\log p(\mathbf{x}_t \mid \mathbf{x}_{t-1})\right]$, and (3) the entropy that expands $q(\mathbf{x}_t)$ and prevents it from collapsing to a point mass.

In order for this recursive estimation to be real-time, we choose $q(\mathbf{x}_t)$ to be a multivariate normal with diagonal covariance $\mathcal{N}(\boldsymbol{\mu}_t, \mathbf{s}_t)$ where $\boldsymbol{\mu}_t$ is the mean vector and $\mathbf{s}_t$ is the diagonal of the covariance matrix in this study. Moreover, to amortize the computational cost of optimization to obtain the best $q(\mathbf{x}_t)$ on each time step, we employ the variational autoencoder architecture [28] to parameterize $q(\mathbf{x}_t)$ with a recognition model. Intuitively, the recognition model embodies the optimization process of finding $q(\mathbf{x}_t)$, that is, it performs an approximate Bayesian filtering computation (in constant time) of Eq. (3) according to the objective function $\mathcal{L}$. We use a recursive recognition model that maps $q(\mathbf{x}_{t-1})$ and $\mathbf{y}_t$ to $q(\mathbf{x}_t)$. In particular, the recognition model takes a deterministic recursive form:

$$[\boldsymbol{\mu}_t, \mathbf{s}_t] = h(\mathbf{y}_t, \mathbf{u}_{t-1}, \boldsymbol{\mu}_{t-1}, \mathbf{s}_{t-1}) \tag{5}$$

Specifically h takes a simple the form of the multi-layer perceptron (MLP) [29] in this study, and we refer to its parameters as the recognition model parameters. Note that the recursive architecture of the recognition model reflects the Markovian structure of the assumed dynamics (c.f., smoothing networks often use bidirectional recurrent neural network (RNN) [30] or graphical models [18, 20]).

The expectations appearing in the reconstruction log-likelihood and dynamics log-likelihood are not always tractable in general. For those intractable cases, one can use the reparameterization trick and stochastic variational Bayes [31, 32]: rewriting the expectations over q as expectation over a standard normal random variable, i.e., $\boldsymbol{\mu}_t + \mathbf{s}_t^{\frac{1}{2}}\mathcal{N}(0, 1)$, and using a single sample for each time step. Hence, in practice, we optimize the following objective function (the other variables and parameters are omitted for brevity),

$$\hat{\mathcal{L}} = \log p(\mathbf{y}_t \mid \tilde{\mathbf{x}}_t, \theta) + \mathbb{E}_{q(\mathbf{x}_t)} \log p(\mathbf{x}_t \mid \tilde{\mathbf{x}}_{t-1}, \theta) + H(q(\mathbf{x}_t)) \tag{6}$$

where $\tilde{\mathbf{x}}_t$ and $\tilde{\mathbf{x}}_{t-1}$ represent random samples from $q(\mathbf{x}_t)$ and $q(\mathbf{x}_{t-1})$ respectively. Note that the remaining expectation over $q(\mathbf{x}_t)$ has closed form solution under our Gaussian state noise, $\boldsymbol{\epsilon}_t$, assumption. Thus, our method can handle arbitrary observation and dynamic model unlike dual form nonlinear Kalman filtering methods that usually suffer difficulties in sampling, e.g. transforming Gaussian random numbers into point process observations.

The schematics of the proposed inference algorithm is summarized by two passes in Figure 1. In the forward pass, the previous latent state generates the new state through the dynamic model, and the new state transforms into the observation through the observation model. In the backward pass, the recognition model recovers the current latent state from the observation, and the observation model, recognition model and dynamic model, are updated by backpropagation.

Algorithm 1 is an overview of the recursive estimation algorithm. Denoting the set of all parameters by $\boldsymbol{\Theta}$ of the observation model, recognition model and dynamic models, the objective function in Eq. (6) is differentiable w.r.t. $\boldsymbol{\Theta}$, and thus we employ empirical Bayes and optimize it through stochastic gradient ascent (using Adam [33]). We outline the algorithm for a single vector time series, but we can filter multiple time series with a common state space model simultaneously, in which case the gradients are averaged across the instantiations. Note that this algorithm has *constant time and space complexity* per time step.

In practice, the measurements $\mathbf{y}_t$ and input $\mathbf{u}_t$ are sampled at a regular interval. Algorithm 1 is called after every such observation event, which will return the state estimate along with the parameters and the dynamical system. One can visualize these for real-time for monitoring, and/or have it streamed to another system for further automated processing (e.g. detect anomalies and raise an alarm or deliver feedback controls).

3 Application to Latent Neural Dynamics

Our primary applied aim is real-time neural interfaces where a population of neurons are recorded while a low-dimensional stimulation is delivered [34, 35, 36]. State-space modeling of such neural time series have been successful in describing population dynamics [37, 8]. Moreover, models of

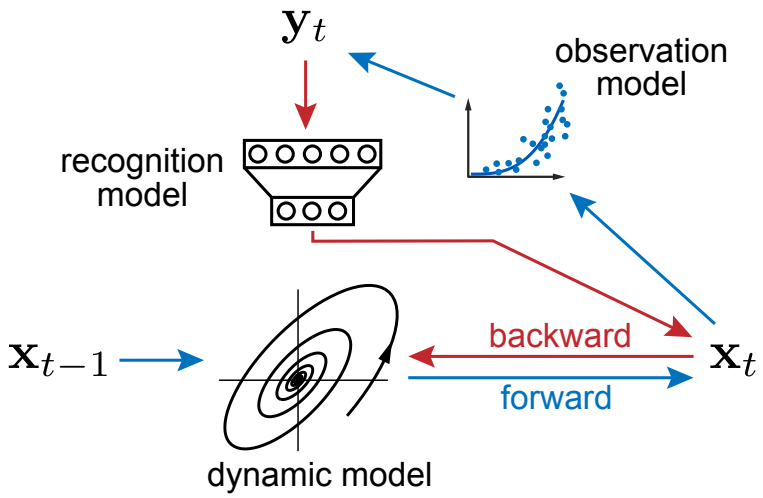

Figure 1: Schematics of variational joint filtering. Blue arrows indicates forward pass in which the previous latent state generates the new state through the state dynamics, and the new state transforms into the observation through the observation model. Red arrows indicate backward pass in which the recognition model recovers the current latent state from the observation. The three components, observation model, recognition model and dynamical system, are updated by backpropagation.

Algorithm 1 Variational Joint Filtering (single step)

procedure VJF($\mathbf{y}_t, \mathbf{u}_{t-1}, \boldsymbol{\mu}_{t-1}, \mathbf{s}_{t-1}, \boldsymbol{\Theta}$)
  $\epsilon_t \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\epsilon_{t-1} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$ ▷ Draw random samples
  $[\boldsymbol{\mu}_t, \mathbf{s}_t] \coloneqq \mathbf{h}(\mathbf{y}_t, \mathbf{u}_{t-1}, \boldsymbol{\mu}_{t-1}, \mathbf{s}_{t-1})$ ▷ State estimation
  $\tilde{\mathbf{x}}_t \coloneqq \boldsymbol{\mu}_t + \mathbf{s}_t^{1/2} \epsilon_t$
  $\tilde{\mathbf{x}}_{t-1} \coloneqq \boldsymbol{\mu}_{t-1} + \mathbf{s}_{t-1}^{1/2} \epsilon_{t-1}$
  Update $\boldsymbol{\Theta}$ with $\nabla_{\boldsymbol{\Theta}} \hat{\mathcal{L}}(\boldsymbol{\Theta}; \mathbf{y}_t, \tilde{\mathbf{x}}_t, \tilde{\mathbf{x}}_{t-1}, \mathbf{u}_{t-1})$ ▷ Model update
  return $\boldsymbol{\mu}_t, \mathbf{s}_t$ and $\boldsymbol{\Theta}$
end procedure

neural computation are often described as dynamical systems [38, 39, 40]. For example, attractor dynamics where the convergence to one of the attractors represents the result of computation [41, 42]. Here we propose a parameterization and tools for visualization of the model suitable for studying neural dynamics and building neural interfaces [9]. In this section, we provide methodological details for the results presented in the next section.

