PLOS COMPUTATIONAL BIOLOGY

RESEARCH ARTICLE

Poisson balanced spiking networks

Camille E. Rullán Buxó[1], **Jonathan W. Pillow**[2]*

1 Center for Neural Science, New York University, New York City, New York, USA, **2** Princeton Neuroscience Institute, Princeton University, Princeton, New Jersey, USA

* pillow@princeton.edu

Abstract

An important problem in computational neuroscience is to understand how networks of spiking neurons can carry out various computations underlying behavior. Balanced spiking networks (BSNs) provide a powerful framework for implementing arbitrary linear dynamical systems in networks of integrate-and-fire neurons. However, the classic BSN model requires near-instantaneous transmission of spikes between neurons, which is biologically implausible. Introducing realistic synaptic delays leads to an pathological regime known as "ping-ponging", in which different populations spike maximally in alternating time bins, causing network output to overshoot the target solution. Here we document this phenomenon and provide a novel solution: we show that a network can have realistic synaptic delays while maintaining accuracy and stability if neurons are endowed with conditionally Poisson firing. Formally, we propose two alternate formulations of Poisson balanced spiking networks: (1) a "local" framework, which replaces the hard integrate-and-fire spiking rule within each neuron by a "soft" threshold function, such that firing probability grows as a smooth nonlinear function of membrane potential; and (2) a "population" framework, which reformulates the BSN objective function in terms of expected spike counts over the entire population. We show that both approaches offer improved robustness, allowing for accurate implementation of network dynamics with realistic synaptic delays between neurons. Both Poisson frameworks preserve the coding accuracy and robustness to neuron loss of the original model and, moreover, produce positive correlations between similarly tuned neurons, a feature of real neural populations that is not found in the deterministic BSN. This work unifies balanced spiking networks with Poisson generalized linear models and suggests several promising avenues for future research.

Author summary

A central idea in neuroscience is that populations of neurons work together to efficiently perform computations, although just how they do that remains unclear. Boerlin *et al* (2013) proposed a powerful framework for embedding linear dynamical systems into populations of spiking neurons, which they called balanced spiking networks (BSNs). Their approach starts from the principle that neurons greedily fire spikes to reduce error in the network output. Here we focus on a key limitation of this framework, which is that the network may become unbalanced in the presence of physiologically plausible

OPEN ACCESS

Citation: Rullán Buxó CE, Pillow JW (2020) Poisson balanced spiking networks. PLoS Comput Biol 16(11): e1008261. https://doi.org/10.1371/journal.pcbi.1008261

Editor: Daniele Marinazzo, Ghent University, BELGIUM

Received: November 11, 2019

Accepted: August 14, 2020

Published: November 20, 2020

Data Availability Statement: All code used to generate the figures in this paper can be found on https://github.com/pillowlab/PoissonBalancedNets.

Funding: CERB was supported by the NSF GRFP (DGE1839302). JWP was supported by grants from the McKnight Foundation, NSF CAREER Award (IIS-1150186), the Simons Collaboration on the Global Brain (SCGB AWD543027), the NIH BRAIN initiative (R01EB026946), and a U19 NIH-NINDS BRAIN Initiative Award (5U19NS104648). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

communication delays. To overcome this shortcoming, propose two different extensions of the BSN framework that rely on probabilistic spiking. In our first model, we replace deterministic spiking of the original BSN with a Poisson spiking rule. In the second, we re-formulate the BSN objective so that Poisson spiking emerges as a way to reduce the expected network error. Our work brings the BSN framework closer to biological realism by increasing the stability and, most importantly, allowing communication delays between neurons without sacrificing accuracy. Furthermore, both probabilistic approaches reproduce key experimentally observed spiking behaviors of neural populations.

Introduction

The brain carries out a wide variety of computations that can be implemented by dynamical systems, from sensory integration [1–4], to working memory [5–7], to movement planning and execution [8–10]. Although the existence of such computations in the brain is well established, the mechanisms by which these computations are implemented in networks of neurons remains poorly understood. One approach to this problem involves statistical modeling, which uses descriptive statistical methods to infer the dynamics of neural activity from recorded spike trains [10–21]. A second approach involves theoretical modeling, which seeks to identify strategies for implementing dynamical systems with networks of idealized model neurons [2, 22–30]. An important example of this second approach is the *balanced spiking network* (BSN) framework introduced by Boerlin *et al* [31].

The BSN model consists of a network of coupled leaky integrate-and-fire (LIF) neurons that can emulate an arbitrary linear dynamical system (LDS). The motivating idea is to design a network that approximates the output of a target LDS with a weighted combination of filtered spike trains. The population is divided into "excitatory" and "inhibitory" populations of neurons, based on whether they contribute positively or negatively to the output. This leads to an intuitive spiking rule: a neuron should spike whenever doing so will reduce the error between the output of the target LDS and the network output, i.e., the weighted combination of filtered spikes emitted so far. To make this work, each neuron has to maintain an internal representation of the error between the desired LDS output and the current network output. Boerlin *et al* showed that, remarkably, this computation can be mapped precisely onto the dynamics of an LIF neuron. A neuron's membrane potential is a local representation of the network-wide error between target output and current network output, and its spike threshold is proportional to the amount by which adding a spike will reduce this error.

The BSN framework has many appealing characteristics. Spiking is efficient, in the sense that every spike contributes meaningfully to reducing error between target and actual output. The computations performed by the model are robust to perturbations and to the loss of neurons. The model also generates irregular spiking activity with intervals that that resemble those observed in real neurons.

However, the original BSN model has an important shortcoming that limits its plausibility as a model for information processing in real neural circuits. Namely, the model requires unrealistically fast propagation of information between neurons. Because every neuron's membrane potential is tracking the overall error between target and actual output, the membrane potential of *all* neurons has to reset whenever any neuron emits a spike. Failure to impose this reset leads to increased activity as multiple neurons attempt to correct same error. In fact, implementations of the BSN model typically impose a rule enforcing that only one neuron is

allowed to spike and required to immediately reset in a single time bin, effectively allowing spikes to propagate faster than the temporal resolution of the simulation (e.g., 0.1ms). Without this rule, the network can easily enter unstable modes in which excitatory and inhibitory populations emit massive spike bursts in alternating time bins, overshooting the target in an attempt to correct the error from the previous time bin.

Here we show that a probabilistic spiking rule can overcome the need for unrealistically fast propagation of spikes in the BSN framework. The basic intuition for our solution is that instead of making neurons spike deterministically whenever doing so will reduce error, we can allow multiple neurons to spike probabilistically such that error will be reduced on average.

We propose two alternate formulations of BSN with Poisson spiking, distinguished from each other by the level at which the network is attempting to minimize the decoding error. First, we describe a 'local' framework, which preserves the original BSN model dynamics but replaces the hard integrate-and-fire spiking rule with the soft firing threshold of the Poisson generalized linear model (GLM) [32–35]. This spiking rule generates stochastic spiking conditioned only on each neuron's local copy of the error which, on average, leads to a reduction in the population-level read-out error.

Second, we propose a 'population' framework that replaces the greedy, single-neuron perspective of the local and BSN models with a rule based on minimizing the expected error at the population level. A vector of spike rates is generated by calculating the expected spike counts that minimizes the total decoding error. The probability of a single neuron spiking depends on its own weight, as with the local rule, but also takes into account the activity of the entire population of neurons and their weights. This coordination leads to spiking activity that is efficient and invariant to network size. Then, we show that both the local- and population-level Poisson frameworks make the BSN robust to synaptic delays. Finally, we compare all three frameworks and show that our local and population frameworks preserve the coding accuracy and robustness to neuron loss of the original BSN, while displaying more biologically plausible correlations between similarly tuned neurons.

This paper is organized as follows. We begin with a pedagogical review to the BSN model (Sec. 2). We then examine the model's dependence on instantaneous spike propagation, and document the unstable behavior that arises if multiple spikes are allowed in a single time bin (Sec. 3). To address this shortcoming, we introduce local and population BSN models with conditionally Poisson spiking (Sec. 4). We go on to illustrate the accuracy and robustness of these models to synaptic transmission delays (Sec. 5). Finally, we compare the network performance of these models to that of the original BSN (Sec. 6).