3.1 Parameterization of the state space model

Having in mind high-temporal resolution neural spike trains where each time bin has at most one action potential per channel, we describe the case for point process observation. However, note that our method generalizes to arbitrary observation likelihoods that are appropriate for other modalities, including calcium imaging or local field potentials. The observed point process time series $\mathbf{y}_t$ is a stream of sparse binary vectors. All experiments of point process observation were binned finely so that the time bins contain one event each at most in this study.

Our observation model, Eq. (7), assumes that the observation vector $\mathbf{y}_t$ is sampled from a probability distribution P determined by the latent state $\mathbf{x}_t$ though a linear-nonlinear map possibly together with extra parameters at each time t,

$$\mathbf{y}_t \sim P(g(\mathbf{C}\mathbf{x}_t + \mathbf{b})) \tag{7}$$

where $g \colon \mathbb{R} \to \mathbb{R}$ is a point-wise map. We use the canonical link $g(\cdot) = \exp(\cdot)$ for Poisson likelihood and identity for Gaussian likelihood in this study. Note that this observation model is not identifiable since $\mathbf{C}\mathbf{x}_t = (\mathbf{C}\mathbf{R})(\mathbf{R}^{-1}\mathbf{x}_t)$ where $\mathbf{R}$ is an arbitrary invertible matrix. We normalize the loading

matrix $\mathbf{C}$ in each iteration. It is straightforward to include more additive exogenous variables, history-filter for refractory period, coupling between processes, and stimulation artifacts [43, 44].

For state dynamic model, we propose to use a specific additive parameterization with state transition function and input interaction as a special case of Eq. (2),

$$\mathbf{x}_{t+1} = \mathbf{x}_t + f(\mathbf{x}_t) + \mathbf{B}_t\mathbf{u}_t + \boldsymbol{\epsilon}_{t+1} \tag{8a}$$

$$f(\mathbf{x}_t) = \mathbf{W}\boldsymbol{\phi}(\mathbf{x}_t) \tag{8b}$$

$$\mathbf{x}_0, \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}) \tag{8c}$$

where $\boldsymbol{\phi}(\cdot)$ is a vector of r continuous basis functions, i.e. $\boldsymbol{\phi}(\cdot) = (\phi_1(\cdot), \ldots, \phi_r(\cdot))^\top$, $\mathbf{W}$ is the weight matrix of the radial basis functions, and $\mathbf{B}_t$ is the interaction with the input $\mathbf{u}_t$. The interaction $\mathbf{B}_t$ can be globally linear, parameterized as a matrix independent from $\mathbf{x}_t$, or locally linear, parameterized as a matrix-valued function of $\mathbf{x}_t$ using also RBF networks. i.e. $\mathrm{vec}(\mathbf{B}(\mathbf{x}_t)) = \mathbf{W}_B\boldsymbol{\phi}(\mathbf{x}_t)$ where $\mathbf{W}_B$ is the respective weight matrix. In this study, we used squared exponential radial basis functions [4, 7, 6, 9],

$$\phi_i(\mathbf{x}) = \exp\left(-\frac{1}{2}\gamma_i\|\mathbf{x} - \mathbf{c}_i\|_2^2\right) \tag{9}$$

with centers $\mathbf{c}_i$ and corresponding inverse squared kernel width γ_i. Though the dynamics can be modeled by other universal approximators such as percepton and RNN, we chose the radial basis function network for the reasons of non-wild extrapolation (zero velocity when the state is far away from data) and fast computation.

The time complexity of our algorithm is $\mathcal{O}(mpr + n(m + p + q))$ where n, m, p, q, r denote the dimensions of observation, latent space, input, the numbers of hidden units and radial basis functions for this specific parameterization. Practically to achieve realistic computation time for real-time applications in neuroscience, the number of radial basis functions and hidden units are constrained by the requirement. Note that the time complexity does not grow with time that enable efficient online inference. If we compare this to an efficient offline algorithm such as PLDS [37] run repeatedly for every new observation ("online mode"), its time complexity is $\mathcal{O}(t \cdot (m^3 + mn))$ per time step at time t which grows as time passes, making it impractical for real-time application.

3.2 Phase portrait analysis

Phase portrait displays key qualitative features of dynamics, and with a little bit of training, it provides a visual means to interpreting dynamical systems. The law that governs neural population dynamics captured in the inferred function $f(\mathbf{x})$ directly represents the velocity field of an underlying smooth dynamics (1a) in the absence of input [4, 9]. In the next section, we visualize the estimated dynamics as phase portrait which consists of the vector field, example trajectories, and estimated dynamical features (namely fixed points) [45]. We can numerically identify candidate fixed points $\mathbf{x}^*$ that satisfy $f(\mathbf{x}^*) \approx 0$. For the synthetic experiments, we performed an affine transformation to orient the phase portrait to match the canonical equations in the main text when the simulation is done through the proposed observation model if the observation model is unknown and estimated.

3.3 Prediction

For state space models, we can predict both future latent trajectory and future observations. The s-step ahead prediction can be sampled from the predictive distributions:

$$p(\mathbf{x}_{t+1:t+s} \mid \mathbf{y}_{\leq t}) = \mathbb{E}_{q(\mathbf{x}_t)}[p(\mathbf{x}_{t+1:t+s} \mid \mathbf{x}_t)] \tag{10a}$$

$$p(\mathbf{y}_{t+1:t+s} \mid \mathbf{y}_{\leq t}) = \mathbb{E}_{p(\mathbf{x}_{t+1:t+s} \mid \mathbf{y}_{\leq t})}[p(\mathbf{y}_{t+1:t+s} \mid \mathbf{x}_{t+1:t+s})] \tag{10b}$$

given estimated parameters by current time t without seeing the data $\mathbf{y}_{t+1:t+s}$ during these steps. In the figures of experiments, we plot the mean of the predictive distribution as trajectories.

4 Experiments on Theoretical Models of Neural Computation

We demonstrate our method on a range of nonlinear dynamical systems relevant to neuroscience. Many theoretical models have been proposed in neuroscience to represent different schemes of

computation. For the purpose of interpretable visualization, we choose to simulate from two or three dimensional dynamical systems. We apply the proposed method to four such low-dimensional models: a ring attractor model as a model of internal head direction representation, an nonlinear oscillator as a model of rhythmic population-wide activity, a biophysically realistic cortical network model for a visual discrimination experiment and a chaotic attractor.

In the synthetic experiments, we first simulated state trajectories by respective differential equations, and then generated either Gaussian or point process observations (to mimic spikes) via Eq. (7) with corresponding distributions. The parameters $\mathbf{C}$ and $\mathbf{b}$ were randomly drawn, and they were constrained to keep firing rate < 60 Hz on average for realistic spiking behavior. All observations are spatially 200-dimensional unless otherwise mentioned. We refer to their conventional formulations under different coordinate systems, but our simulations and inferences are all done in Cartesian coordinates. Note that we focus on online learning in this study and always train our model with streaming data, even while comparing with offline methods.

The approximate posterior distribution is defined recursively in Eq. (5) as diagonal Gaussian with mean and variance determined by corresponding observation, input and previous step via a recurrent neural network. We used a one-hidden-layer MLP in this study. Typically the state noise variance σ^2 is unknown and has to be estimated from data. To be consistent with Eq. (8c), we set the starting value of σ^2 to be 1, and hence $\boldsymbol{\mu}_0 = \mathbf{0}, \mathbf{s}_0 = \mathbf{I}$. We initialize the loading matrix $\mathbf{C}$ by factor analysis, and column-wisely normalize it by ℓ_2 norm every iteration to keep the system identifiable.

4.1 Ring attractor

Continuous attractors are often used as models for neural representation of continuous variables [6, 5]. For example, a bump attractor network with ring topology is proposed as the dynamics underlying the persistently active set of neurons that are tuned for the angle of the animal's head direction [46]. Here we use the following 2-variable reduction of the ring attractor system: First, we study the following two-variable ring attractor system:

$$\begin{aligned} \tau_r \dot{r} &= r_0 - r \\ \tau_\varphi \dot{\varphi} &= I \end{aligned} \tag{11}$$

where φ represents the direction driven by input I, and r is the radial component representing an internal circular variable, such as head direction. We simulated 100 trajectories (1000 steps) with step size $\Delta t = 0.1$, $r_0 = 1$, $\tau_r = 1$, $\tau_\varphi = 1$ with Gaussian state noise (std $= 0.005$) added each step. Though the ring attractor is defined in polar coordinate system, we transformed it into Cartesian system for simulation and training. In simulation we used strong input (tangent drift) to keep the trajectories flowing around the ring clockwise or counter-clockwise. The point process observations were generated by passing the states through a linear-nonlinear map (Eq. (7)) and sampling from a Poisson distribution. We streamed the observations into the proposed algorithm that consists of point process likelihood, dynamic model with 20 radial basis functions and locally linear input interaction in Eq. (2) and a recognition MLP with 100 hidden units.

Figure 2A illustrates one latent trajectory (black) and its variational posterior mean (blue). These two trajectories start at green circle and diamond respectively and end at the red markers. The inference starts near the center (origin) that is relatively far from the true location because the initial posterior mean is set at zero. The final states are very close which implies that the recognition model works well. Figure 2B shows the reconstructed velocity field by the model. We visualized the velocity as colored directional streamlines. We can see the velocity toward the ring attractor and the speed is smaller closer to the ring. The model also identifies a number of fixed points arranged around the ring attractor via numerical roots finding. Figure 2C shows the distribution of posterior means of all data points in the state space. We have more confidence of the inferred dynamical system in the denser area.

Figure 2D shows the three components of Eq. (4) and the objective lower bound clearly, demonstrating the convergence of the algorithm. We can see each component reaches a plateau within 400 sec. As the reconstruction and dynamics log-likelihoods increase, the recognition model and dynamical model are getting more accurate while the decreasing entropy indicates the increasing confidence (inverse posterior variance) on the inferred latent states. The average computation time of a joint estimation step is 1.1 ms (hardware specification: Intel Xeon E5-2680 2.50G Hz, 128GB RAM, no GPU).

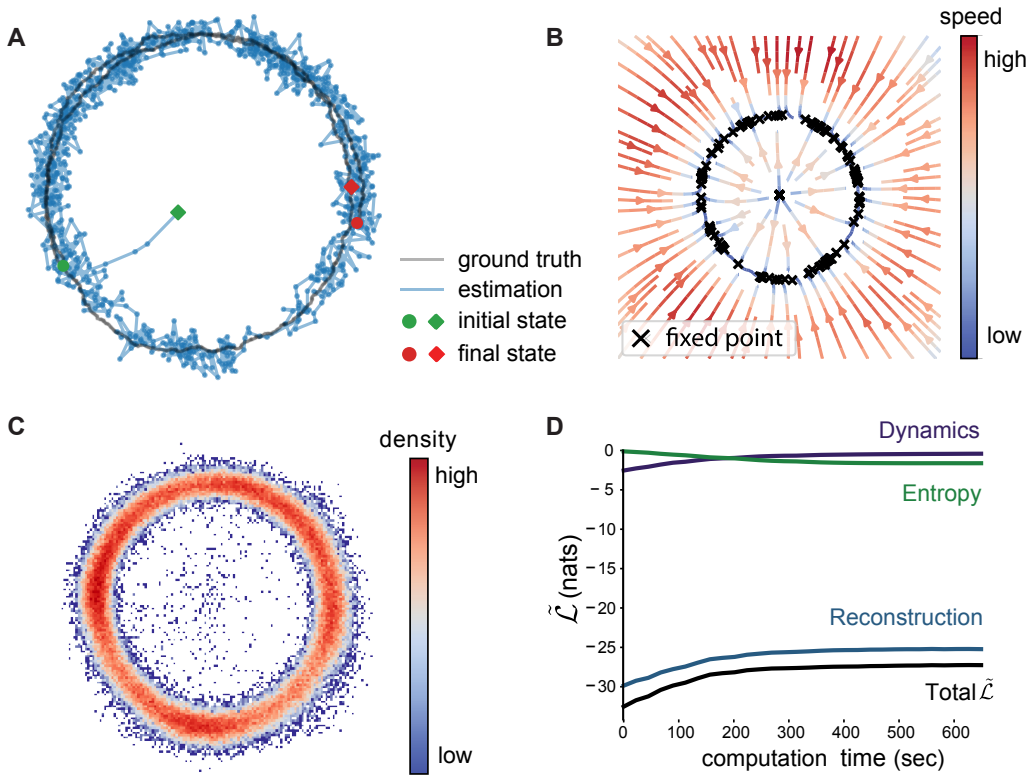

Figure 2: Ring attractor model. **(A)** One latent trajectory (black) in the training set and the corresponding filtered mean μ_t (blue). **(B)** Velocity field reconstructed from the trained proposed model. The colored streamlines indicates the speed and the directions. The black crosses are candidate fixed points obtained from inferred dynamics. Note the collection of fixed points around the ring shape. The central fixed point is unstable. **(C)**Density of the posterior means. The density of inferred means of all trajectories in the training set. The higher it is, the more confidence we have on the inferred dynamics where we have more data. **(D)** Convergence on the ring attractor. We display the three components of the objective lower bound: reconstruction log-likelihood, dynamics log-likelihood, entropy, and the lower bound itself from Eq. (4). The average computation time per step is 1.1 ms (more than 900 data points per sec).