Results

Background: Balanced spiking network model

Here we provide a brief introduction to the original balanced spiking network (BSN) framework introduced by Boerlin, Machens, & Denève [31]. The goal is to design a spiking network that can accurately implement an arbitrary linear dynamical system. Consider a linear dynamical system defined by:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + \mathbf{c}(t), \qquad (target\ dynamics) \tag{1}$$

where $\mathbf{x}(t) = (x_1(t)\ldots x_J(t))^\top$ is a vector of J dynamic variables that we will refer to as the *target*, A is the $J \times J$ linear dynamics matrix, and $\mathbf{c}_t = (c_i(t), \ldots c_J(t))^\top$ is a J-dimensional vector of inputs. The BSN model consists of a spiking network of N neurons that attempts to

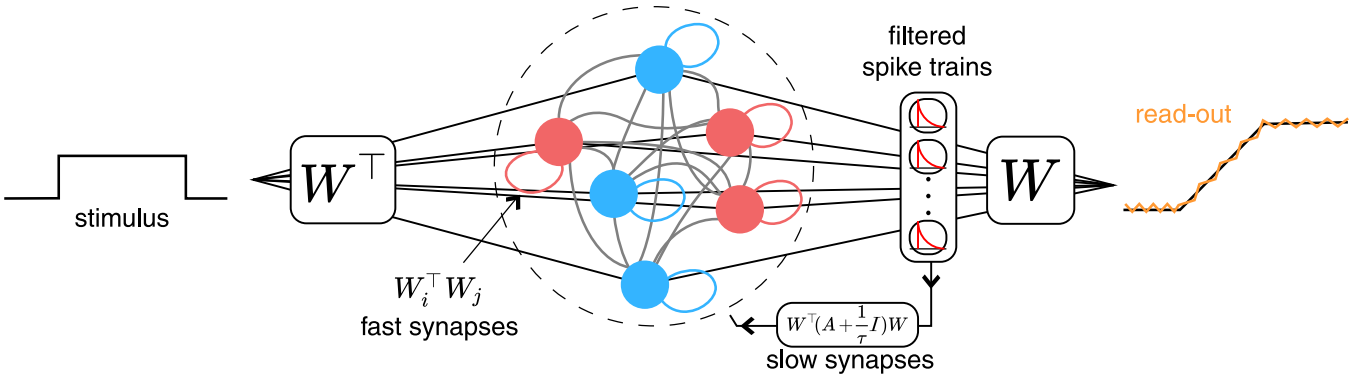

Fig 1. A diagram illustrating the BSN. Neurons receive stimulus input projected onto the transpose of a set of linear weights, $W^\top$, and the output is reconstructed by filtering spikes through the same weights, W. Neurons are connected via two coupling weights: fast synapses, $W^\top W$, which instantaneously propagate individual spikes through the network, and slow synapses, $W^\top\left(A+\frac{1}{\tau}\right)W$, which implement network dynamics by feeding the filtered spike trains back into all neurons in the network. The network is divided into two equal populations of positive (red) and negative (blue) output weights, whose spikes have opposite effects on network output. Self-connections for these neurons are shaded in their respective colors, for visualization, but are always negative.

https://doi.org/10.1371/journal.pcbi.1008261.g001

approximate the target output $\mathbf{x}(t)$ via a weighted combination of filtered spike trains:

$$\hat{\mathbf{x}}(t) = W\mathbf{r}(t) \qquad (network\ readout) \tag{2}$$

where $\mathbf{r}(t)$ is the set of spike trains convolved with an exponential decay function, and W are $J \times N$ readout weights. (See Fig 1 for a schematic.) In general, for a 1-D dynamical system, the population is divided into equal pools of 'positive' and 'negative' neurons (depending on the signs of their individual weight components) although this is not a strict requirement. For $J > 1$, the 'positive' vs 'negative' distinction does not necessarily apply as the signs of the weights need not be consistent across dimensions.

The i'th component of the vector $\mathbf{r}(t)$ is given by

$$r_i(t) = s_i(t) * h(t) = \int_0^t e^{-t'/\tau} s_i(t')dt', \qquad (filtered\ spike\ trains) \tag{3}$$

where $s_i(t) = \sum_{t_{sp}^i} \delta(t - t_{sp}^i)$ denotes the i'th neuron's spike train, defined by a series of delta functions at spike times $\{t_{sp}^i\}$, and τ is the time constant of the exponential filter $h(t)$.

From this starting assumption, Boerlin *et al* introduce a greedy update rule that causes a neuron to spike whenever doing so will reduce the squared error between target $\mathbf{x}(t)$ and network output $\hat{\mathbf{x}}(t)$,

$$E(t) = ||\mathbf{x}(t) - \hat{\mathbf{x}}(t)||_2^2, \qquad (error\ function) \tag{4}$$

which is mathematically equivalent to the threshold-crossing spiking rule in a leaky integrate-and-fire (LIF) neuron.

Here we recapitulate the derivation of this spiking rule in discrete time, for clarity and ease of implementation. Let $\hat{\mathbf{x}}_t = W\mathbf{r}_t$ denote the network output at time bin t. The effect of adding a spike from neuron i in this time bin would be to augment the output vector by that neuron's decoding weight vector $\mathbf{w}_i$, which is given by i'th row of the decoding weight matrix W. Thus, network output is $\hat{\mathbf{x}}_t$ if neuron i is silent and $\hat{\mathbf{x}}_t + \mathbf{w}_i$ if it spikes. This suggests that the neuron

should spike if doing so will result in smaller error, or

$$||\mathbf{x}_t - (\hat{\mathbf{x}}_t + \mathbf{w}_i)||_2^2 < ||\mathbf{x}_t - \hat{\mathbf{x}}_t||_2^2. \tag{5}$$

Simplifying this expression yields the condition that the neuron should spike if projection of the error vector onto $\mathbf{w}_i$ is greater than half the squared L_2 norm of $\mathbf{w}_i$:

$$\mathbf{w}_i^\top(\mathbf{x}_t - \hat{\mathbf{x}}_t) > \tfrac{1}{2}||\mathbf{w}_i||_2^2. \qquad (spike\ condition\ for\ neuron\ i) \tag{6}$$

Boerlin *et al* therefore suggest regarding the time-dependent left hand side of (Eq 6) as the membrane potential for neuron i, and the right hand side as its spike threshold T_i:

$$v_{[i]t} = \mathbf{w}_i^\top(\mathbf{x}_t - \hat{\mathbf{x}}_t) \qquad (membrane\ potential) \tag{7}$$

$$T_i = \tfrac{1}{2}||\mathbf{w}_i||_2^2 \qquad (threshold). \tag{8}$$

Under this view, each neuron is computing a local approximation to the difference between the true target $\mathbf{x}_t$ and the network's current output $\hat{\mathbf{x}}_t = W\mathbf{r}_t$, projected onto that neuron's weight vector $\mathbf{w}_i$.

The only missing piece from this expression is that the neurons do not of course have access to the true value of $\mathbf{x}_t$. But they do have implicit access to A, and thus to the dynamics governing $\mathbf{x}_t$. Boerlin *et al* therefore propose that the the network output $\hat{\mathbf{x}}_t$ is sufficiently close to the target output $\mathbf{x}_t$ that it can be used to accurately approximate the desired dynamics: $A\mathbf{x}_t \approx A\hat{\mathbf{x}}_t$. To make this explicit, we introduce a proxy variable $\mathbf{z}_t$, which denotes the network's own (internal) approximation to the true target $\mathbf{x}_t$. This proxy variable evolves according to

$$\begin{aligned} \mathbf{z}_t &= \mathbf{z}_{t-1} + \Delta(A\hat{\mathbf{x}}_{t-1} + \mathbf{c}_t) \\ &= \mathbf{z}_{t-1} + \Delta(AW\mathbf{r}_{t-1} + \mathbf{c}_t), \quad (proxy\ variable) \end{aligned} \tag{9}$$

where for simplicity we use a forward Euler method for integrating the dynamics equation (Eq 1) with time bin size Δ. Higher accuracy can be achieved using exponential Euler integration (see Methods). Of course $\mathbf{z}_t$ is never represented explicitly; the network tracks $\mathbf{z}_t$ via its projection onto the decoding weights W, as we will see shortly.