4.2 Nonlinear oscillator

Dynamical systems have been a successful application in the biophysical models of single neuron in neuroscience. We used a relaxation oscillator, the FitzHugh-Nagumo (FHN) model [47], which is a 2-dimensional reduction of the Hodgkin-Huxley model: We used a with the following nonlinear state dynamics:

$$\begin{aligned} \dot{v} &= v(a-v)(v-1) - w + I, \\ \dot{w} &= bv - cw, \end{aligned} \tag{12}$$

where v is the membrane potential, w is a recovery variable and I is the magnitude of stimulus current in modeling single neuron biophysics. This model was also used to model global brain state that fluctuates between two levels of excitability in anesthetized cortex [48]. We use the following parameter values $a = -0.1$, $b = 0.01$, $c = 0.02$ and $I = 0.1$ to simulate 100 trajectories of 1000 steps with step size 0.5 and Gaussian noise (std=0.002). At this regime, unlike the ring attractor, the spontaneous dynamics is a periodic oscillation, and the trajectory follows a limit cycle. The point process observations were also sampled via the observation model of the same parametric form as that of the ring attractor example. We used 20 radial basis functions for dynamic model and 100 hidden units for recognition model. While training the model, the input was clamped to zero, and expect the model to learn the spontaneous oscillator.

We compare the state estimation with the standard particles filtering (PF) which are powerful online methods theoretically capable of producing arbitrarily accurate filtering distribution. We run two variants of the particle filter with different proposal distributions. One used diffusion as the proposal, i.e. $\mathbf{x}_t = \mathbf{x}_{t-1} + \epsilon_t$ where $\mathbf{x}$ is the vector of state variables v and w, and the other, a.k.a. bootstrap particle filter [49], used the true dynamics in Eq. (12). We provided the true parameters for the observation model and noise term to PF which gives them an advantage. Both particle filters and VJF were run on 50 realizations of 5000-step long observation series. Figure 3 shows the root mean squared deviations (RMSE) (mean and standard error over 50 realizations). It is expected that the bootstrap particle filter outperformed the diffusion particle filter since the former utilized the true dynamics. One can see the state estimation by VJF improved as learning carrying on and eventually outperformed both particle filters. Note that VJF had to learn the parameters of likelihood, dynamic model and recognition model during the run. We varied the number of RBFs (20 and 30) but the results are not substantially different.

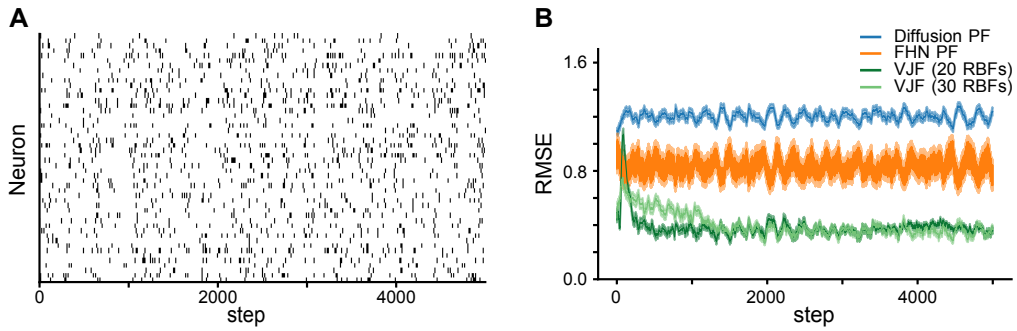

Figure 3: Nonlinear oscillator (FitzHugh-Nagumo) observation and state estimation. **(A)** The synthetic spike train. **(B)** RMSEs (the shades are s.e.m.) of state estimation on the observation in (A). We report the RMSE of estimated states among the proposed method VJF and two particle filters, one with diffusion dynamics and the other with the true FHN dynamics. Varying the number of RBFs did not substantially change the quality of the results.

We also reconstructed the phase portrait (Fig. 4B) comparing to the truth (Fig. 4C). The two dashed lines are the theoretical nullclines of the true model on which the velocity of corresponding dimension is zero. The reconstructed field shows a low speed valley overlapping with the nullcline especially on the right half of the figure. At the intersection of the two nullclines there is an unstable fixed point. We can see the identified fixed point is close to the intersection. As most of the trajectories lie on the oscillation path (limit cycle) with merely few data points elsewhere, the inferred system shows the oscillation dynamics similar to the true system around the data region. The difference mostly happens in the region far from the trajectories because of the lack of data.

We run a long-term prediction using VJF without seeing the future data $\mathbf{y}_{t+1:T}$ during these steps ($T = 1000$ steps $= 1$ sec) beginning at the final state of training data. We show the truth and prediction in figure 4D. The upper row is the true latent trajectory and corresponding observations. The lower row is the filtered trajectory and prediction by the proposed method. The light-colored parts are the 500 steps of inference before prediction and the solid-colored parts are 1000 steps truth and prediction. We also show the sample observations from the trained observation model during the prediction period.

One of the popular latent process modeling tools for point process observation that can make prediction is the Poisson Linear Dynamical System (PLDS) [37] which assumes latent linear dynamics. We compared PLDS fit with EM on its long-term prediction on both the states and spike trains (Fig. 4). This demonstrates the nonlinear dynamical model outperforming the linear model even in the unfair online setting.

To compare to the methods with nonlinear dynamical models, we also run latent factor analysis via dynamical systems (LFADS) [50] offline using the same data. LFADS implements its dynamical model with the gated recurrent unit (GRU) [51] that requires high dimensions. For this 2D system, we tried different GRU dimensionalities. We made minimal changes to its recommended setting

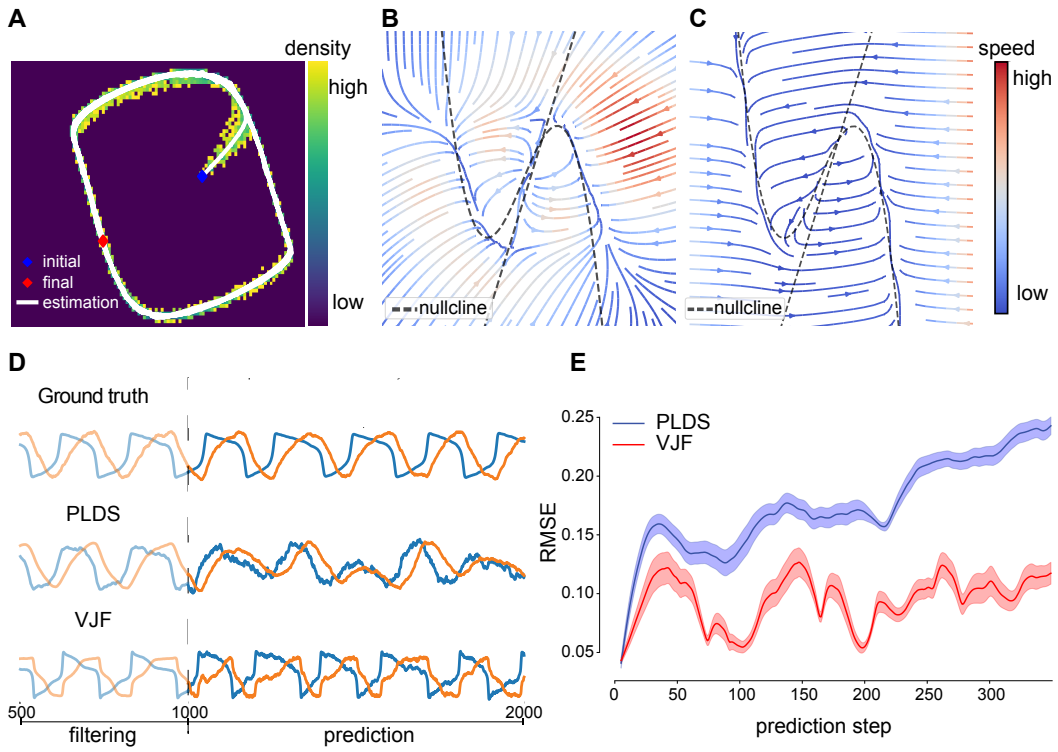

Figure 4: Nonlinear oscillator (FitzHugh-Nagumo) dynamical system and prediction. **(A)** One inferred latent trajectory and the density of posterior means of all trajectories. Most of the inferred trajectory lie on the oscillation path. **(B)** Velocity field reconstructed by the inferred dynamical system. **(C)** Velocity field of the true dynamical system. The dashed lines are two nullclines of the true model on which the gradients are zero so as the velocity. **(D)** 1000-step prediction continuing the trajectory and sampled spike trains compared to ground truth from (A). **(E)** Mean (solid line) and standard error (shade) of root mean square error of prediction of 2000 trials. The prediction started at the same states for the true system and models. Note that PLDS fails to predict long term due to its linear dynamics assumption. A linear dynamical system without noise can only produce damped oscillations.