Simulating the BSN model. Simulating the BSN model for a single time bin can be described by a sequence of three steps:

1. Calculate the "pre-spike" membrane potential for each neuron by combining inputs from the previous time step and external input.
2. Apply the threshold to determine which neurons (if any) emit spikes.
3. Reset to obtain "post-spike" membrane potentials $\mathbf{v}_t$ and update filtered spike trains $\mathbf{r}_t$.

We will describe each of these steps in turn. First, the update rule for the pre-spike membrane potential (consistent with Eq 7) is:

$$\begin{aligned} v_{[i]t}^{(pre)} &= \mathbf{w}_i^\top(\mathbf{z}_t - \hat{\mathbf{x}}_t^{(pre)}) \\ &= \mathbf{w}_i^\top\left(\mathbf{z}_{t-1} + \Delta(AW\mathbf{r}_{t-1} + \mathbf{c}_t) - \left(1 - \Delta\frac{1}{\tau}\right)W\mathbf{r}_{t-1}\right) \\ &= v_{[i]t-1} + \Delta\mathbf{w}_i^\top\left(\left(A + \frac{1}{\tau}I\right)W\mathbf{r}_{t-1} + \mathbf{c}_t\right) \quad (pre\text{-}spike\ membrane\ potential) \end{aligned} \tag{10}$$

where $v_{[i]t}^{(pre)}$ denotes the pre-spike membrane potential for neuron i at time bin t,

$\hat{\mathbf{x}}_t^{(pre)} = \left(1 - \Delta\frac{1}{\tau}\right) W\mathbf{r}_{t-1}$ denotes the network output for the current time bin *before spiking*, and $v_{[i]t-1} = \mathbf{w}_i{}^\top(\mathbf{z}_{t-1} - W\,\mathbf{r}_{t-1})$ denotes the (post-spike) membrane potential from the previous time step.

Second, spikes for the current time bin are computed by determining whether pre-spike membrane potential exceeds threshold ($v_{[i]t}^{(pre)} > T_i$). When this occurs, the neuron records a spike: $s_{[i]t} = 1$.

Lastly, the filtered spike trains $r_{[i]t}$ are augmented and membrane potential is reset:

$$r_{[i]t} = \left(1 - \Delta\tfrac{1}{\tau}\right) r_{[i]t-1} + s_{[i]t} \qquad (\textit{filtered spike train update}) \tag{11}$$

$$v_{[i]t} = \mathbf{w}_i{}^\top(\mathbf{z}_t - \hat{\mathbf{x}}_t) = v_{[i]t}^{(pre)} - \mathbf{w}_i{}^\top W\mathbf{s}_t, \qquad (\textit{post-spike membrane potential}) \tag{12}$$

which ensures that post-spike membrane potential equals the difference between the projected proxy variable and network output.

Vector update rules. For convenience, we can rewrite the BSN update equations in vector form. The pre-spike membrane potential is given by:

$$\begin{aligned} \mathbf{v}_t^{(pre)} &= W^\top(\mathbf{z}_t - \hat{\mathbf{x}}_t^{(pre)}) \\ &= \mathbf{v}_{t-1} + \Delta W^\top\left(A + \tfrac{1}{\tau}I\right) W\mathbf{r}_{t-1} + \Delta W^\top\mathbf{c}_t \\ &= \mathbf{v}_{t-1} + \Delta(\Omega\mathbf{r}_{t-1} + W^\top\mathbf{c}_t) \qquad (\textit{vector pre-spike membrane potential}) \end{aligned} \tag{13}$$

where $\Omega = W^\top\left(A + \frac{1}{\tau}\right)W$ are the coupling weights from $\mathbf{r}_{t-1}$ to the pre-spike membrane potential, which implement computation of the divergence between the target $\mathbf{z}_t$ and passive decay of $\mathbf{r}_t$ in the absence of spiking.

Once the spike vector $\mathbf{s}_t$ has been computed, the filtered spike trains and network output for the current time bin are given by:

$$\mathbf{r}_t = \left(1 - \Delta\tfrac{1}{\tau}\right)\mathbf{r}_{t-1} + \mathbf{s}_t \qquad (\textit{vector of filtered spike trains}) \tag{14}$$

$$\hat{\mathbf{x}}_t = W\mathbf{r}_t, \qquad (\textit{vector network output}) \tag{15}$$

and the vector of post-spike membrane potentials is given by

$$\begin{aligned} \mathbf{v}_t &= W^\top(\mathbf{z}_t - \hat{\mathbf{x}}_t) \\ &= \mathbf{v}_t^{(pre)} - W^\top W\mathbf{s}_t \\ &= \mathbf{v}_{t-1} + \Delta(\Omega\mathbf{r}_{t-1} + W^\top\mathbf{c}_t) - W^\top W\mathbf{s}_t, \qquad (\textit{vector post-spike membrane potential}) \end{aligned} \tag{16}$$

which reflects reset of the pre-spike membrane potential after spiking, but can equivalently be seen to be the projected difference between the proxy variable $\mathbf{z}_t$ and the current network output $\hat{\mathbf{x}}_t$ in the current time bin.

It is worth noting that this model requires the instantaneous propagation of spikes between neurons. After a spike, the membrane potential reset (Eq 16) updates $\mathbf{v}_t$ for all neurons based on the spikes in the current time bin via the fast weights, $-W^\top W$. Although Boerlin *et al* refer to the weights $-W^\top W$ as "fast synapses" and the Ω as "slow synapses", note that the $\Omega\,\mathbf{r}_{t-1}$ term also involves near-instantaneous propagation of information, since the exponentially-filtered spike trains $\mathbf{r}$ jump by 1 after every spike.

The full BSN model first described in [31] contained additional penalties on $\mathbf{r}_t$ in the objective function, which had the effect of reducing spiking by trading off minimization of error (Eq 4) against a cost of inserting spikes. Although we have left these terms out our derivation

here for simplicity, including them has the limited effect of changing the spike threshold and post-spike reset and does not change the nature of our findings. The simulations of the original BSN shown in the following sections use the full version as described in the Methods section. Simulation parameters for each figure are also included in the Methods section.

Limitations of the BSN model

A key limitation of the BSN model is that it requires unrealistically fast communication between neurons. In the standard integrate-and-fire model, a spike resets only the membrane potential of the neuron that emitted it. In the BSN model, by contrast, the membrane potentials of *all* neurons reset following a spike from *any* neuron via the $-W^\top W\,\mathbf{s}_t$ term in (Eq 16). The instantaneous reset following a spike is necessary to ensure that each neuron's membrane potential maintains an accurate representation of the read-out after each spike. From a normative standpoint, the hard LIF threshold entails that maintaining an accurate local copy of the error is critical for the network to encode the target.

In fact, the problem is slightly worse than this: standard implementations of the BSN model include a constraint that only one neuron is allowed to spike in a single time bin as a way of imposing instantaneous (i.e., sub-time bin) communication. Without this constraint, neurons with similar output weights tend to spike in the same time bin, when a spike from any one of them would have sufficed to compensate for error in network output. This causes the network output to dramatically overshoot the target. In the subsequent time bin, neurons with opposite-sign weights fire to compensate for this error, and overshoot the target by a large amount in the opposite direction. This sets up a pathological pattern of oscillatory firing known as "ping-ponging", in which two populations spike maximally in alternating time bins of the discrete simulation [31, 36].

Fig 2 illustrates how ping-pong behavior can arise if multiple spikes are allowed in a single time bin. We set the BSN model to implement a 1-dimensional exact integrator, $\dot{x}(t) = c(t)$, using the same parameters as the example from figure 1C of [31]. In brief, the network contained 400 neurons, divided into two equal sized populations with output weights of +0.1 and -0.1, respectively, which we refer to as positive-output and negative-output neurons (see Methods for complete details). When the rule forbidding multiple spikes per time bin is imposed, the network accurately tracks the target output variable (Fig 2A). However, removing this constraint—allowing all neurons with membrane potential above threshold to fire—results in ping-pong behavior and large errors in tracking the target (Fig 2B).

Note that ping-ponging could be eliminated by computing threshold-crossing times with extremely high temporal precision, either using very small time bins or spike-time interpolation methods [37, 38]. Such precision may allow us to identify the first neuron to spike and to near-instantaneously reset the membrane potential of other neurons, preventing overshoot of the target. However, this near-instantaneous reset of the membrane potential in other neurons is at odds with the timescale of synaptic communication, and represents a shortcoming of the original BSN framework that our work seeks to overcome. (See Discussion for an overview of other approaches to the problem of instantaneous synapses, e.g. [36]). In the following sections, we imposed the constraint that only one neuron could spike per time bin for all simulations of the original BSN model, consistent with [31].

BSN with conditionally Poisson neurons

To overcome the problems of instantaneous transmission and network instability, we propose two novel formulations of balanced spiking networks with conditionally Poisson neurons: (1) a local framework, where each neuron spikes independently based on its local estimate of

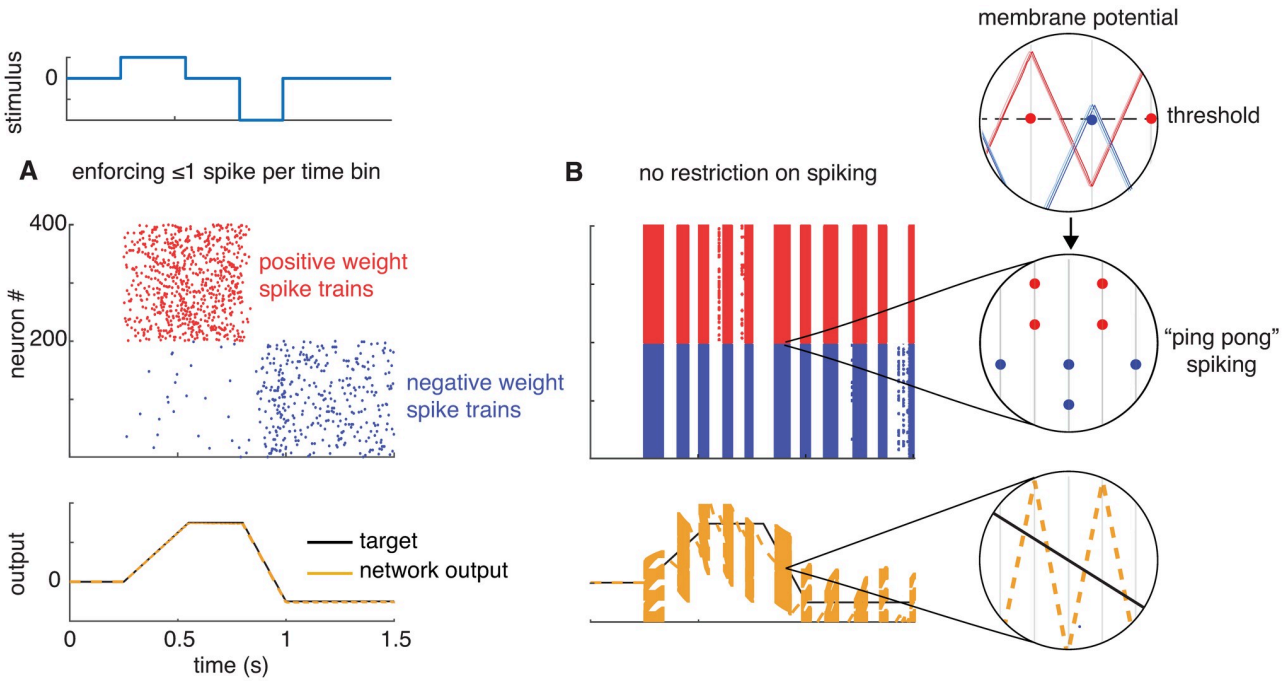

Fig 2. Balanced spiking network implementing an exact integrator. The network consists of 400 neurons, divided into two populations with output weights of + 0.1 (red) and −0.1 (blue). **(A)** Simulation results under the condition that only one neuron is allowed to fire per discrete time bin. **(B)** Simulation results when all neurons whose membrane potential is above threshold in a single time bin are allowed to fire, leading to "ping-pong" behavior. Insets show that the read-out (yellow) is alternating between large over- and under-estimates of the target (in black). Insets show, in order from top to bottom: the voltage traces of neurons in both positively and negatively weighted populations for a small time window, the resulting spikes in each time bin, and the resulting read-out (yellow) and target (black). Since the weights and inputs are identical across populations, so are the voltage traces. Ping-ponging results, as all neurons within a population cross the threshold in the same time bin, spike, and cause the read-out to oscillate between over- and under-estimates of the target.

https://doi.org/10.1371/journal.pcbi.1008261.g002

network error; and (2) a population-level framework, which sets firing rates to reduce expected error for the entire population.

The key idea in both frameworks is to replace the deterministic integrate-and-fire spiking rule with probabilistic spiking. Under this modified spiking rule, spiking is governed by an instantaneous probability of spiking λ_t, also known as the *conditional intensity*, such that spiking is independent with probability $\Delta\lambda_t$ in any small time window of width Δ. This results in an auto-regressive Poisson generalized linear model (GLM), also known as a Cox process [33–35]. This model has a quasi-realistic biophysical interpretation [32, 39, 40], and recent work has shown that it can capture a wide range of dynamical behaviors found in real neurons [35].

Local framework. A simple way to introduce probabilistic spiking to the BSN framework is to replace the hard spike threshold of the integrate-and-fire model with a soft threshold, so that spike probability grows as a nonlinear function of membrane potential, an approach also known as the "escape-rate approximation" [32]. Specifically, we define each neuron's conditional intensity function to be a sigmoidal function of membrane potential:

$$\lambda_t = f(v_t) = \frac{F_{max} - F_{min}}{1 + e^{-\alpha(v_t - T)}} + F_{min}, \qquad (nonlinearity) \tag{17}$$

where v_t is the membrane potential at time t, T is the spike threshold, α is a slope parameter governing the sharpness of the threshold, F_{max} is the maximal firing rate and F_{min} is a baseline firing rate, meant to simulate random firing activity in the absence of a stimulus. The probability of spiking in a small time interval is proportional to λ_t, which models the spike response as