including only the generator dimensionality, batch and no controller. The result shows that LFADS requires much higher dimension than the true system to capture the oscillation (Fig. S1). (The figure of its inferred trajectories is shown in the supplement.) We report the fitted log-likelihood per time bin -0.1274, -0.1272 and -0.1193 for 2D, 20D and 50D GRU respectively. In comparison, the log-likelihood of the proposed approach is -0.1142 with a 2D dynamical model (higher the better).

4.3 Fixed point attractor for decision-making

Perceptual decision-making paradigm is a well-established cognitive task where typically a low-dimensional decision variable needs to be integrated over time, and subjects are close to optimal in their performance. To understand how the brain implements such neural computation, many competing theories have been proposed [40, 5, 52, 41, 53]. We test our method on a simulated biophysically realistic cortical network model for a visual discrimination experiment [41]. In the model, there are two excitatory subpopulations that are wired with slow recurrent excitation and feedback inhibition to produce attractor dynamics with two stable fixed points (Fig. 5A). Each fixed point represents the final perceptual decision, and the network dynamics amplify the difference between conflicting inputs and eventually generates a binary choice.

Note that, different from the former examples that use a linear-nonlinear map of latent states, the point process observations (spikes) of this experiment were directly sampled from the spiking neural

network [1](1 ms binwidth) that was governed by its own high-dimensional intrinsic dynamics. It is filling the gap between fully specified state space models and real neuron populations.

We subsampled 480 selective neurons out of 1600 excitatory neurons from the simulation to be observed by our algorithm. The simulated data is organized into decision-making trials where each trial lasts for 2 sec and with different strength of visual evidence, controlled by "coherence". Our method with 20 radial basis functions learned the dynamics from 140 training trials (20 per coherence level c, $c = -1, -0.2, -0.1, 0, 0.1, 0.2, 1$).

Figure 5C shows the velocity field at zero coherence stimulus as colored streamlines. Note that our approach did not have prior knowledge of the network dynamics as the mean-field reduction [53] in Figure 5B. Although the absolute arrangement is dissimilar, the topology and relation of the five identified fixed points show correspondence with the mean-field reduction. The inference was completely data-driven (partial observation of spike trains) while the mean-field method required knowing the true dynamical model of the network and careful approximation by [53]. We showed that our method can provide a qualitatively similar result to the theoretical work which reduces the dimensionality and complexity of the original network.

4.4 Chaotic dynamics

Chaotic dynamics (or edge-of-chaos) have been postulated to support asynchronous states in the cortex, and neural computation over time by generating rich temporal patterns [54, 55]. We consider the 3-dimensional standard Lorenz attractor as an example chaotic system to demonstrate the flexibility of our method. We simulated 216 latent trajectories from:

$$\dot{x} = 10(y - x), \quad \dot{y} = x(28 - z) - y, \quad \dot{z} = xy - \frac{8}{3}z. \tag{13}$$

The each coordinate of the initial states are on the uniform grid of 6 values in $[-50, 50]$ inclusively, of which the combination results in 216 unique states. We discarded the first 500 transient steps of each trajectory and then use the following 1000 steps. We generated 200-dimensional Gaussian observations driven by the trajectories. Figure 6A shows estimated latent trajectory and the ground truth. One can see that the estimation lies in a similar manifold. In addition, we predicted 500 steps of future latent states without knowing the respective observations. Figure 6B shows 4 predicted trajectories starting from different initial states. One can see that the inferred system could generate qualitatively similar trajectory at most initial states but not perfectly for the true system is chaotic.

4.5 Nonstationary system

Another feature of our method is that its state dynamics estimate never stops. As a result, the algorithm is adaptive, and can potentially track slowly varying (nonstationary) latent dynamics. To test this feature, we compared a dual EKF and the proposed approach on nonstationary linear dynamical system. A spiral-in linear system was suddenly changed from clockwise to counter-clockwise at the 2000th step and the latent state was perturbed (Fig. 7). To adapt EKF, we used Gaussian observations that were generated through linear map from 2-dimensional state to 200-dimensional observation with additive noise ($\mathcal{N}(0, 0.5)$). To focus on the dynamics, we fixed all the parameters except the transition matrix for both methods, while our approach still has to learn the recognition model in addition. Figure 7 shows that our approach achieved better online performance as dual EKF in this experiment.

5 Real Neurophysiological Application

We applied the proposed method to a large scale recording to validate that it picks up meaningful dynamics. The dataset [56] consists of 148 simultaneously recorded single units from the primary cortex (V1) while directional drifting gratings were presented to an anesthetized monkey for around 1.3s per trial (Fig. 8A). We used the spike trains from 63 well-tuned units. The spike times were binned with a 1ms window (max 1 spike per bin). There is one continuous circular variables in the stimuli space: temporal phase of oscillation induced by the drifting gratings.

[1]The detail of the spiking neural network can be found in [41] and the code can be found at `https://github.com/xjwanglab/book/tree/master/wang2002`.

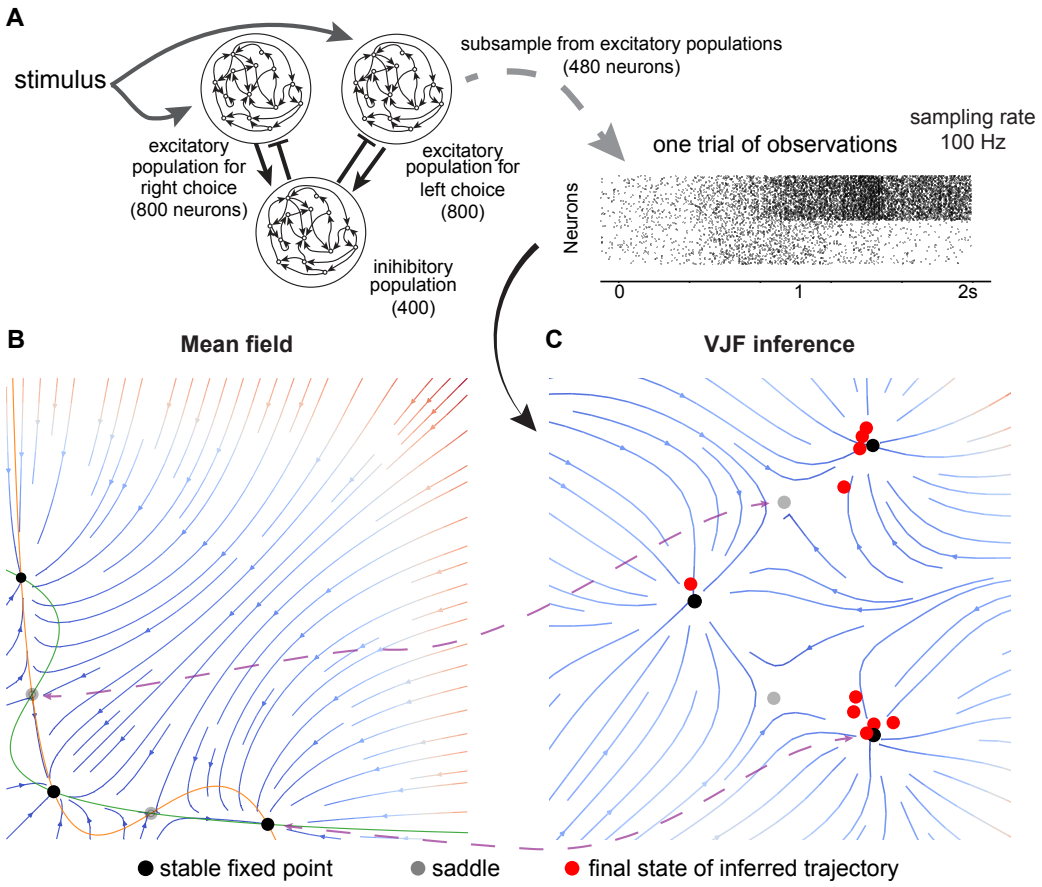

Figure 5: Fixed point attractor for decision-making. **(A)** Schematics of the neural network. There are two excitatory populations that are wired with slow recurrent excitation and feedback inhibition to produce attractor dynamics. The simulation was organized into decision-making trials. Each trial begins with a 0.5s period of spontaneous activity, and then the input is given to the two excitatory populations for 1.5s. We subsampled 480 selective neurons out of 1600 excitatory neurons from the simulation to be observed by our algorithm. **(B)** Mean field reduction of the network. Theoretical work has shown that the collective population dynamics can be reduced to 2 dimensions [53]. **(C)** VJF inferred dynamical model. The red dots are the inferred final states of zero-coherent trials. The black dots are fixed points (the solid are stable and the gray are unstable). Although the absolute arrangement is dissimilar, the topology and relation of the five identified fixed points show correspondence (indicated by purple lines).

A partial warm-up helps with the training. We chose a good initialization for the observation model, specifically the loading matrix and bias. There are 72 motion directions in total, each repeated 50 trials. We used the trials corresponding to 0 deg direction to initialize the observation model with dimensionality reduction methods such as variational latent Gaussian processes [8], and then trained VJF with a 2D dynamic model fully online on the trials corresponding to 180 deg direction that it had not seen before. Since we do not have long enough continuously-recorded trials, we concatenated the trials (equivalent to 500s) as if they were continuously recorded to mimic an online setting. As expected, Figure 8B and C shows the inferred dynamical system is able to implement the oscillation. The two goodness of fit measures (log-likelihood and ELBO) in Figure 8D shows that our method benefits from but does not necessarily require such a warm-up. The model with warm-up initialization has better starting goodness of fit than the random initialized model but the random initialized model eventually achieved similar goodness of fit with adequate amount of data.

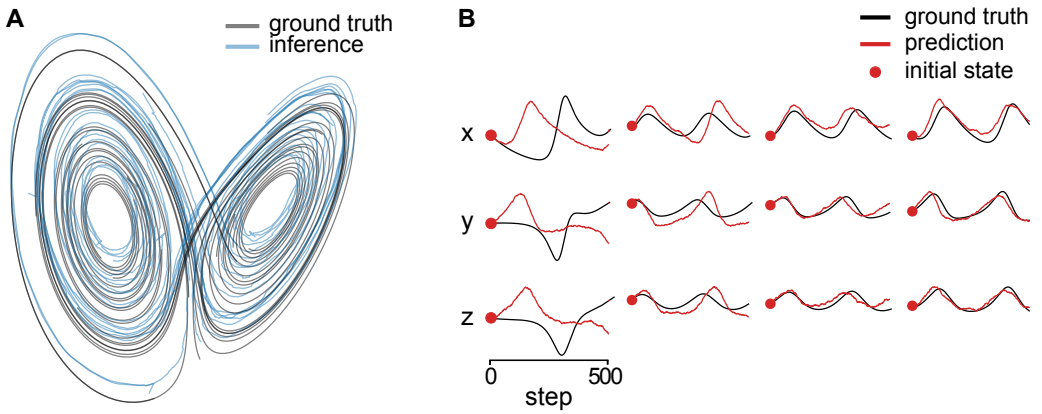

Figure 6: Lorenz attractor. **(A)** Estimated state trajectory (blue) and the ground truth (black) in 3D. **(B)** We predict 500 steps of future latent states (without knowing the respective observations) starting from 4 different initial states (red dots) using the inferred dynamical system. The red lines are the prediction and the black lines are the corresponding ground truth state.

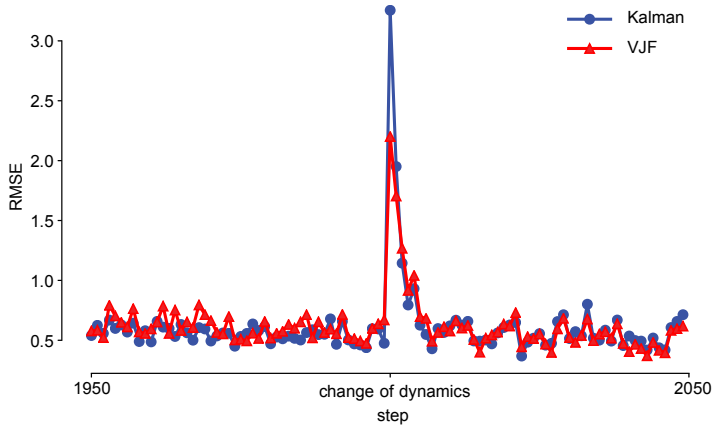

Figure 7: Prediction of nonstationary dynamical system. The colored curves (blue: EKF, red: VJF) are the mean RMSEs of one-step-ahead prediction of nonstationary system during online learning (50 trials). The linear system was changed and the state was perturbed at the 2000th step (center). The lines are average RMSEs. Both online algorithms quickly learned the change after a few steps.

6 Discussion

Neurotechnologies for recording the activity of large neural populations during meaningful behavior provide exciting opportunities for investigating the neural computations that underlie perception, cognition, and decision-making. However, the datasets provided by these technologies currently require sophisticated offline analyses that slow down the scientific cycle of experiment, data analysis, hypothesis generation, and further experiment. Moreover, in closed-loop neurophysiological setting, real-time adaptive algorithms are extremely valuable [57].

To fulfill this demand, we proposed an online algorithm for recursive variational Bayesian inference that simultaneously performs system identification and state filtering under the framework of state space modeling, in hope that it can greatly impact neuroscience research and biomedical engineering. There is no other method capable of all features, hence we compared several methods in different measures, often giving them the advantage. We showed that our proposed method consistently outperforms the state-of-the-art methods.