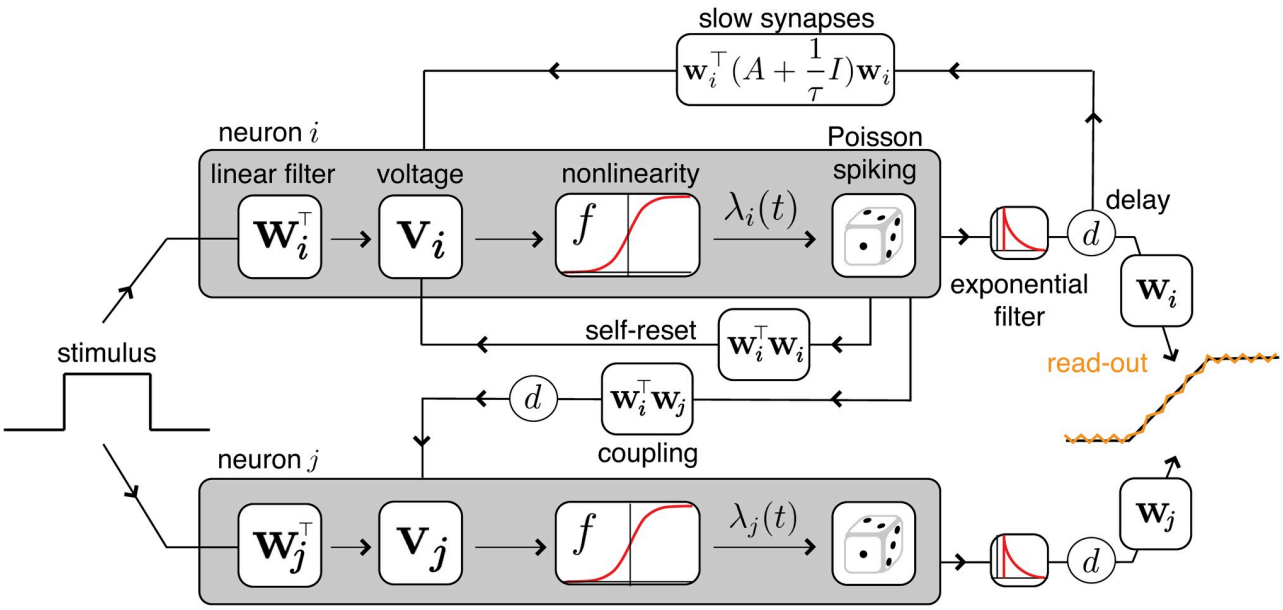

Fig 3. Schematic of two neurons in a BSN with conditionally Poisson neurons. The stimulus influences each neuron's membrane potential $\mathbf{v}_i$ via a set of input weights $W^\top$. The neurons reset themselves via instantaneous, fast synapses. Fast connections to other neurons propagate the effects of spikes with a synaptic time delay d. The desired linear dynamics are implemented via slow weights (through spike trains filtered by an exponential) also with a time delay d. Within each neuron, spiking is probabilistic with an instantaneous probability of firing $\lambda_i(t) = f(v_i(t))$, where $f(\cdot)$ is a nonlinear function of voltage. Self-connections are only shown for neuron i.

https://doi.org/10.1371/journal.pcbi.1008261.g003

an inhomogenous Poisson process. We refer to this as the local Poisson framework because, like the BSN, spikes are generated by internal dynamics that evolve according to local copies of the representation error. See Fig 3 for a schematic.

Although it is common to use exponential nonlinearities for Poisson GLMs, here we have used a scaled sigmoid function to control both the suddenness of firing onset and the maximum achievable firing rate. The parameter α controls the precision of firing onset, while F_{max} and F_{min} control the range of firing rates within a small time window. The resulting function (Eq 17) resembles an exponential function at low firing rates but saturates at a maximum of F_{max} (see Fig 4). If we let both α, $F_{max} \to \infty$, we recover the (hard-threshold) integrate-and-fire rule of the original BSN, in which a spike occurs probability 1 when $v_t > T$. However, for finite α and F_{max}, the onset of spiking is more gradual.

As a practical matter, we do not wish to allow a single neuron to fire multiple spikes in a single time bin, because the first spike would preclude additional spikes in the same time bin due to "reset" of the membrane potential. We therefore simulate the model with the spiking rule:

$$P(s_t = 1|\lambda_t) = 1 - \exp(-\Delta\lambda_t), \qquad (local\ Poisson\ framework\ firing\ rule) \qquad (18)$$

where $\exp(-\Delta\lambda_t)$ is the probability of observing no spikes in a time bin of size Δ under the Poisson model. For each time-step, we update $\mathbf{v}$ as in the original BSN model, pass it through the nonlinear function $f(\cdot)$ to obtain the vector of Poisson firing rates, λ_t, and draw spikes as independent Bernoulli random variables with probability as given above.

Fig 4 illustrates how α affects spiking precision by comparing spike times of a single neuron integrating a noisy stimulus implementing the original BSN framework (Fig 4D) to the local framework with different values of α (Fig 4E). As we expect, for high values of α we recover the precise spiking behavior of the BSN model. As α decreases, spike times spread around the ideal BSN spikes. Note that the precise amount of spikes fired in the time window (four) is

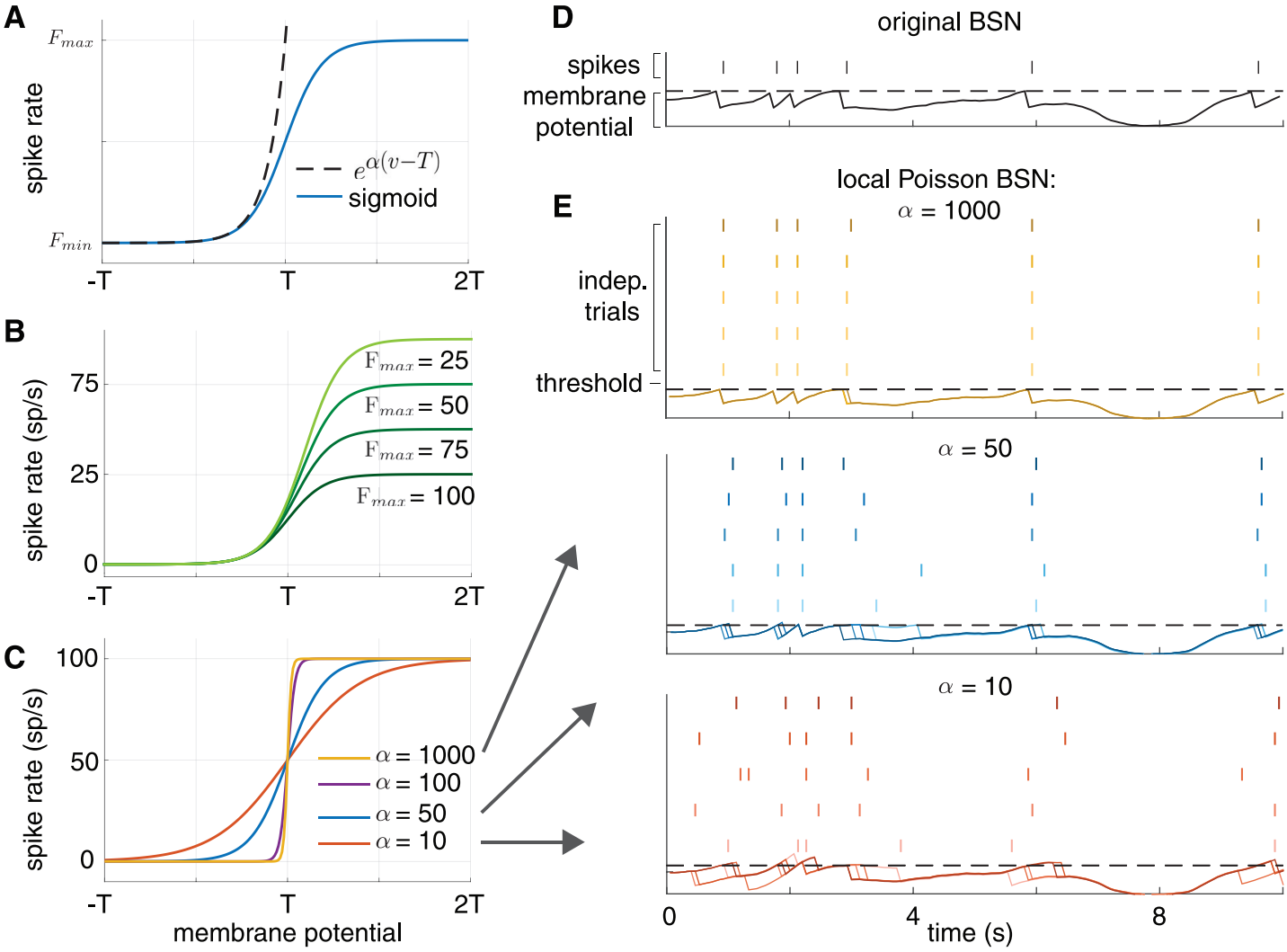

Fig 4. **(A)** The conditional intensity for the exponential non-linearity (dashed lines) and the sigmoid non-linearity (solid lines). The conditional intensity of the sigmoidal non-linearity closely follows that of the exponential non-linearity for sub-threshold voltages, but levels off after threshold, keeping firing rates stable. **(B)** Family of nonlinearities with varying F_{max}. Increasing F_{max} raises the firing rate at which the nonlinearity saturates. **(C)** Family of nonlinearities with varying α. Increasing α increases the steepness of the nonlinearity, which approaches a hard-threshold function as $\alpha \to \infty$ (like the BSN). **(D)** Simulation of the original BSN implementing an exact integrator, showing membrane potential and spikes of a single example neuron. **(E)** Spikes and membrane potential of the same neuron in a local Poisson BSN implementation of the same system. High α simulations (yellow) replicate the behavior of the BSN integrator. Lowering α to 50 (blue) or 10 (red) results in a spread of spikes centered around the deterministic BSN spikes.

https://doi.org/10.1371/journal.pcbi.1008261.g004

conserved, although the timing is highly variable. On average, though, the local framework has spike times centered around those of the original BSN.

Robustness to parameters of the nonlinearity. We studied the effects of varying α, F_{max}, and F_{min} on the performance of the homogeneous integrator network (Fig 5) and found that there exists a wide range of values for which the network error is low and the spiking activity is efficient (Fig 5B and 5C). The rightmost panel on Fig 5A shows raster plots and corresponding read-outs for 'optimal' parameter settings (defined as being in the low error and activity range, denoted by the red * in Fig 5B and 5C), and in the subsequent panels we modify each of the parameters in turn to show qualitative changes in spiking activity and read-out accuracy. Decreasing α and F_{max} negatively impacts read-out quality, while removing background spiking returns the network to a high-precision, synchronized regime.

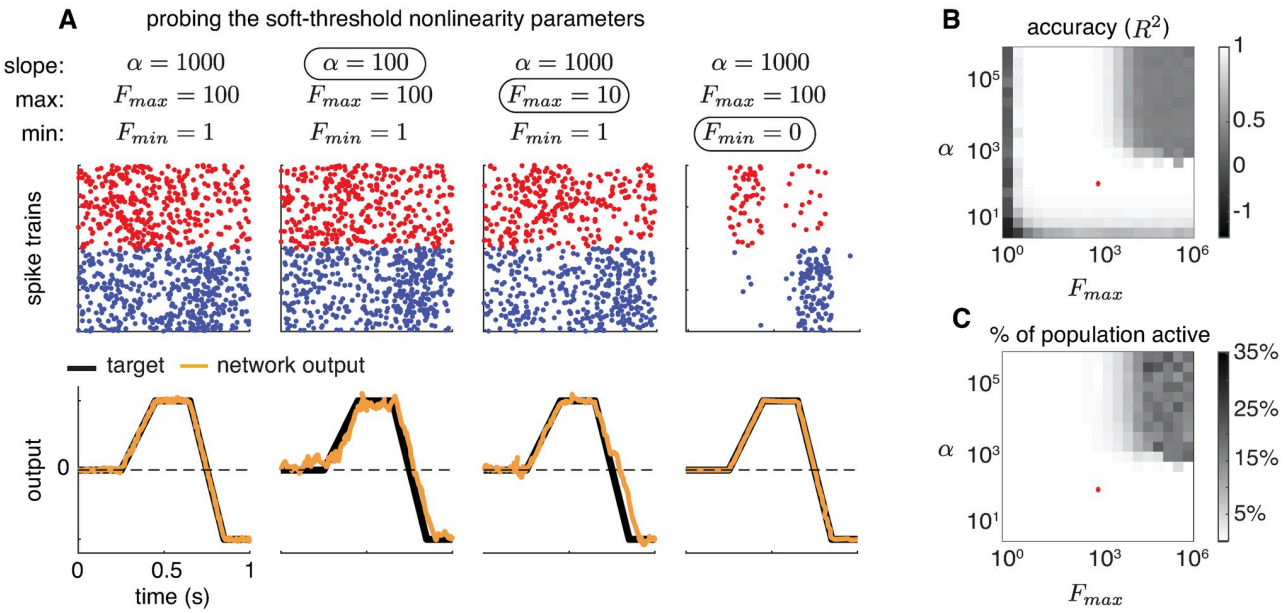

Fig 5. **(A)** Simulations of local Poisson model showing the effects of varying the parameters of the soft-threshold nonlinearity on performance. Relevant parameters are the slope α, maximal firing rate F_{max}, and baseline firing rate F_{min}. Network dynamics implemented an exact 1D integrator and the stimulus was the same as Fig 2. Red and blue dots indicate spikes from neurons with positive and negative output weights, respectively. **(B)** Network performance as quantified by R^2 across a range of parameter settings with baseline fixed at $F_{min} = 0$ (log-scale). Red asterisk indicates the values for the rightmost column of A ($\alpha = 1000$, $F_{max} = 100$). Accuracy remains high across a broad range of parameter values, falling substantially below 1 when slope and maximum firing rate are both large or very low. **(C)** Percent of the neural population active as a function of α and F_{max}, with F_m *in* $= 0$. The network shows ping-ponging behavior in upper right corner, where the model approaches a deterministic, hard-threshold firing rule.

https://doi.org/10.1371/journal.pcbi.1008261.g005

As shown in figure (Fig 5B), network performance noticeably deteriorates for very large values of α and F_{max}. This happens because we are forcing the precision to be too high through α while allowing neurons to spike too frequently through F_{max}. Since the spiking rule is localized, large proportions of the population are active at the same time (Fig 5C), much what happens when the original BSN starts to ping-pong. However, the values at which this happens (e.g., $F_{max} > 10^3$ spikes/second) are well above what we would expect to see in a biological system. Likewise, we observe a low R^2 value when α and F_max are very low. In this case, the probability of spiking is simply too small for spikes to be fired in a given time bin, despite large coding errors. Although these results are using relatively simple target dynamics, we still observe a wide ranges of stable F_{max} and α settings for more complex or multi-dimensional simulations.

The computational advantage of our probabilistic spiking rule is that it introduces uncertainty into spike timing, therefore preventing all neurons from firing at once. The parameters controlling this asynchrony, F_{max}, F_{min} and α, have direct physical interpretations (maximal and minimal/background firing rate and error tolerance, respectively) which can be mapped on to characteristics of real neural circuits.

In theory, a BSN network with voltages corrupted by additive Gaussian noise could behave similarly to the local framework. However, we found that this model still tended to exhibit ping-ponging unless noise amplitude approached the amplitude of signal-induced fluctuations in membrane potential, which adversely affected coding accuracy. By contrast, the local framework is stable even when the resets of other neurons is delayed to the next time bin (see Figs 5 and 6) or by several milliseconds (Fig 7). Intuitively, the reason for this is that, in addition to the slope parameter alpha, which controls the steepness of the soft-threshold-crossing process (in a way, similar to additive voltage noise in the original BSN), we have a limit on the maximal firing rate F_{max}, which ensures that there will not be too many spikes across the population