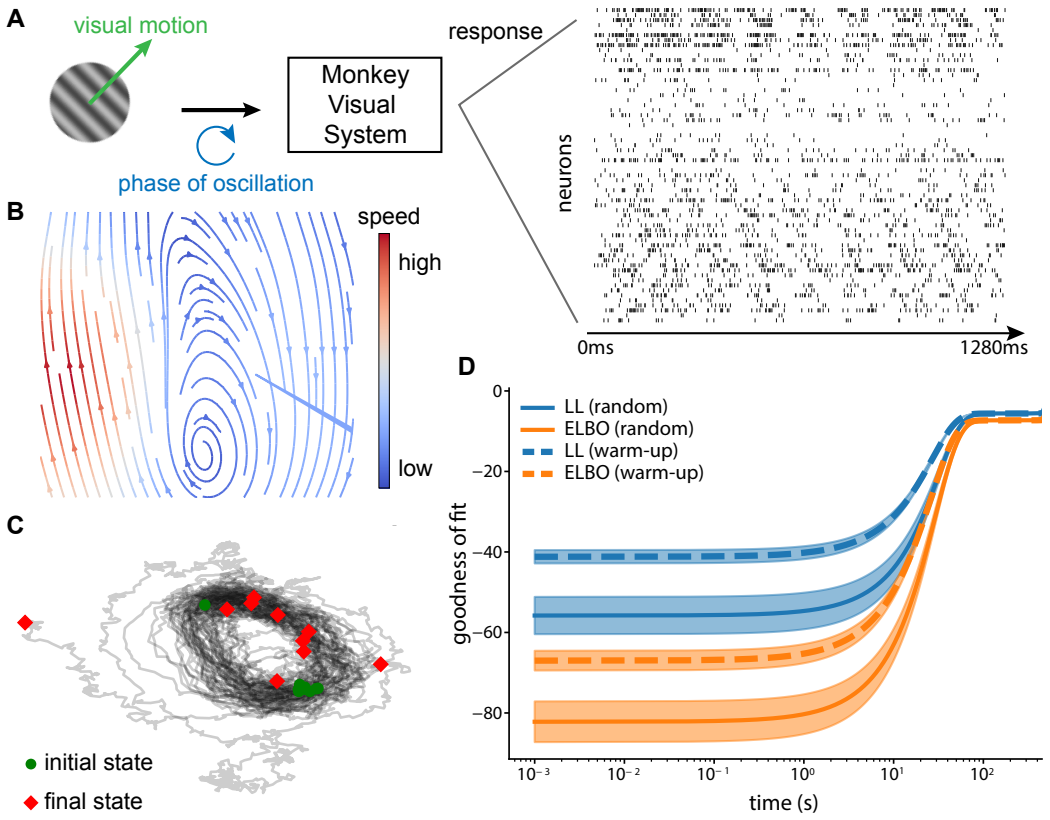

Figure 8: **(A)** Neurophysiological experiment. Drifting gratings were shown to the monkey (on the left). The neural spike trains (63 neurons, 1280 ms) from area V1 during the motion onset were recorded (on the right). Each row is one neuron and the binwidth is 1 ms. The phase of the oscillation forms a circular variable. **(B)** Phase portrait of the inferred dynamical system (arrows: direction, blue: low speed, red: high speed). The flow shows that the inferred system forms an oscillator. **(C)** Trajectories simulated from the inferred dynamical system. We simulated 10 state trajectories using the inferred system with random initial states (1000 steps each, black lines: trajectories, green circles: initial states, red diamonds: final states). The trajectories also confirm that the inferred system captured the oscillation underlying the data. **(D)** Convergence of the online method in terms of its goodness-of-fit. We calculated two goodness-of-fit measures (mean $\pm$ standard deviation, 10 repetitions), log-likelihood (LL) and ELBO for two strategies of initializing the observation model, warm-up and random initialization. Warm-up indicates that we initialized the observation model using dimensionality reduction methods before VJF; Random initialization indicates that the parameters of observation model were randomly drawn and learned completely by VJF.

Using the language of dynamical systems, we interpret the target system not via model parameters but via dynamical features: fixed points, limit cycles, strange attractors, bifurcations and so on. In our current approach, this interpretation heavily relies on visual inspection of the qualitative nonlinear dynamical system features. In contrast, most popular state space models assume linear dynamics [11, 37, 58] which is appropriate for smoothing latent states, but not expressive enough to recover the underlying vector field. Recently the Koopman theory that allows representation of general nonlinear dynamics as linear operators in infinite dimensional spaces [59] has gained renewed interest in modeling nonlinear dynamics. Although elegant in theory, in practice, however, the Koopman operators need to be truncated to a finite dimensional space with linear dynamics [60]. We note that the resulting linear models do not allow for topological features such as multiple isolated fixed points, nonlinear continuous attractors, stable limit cycles—features critical for non-trivial neural computation.

Our algorithm is highly flexible and general—it allows a wide range of observation models (likelihoods) and dynamic models, is computationally tractable, and produces interpretable visualizations of complex collective network dynamics. Our key assumption is that the dynamics consists of a continuous and slow flow, which enable us to parameterize the velocity field directly. This assumption reduces the complexity of the nonlinear function approximation, thus it is easy to identify the fixed/slow points. Specifically we chose the radial basis function network to model the dynamics for our experiments, which regularizes and encourages the dynamics to occupy a finite phase volume around the origin.

Our method has a number of hyperparameters. In the experiments, the differentiable hyperparameters were learnt via gradient descent while the selection of the other hyperparameters were made simple. In general, our method was robust; Perturbing the number of RBFs did not produce qualitatively different results (Fig. 3). [61] discussed growing radial basis function network adaptively which could be incorporated in our method to enable online tuning of the number of RBFs. The depth and width of neural networks were chosen empirically to improve the interpretability of resulting dynamical systems, but tuning did not result in large changes in the results.

This work opens many avenues for future work. One direction is to apply this model to large-scale neural recording from a behaving animal. We hope that further development would enable on-the-fly analysis of high-dimensional neural spike train during electrophysiological experiments. Clinically, a nonlinear state space model provides a basis for nonlinear feedback control as a potential treatment for neurological diseases that arise from diseased dynamical states.

References

[1] S. Haykin and J. Principe. Making sense of a complex world [chaotic events modeling]. *IEEE Signal Processing Magazine*, 15(3):66–81, May 1998.

[2] J. Ko and D. Fox. GP-BayesFilters: Bayesian filtering using gaussian process prediction and observation models. *Autonomous Robots*, 27(1):75–90, 5 2009.

[3] C. L. C. Mattos, Z. Dai, A. Damianou, et al. Recurrent gaussian processes. *International Conference on Learning Representations (ICLR)*, 2016.

[4] S. Roweis and Z. Ghahramani. *Learning nonlinear dynamical systems using the expectation-maximization algorithm*, pages 175–220. John Wiley & Sons, Inc, 2001.

[5] V. Mante, D. Sussillo, K. V. Shenoy, and W. T. Newsome. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503(7474):78–84, November 2013.

[6] D. Sussillo and O. Barak. Opening the black box: Low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Computation*, 25(3):626–649, March 2013.

[7] R. Frigola, Y. Chen, and C. E. Rasmussen. Variational gaussian process state-space models. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, pages 3680–3688, Montreal, Canada, 2014.

[8] Y. Zhao and I. M. Park. Variational latent Gaussian process for recovering single-trial dynamics from population spike trains. *Neural Computation*, 29(5), May 2017.