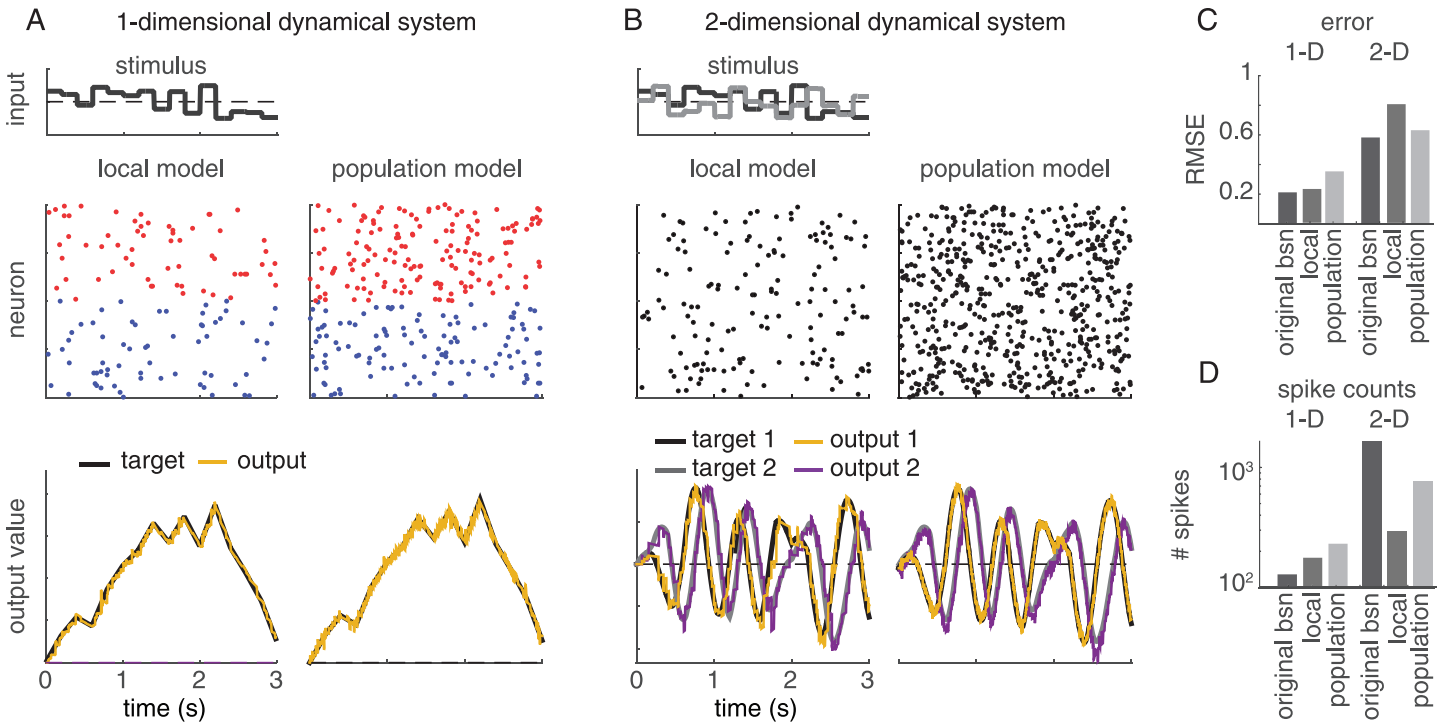

Fig 6. Simulations of the local and population frameworks implementing a 1D and 2D dynamical system. (**A**) The target was a 1-dimensional integrator: $\dot{x}(t) = c(t)$. Left side shows spikes and outputs from local Poisson model, while right side shows spikes and outputs for population Poisson model. As in previous figures, red dots indicate spikes from neurons with positive output weights, blue dots indicate spikes from neurons with negative weights. (**B**) The target was a 2-dimensional oscillator $\dot{x}_1(t) = -x_1(t) - 10x_2(t) + c(t); \dot{x}_2(t) = 10x_1(t) - x_2(t) + c(t)$. For the population model, the time window for computing expected spike count was κ = 5ms (50 time bins). Weights were randomized to be positive or negative in either dimension, such that neurons are no longer divided into strictly positive- or negative-weight groups. **(C)** Accuracy (measured by root-mean-squared error) of the two models for 1D and 2D systems. (**D**) Number of spikes emitted by each model during simulations (log scale).

https://doi.org/10.1371/journal.pcbi.1008261.g006

even when many neurons are driven simultaneously across their (noisy) threshold. That is, it would take several time bins for them all to fire given the value of F_{max}, which is enough time for synaptic inhibition to arrive and prevent firing. Moreover, in Fig 5B we show that there is a broad range of parameters over which the local population is able to achieve stable, accurate coding. Strictly speaking, by introducing extra degrees of freedom, we are losing the strict normative angle of the original BSN framework. However, what we gain in the process—a considerably expanded space of stable network configurations—makes it possible to introduce realistic communication delays between neurons.

Population framework

We now describe a second framework for implementing BSNs with conditionally Poisson neurons. In this approach, we take a population-level instead of a neuron-level view of the optimization problem to be solved. Instead of assigning each neuron to carry an independent representation of the error between the target and actual network output, we assign each neuron an analog probability of firing such that expected number of spikes across the network compensates appropriately for the total error. We refer to this as the population framework.

The derivation of this framework starts from the same foundation of the original BSN, namely, an error function describing the discrepancy between target and actual network output. However, instead of specifying that each neuron should spike whenever doing so will reduce error (Eq 5), we compute a vector of spike rates λ, such that the expected spike response

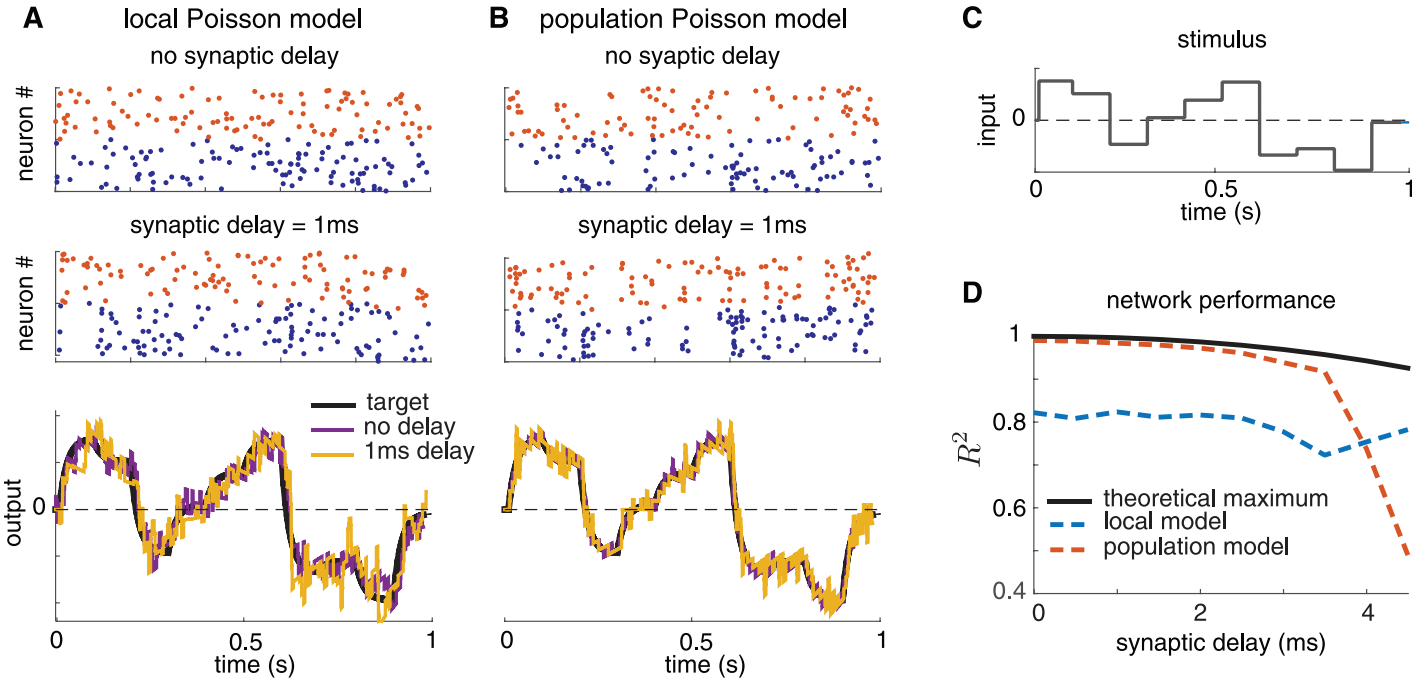

Fig 7. Illustration of local and population conditionally Poisson BSN frameworks with synaptic delays. (A) Spike trains simulated from the local Poisson framework implementing a 1D exact integrator, both without (top) and with a 1-ms synaptic delay (middle). The network output accurately tracked the target variable for both models (bottom). As before, red/blue spike trains indicate neurons with positive/output weights. **(B)** Analogous plots for population Poisson framework. **(C)** Stimulus used for simulations shown in A and B. **(D)** Coefficient of determination (R^2) computed using 50 simulations of each framework. Black trace indicates the maximum possible R^2 value that could be obtained given the exponential Euler integration rule (see Methods).

https://doi.org/10.1371/journal.pcbi.1008261.g007

across the population over some time window of length κ will minimize error. This leads to the following network objective function:

$$E_t = ||\mathbf{x}_t - (\hat{\mathbf{x}}_t + \kappa W\boldsymbol{\lambda})||_2^2, \qquad (\text{`}population\text{'}\ error\ function) \tag{19}$$

where $\mathbf{x}_t$ is the target output at time t, $\hat{\mathbf{x}}_t$ is the actual network output at time t, W are the decoding weights, and $\boldsymbol{\lambda}$ is the vector of firing rates (conditional intensities) of a network of Poisson spiking neurons. In this expression, $\kappa W\boldsymbol{\lambda}$ is expected contribution to network output over a time window of size κ. For implementation in discrete time, κ should be an integer multiple of the bin size Δ.