[9] Y. Zhao and I. M. Park. Interpretable nonlinear dynamic modeling of neural trajectories. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[10] A. A. Russo, S. R. Bittner, S. M. Perkins, et al. Motor cortex embeds muscle-like commands in an untangled population response. *Neuron*, 97(4):953–966, feb 2018.

[11] Y. Ho and R. Lee. A Bayesian approach to problems in stochastic estimation and control. *IEEE Transactions on Automatic Control*, 9(4):333–339, October 1964.

[12] S. Särkkä. *Bayesian filtering and smoothing*. Cambridge University Press, 2013.

[13] S. S. Haykin. *Kalman filtering and neural networks*. Wiley, 2001.

[14] Z. Ghahramani and S. T. Roweis. Learning nonlinear dynamical systems using an EM algorithm. In M. J. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 431–437. MIT Press, 1999.

[15] H. Valpola and J. Karhunen. An unsupervised ensemble learning method for nonlinear dynamic State-Space models. *Neural Computation*, 14(11):2647–2692, November 2002.

[16] R. Turner, M. Deisenroth, and C. Rasmussen. State-space inference and learning with gaussian processes. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 868–875, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[17] M. D. Golub, S. M. Chase, and B. M. Yu. Learning an internal dynamics model from control demonstration. *JMLR workshop and conference proceedings*, pages 606–614, 2013.

[18] E. Archer, I. M. Park, L. Buesing, J. Cunningham, and L. Paninski. Black box variational inference for state space models. *ArXiv e-prints*, November 2015.

[19] R. G. Krishnan, U. Shalit, and D. Sontag. Deep Kalman filters. *arXiv*, abs/1511.05121, November 2015.

[20] M. Johnson, D. K. Duvenaud, A. Wiltschko, R. P. Adams, and S. R. Datta. Composing graphical models with neural networks for structured representations and fast inference. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2946–2954. Curran Associates, Inc., 2016.

[21] R. G. Krishnan, U. Shalit, and D. Sontag. Structured inference networks for nonlinear state space models. *arXiv*, abs/1511.05121, 2016.

[22] M. Karl, M. Soelch, J. Bayer, and P. van der Smagt. Deep variational Bayes filters: Unsupervised learning of state space models from raw data. In *5th International Conference on Learning Representations*, 2017.

[23] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2746–2754. Curran Associates, Inc., 2015.

[24] T. Broderick, N. Boyd, A. Wibisono, A. C. Wilson, and M. I. Jordan. Streaming variational bayes. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 1727–1735. Curran Associates, Inc., 2013.

[25] E. A. Wan and R. Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, Lake Louise, Alta., Canada, August 2000. IEEE.

[26] E. A. Wan and A. T. Nelson. *Dual extended Kalman filter methods*, pages 123–173. John Wiley & Sons, Inc, 2001.

[27] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, August 1991.

[28] G. Hinton, P. Dayan, B. Frey, and R. Neal. The "wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, May 1995.

[29] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer-Verlag GmbH, 2009.

[30] D. Sussillo, R. Jozefowicz, L. F. Abbott, and C. Pandarinath. LFADS - latent factor analysis via dynamical systems. *arXiv*, abs/1608.06315, August 2016.

[31] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, May 2014.

[32] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*, May 2014. arXiv: 1312.6114.

[33] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[34] J. P. Newman, M.-f. Fong, D. C. Millard, et al. Optogenetic feedback control of neural activity. *eLife*, 2015.

[35] A. El Hady. *Closed Loop Neuroscience*. Academic Press, London, United Kingdom, 2016.

[36] D. Hocker and I. M. Park. Myopic control of neural dynamics. *PLOS Computational Biology*, 2019.

[37] J. H. Macke, L. Buesing, J. P. Cunningham, et al. Empirical models of spiking in neural populations. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1350–1358. Curran Associates, Inc., 2011.

[38] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, April 1982.

[39] P. Dayan and L. F. Abbott. *Theoretical neuroscience: computational and mathematical modeling of neural systems*. Massachusetts Institute of Technology Press, 2001.

[40] O. Barak, D. Sussillo, R. Romo, M. Tsodyks, and L. F. Abbott. From fixed points to chaos: three models of delayed discrimination. *Progress in neurobiology*, 103:214–222, April 2013.

[41] X.-J. Wang. Probabilistic decision making by slow reverberation in cortical circuits. *Neuron*, 36(5):955–968, December 2002.

[42] J. Nassar, S. W. Linderman, M. Bugallo, and I. M. Park. Tree-structured recurrent switching linear dynamical systems for multi-scale modeling. In *International Conference on Learning Representations (ICLR)*, November 2019.

[43] J. W. Pillow, J. Shlens, L. Paninski, et al. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995–999, July 2008.

[44] W. Truccolo, U. T. Eden, M. R. Fellows, J. P. Donoghue, and E. N. Brown. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology*, 93(2):1074–1089, February 2005.

[45] S. H. Strogatz. *Nonlinear Dynamics and Chaos*. Studies in nonlinearity. The Perseus Books Group, January 2000.

[46] A. Peyrache, M. M. Lacroix, P. C. Petersen, and G. Buzsaki. Internally organized mechanisms of the head direction sense. *Nature Neuroscience*, 18(4):569–575, March 2015.

[47] E. M. Izhikevich. *Dynamical systems in neuroscience : the geometry of excitability and bursting*. Computational neuroscience. MIT Press, 2007.

[48] C. Curto, S. Sakata, S. Marguet, V. Itskov, and K. D. Harris. A simple model of cortical dynamics explains variability and state dependence of sensory responses in Urethane-Anesthetized auditory cortex. *The Journal of Neuroscience*, 29(34):10600–10612, August 2009.

[49] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F Radar and Signal Processing*, 140(2):107, 1993.

[50] C. Pandarinath, D. J. O'Shea, J. Collins, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, 15(10):805–815, sep 2018.

[51] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. 2014.

[52] S. Ganguli, J. W. Bisley, J. D. Roitman, et al. One-dimensional dynamics of attention and decision making in LIP. *Neuron*, 58(1):15–25, April 2008.

[53] K.-F. Wong and X.-J. Wang. A recurrent network mechanism of time integration in perceptual decisions. *The Journal of Neuroscience*, 26(4):1314–1328, January 2006.

[54] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14:2531–2560, 2002.

[55] R. Laje and D. V. Buonomano. Robust timing and motor patterns by taming chaos in recurrent neural networks. *Nature Neuroscience*, 16(7):925–933, May 2013.

[56] A. B. A. Graf, A. Kohn, M. Jazayeri, and J. A. Movshon. Decoding the activity of neuronal populations in macaque primary visual cortex. *Nature Neuroscience*, (2):239–245, January 2011.

[57] I. D. Jordan and I. M. Park. Birhythmic analog circuit maze: A nonlinear neurostimulation testbed. *Entropy*, 22(5):537, May 2020.

[58] T. Katayama. *Subspace methods for system identification*. Springer, 2005.

[59] B. O. Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences of the United States of America*, 17(5):315–318, May 1931.

[60] B. W. Brunton, L. A. Johnson, J. G. Ojemann, and J. N. Kutz. Extracting spatial–temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition. *Journal of Neuroscience Methods*, 258:1–15, jan 2016.

[61] W. Liu, I. Park, and J. C. Principe. An information theoretic approach of designing sparse kernel adaptive filters. *IEEE Transactions on Neural Networks*, 20(12):1950–1961, dec 2009.