To minimize the above error, we set the instantaneous spike rate vector equal to the least-squares solution:

$$\boldsymbol{\lambda}_t = \frac{1}{\kappa}\tilde{W}(\mathbf{x}_t - \hat{\mathbf{x}}_t) \tag{20}$$

where $\tilde{W} = W^\top(WW^\top)^{-1}$ is the Moore-Penrose pseudo-inverse of the decoding weight matrix W. Poisson neurons firing independently with conditional intensity $\boldsymbol{\lambda}_t$ will therefore minimize the expected error between target and actual network output. Note that if W has orthogonal unit-vector rows, such that $WW^\top = I$, then $\tilde{W} = W^\top$ and we obtain the same encoding weights as the original BSN framework. For the implementation of the population framework, we make the same substitution of the proxy variable $\mathbf{z}$ for $\mathbf{x}$.

In the local framework, increasing the population size means more neurons are competing to reduce the read-out error in a single time window. This increased activity can lead to ping-ponging. The population level view of the problem scales the probability of spiking, $\boldsymbol{\lambda}_{[i]}$,

by $WW^\top$, which increases with population size. This re-scaling effectively spreads the 'responsibility' of correcting an error across the entire population by incorporating information about the total network size in an individual neuron's activity.

However, the solution in (Eq 19) is not valid generally because the right-hand-side can take on negative values, whereas the conditional intensity for a Poisson process must be positive. To overcome this, we create two mirrored copies of the population. Positive firing rates are assigned to one copy with weight vector W, and negative firing rates are assigned to the other copy with weight vector $-W$. We distinguish between these neurons and 'anti-neurons' by the sign of their membrane potential as determined by the least squared solution. Similarly to the local and BSN models, spikes from either population will have opposite effects on the output variable. However, in this case these designations are not fixed labels and do not apply to the actual sign of $\mathbf{w}_i$. For example, spikes from the i'th neuron will contribute $\mathbf{w}_i$ to the network output, while a spike from its 'anti-neuron' counterpart will have a contribution of $-\mathbf{w}_i$ to network output, but the entries of $\mathbf{w}_i$ themselves may be positive or negative.

Formally, we define the population framework in terms of the update equations:

$$\begin{aligned}\mathbf{v}_t &= \tilde{W}(\mathbf{z}_t - \hat{\mathbf{x}}_t) \\ &= \mathbf{v}_{t-1} + \Delta(\tilde{\Omega}\mathbf{r}_{t-1} + \tilde{W}\mathbf{c}_t) \quad (pre\text{-}spike\ membrane\ potential)\end{aligned} \tag{21}$$

$$\boldsymbol{\lambda}_t^{(+)} = \frac{1}{\kappa}\max(\mathbf{v}_t, 0) \qquad (neuron\ spike\ rate) \tag{22}$$

$$\boldsymbol{\lambda}_t^{(-)} = \frac{1}{\kappa}\max(-\mathbf{v}_t, 0) \qquad (anti\text{-}neuron\ spike\ rate) \tag{23}$$

$$\mathbf{s}_t^{(+)} \sim \text{Poiss}(\Delta\boldsymbol{\lambda}_t^{(+)}) \qquad (neuron\ spikes) \tag{24}$$

$$\mathbf{s}_t^{(-)} \sim \text{Poiss}(\Delta\boldsymbol{\lambda}_t^{(-)}) \qquad (anti\text{-}neuron\ spikes) \tag{25}$$

$$\mathbf{r}_t = \left(1 - \Delta\tfrac{1}{\tau}\right)\mathbf{r}_{t-1} + \mathbf{s}_t^{(+)} - \mathbf{s}_t^{(-)} \qquad (population\ filtered\ spike\ rates) \tag{26}$$

where $\tilde{\Omega} = \tilde{W}\left(A + \frac{1}{\tau}\right)W$ are the coupling weights from $\mathbf{r}_{t-1}$ to the pre-spike membrane potentials. This differs from the standard BSN framework in that spikes, rather than being driven by deterministic threshold crossing, arise from a Poisson process with conditional intensity $\boldsymbol{\lambda}_t^{(+)}$ or $\boldsymbol{\lambda}_t^{(-)}$. If $\mathbf{v}_{[i]t}$ is negative, then $\boldsymbol{\lambda}_{[i]t}^{(+)}$ is set to zero, and the corresponding anti-neuron's firing rate is positive. The voltage updates and spiking resets are identical to the local and BSN models.

Fig 6 shows a comparison of the activity of the local and population-level frameworks, as well as the accuracy and spike counts between all three models. Fig 6A and 6B show that the local and population models perform similarly, although the population framework has a higher spike rate. Between the three models, the original BSN model has a lower decoding error than the local and population Poisson models for both the one-dimensional and two-dimensional dynamical systems (Fig 6C), although at the expense of a greatly increased number of spikes (Fig 6D). This is due to the deterministic spike rule, which enforces that any spike fired must perfectly compensate for the error in every given time bin (or as well as is allowed by the size of the decoding weights, W). Spiking in the local and population frameworks is *driven* by this requirement, but is stochastic.

However, in terms of accuracy, all three models had similar R^2 values: 0.9961 (BSN), 0.9957 (local), and 0.9928 (population) for the 1-D and 0.9686 (BSN), 0.9395 (local) and 0.9565 (population) for the 2-D dynamical system. Thus, the probabilistic frameworks, on average, compensate for the error as well as the original BSN. But in any given time bin, there is a certain probability of firing too early or too late to optimally compensate for the error, which results in a higher RMSE. Note that the performance of the local and population models depends on the parameters of the firing rate nonlinearity and on κ, respectively, and modifying those can move the models into lower-error (and higher spike count) regimes.

Incorporating synaptic time delays

The original BSN model relies on near-instantaneous synaptic communication between neurons since all neurons in the population must reset immediately after a spike in any neuron. A more realistic model would require that synaptic inputs arrive only after a brief synaptic delay; only the reset of a neuron's own membrane potential following a spike could be considered instantaneous.

To test the robustness of the two Poisson BSN frameworks introduced above, we altered synaptic currents to incorporate a synaptic delay in neural outputs. In the revised model, filtered spike trains and "fast" membrane potential resets are received by other neurons only after a synaptic delay d. Thus, if neuron fires at time t, it resets its own membrane potential in the next time bin, but we will update spike trains and filtered spike trains received by other neurons only at time $t + d$.

To compensate for synaptic delays, we altered the network dynamics so that membrane potential—instead of reflecting the *current* error, as in the original BSN— reflects the network error extrapolated d time steps into the future. The basic logic of this approach is that the network dynamics should look into the future to predict whether firing a spike now will reduce error at the time when the spike actually arrives. The resulting solution is equivalent to minimizing an objective function (Eqs 4 or 19) where $\mathbf{x}_{t+d}$ and $\hat{\mathbf{x}}_{t+d}$ take the place of $\mathbf{x}_t$ and $\hat{\mathbf{x}}_t$.:

$$\mathbf{v}_t = W^\top(\mathbf{z}_{t+d} - \hat{\mathbf{x}}_{t+d}), \qquad (\textit{membrane potential with synaptic delay}) \tag{27}$$

Here, $\mathbf{z}_{t+d} = \exp(Ad)\mathbf{z}_t + \frac{c}{A}(e^{Ad} - 1)$ and $\hat{\mathbf{x}}_{t+d} = \exp(-d/\tau)\hat{\mathbf{x}}_t = W\exp(-d/\tau)\mathbf{r}_t$ represent the extrapolated target and network outputs at time $t + d$, respectively. This solution arises by analytically solving the differential equation $\frac{d\mathbf{z}}{dt} = A\mathbf{z} + c$, which describes the target dynamics, for $\mathbf{z}$ at time $t + d$ given an initial condition $\mathbf{z}_t$ and $c = c(t)$, and solving the equation $\frac{d\mathbf{r}}{dt} = -(1/\tau)\mathbf{r}$, which describes passive network dynamics in the absence of spiking, for $\mathbf{r}$ at time $t + d$ given an initial condition $\mathbf{r}_t$. For the population-level Poisson framework, the encoding weights $\tilde{W}$ replace $W^\top$ in (Eq 27). (See Methods for details).

Fig 7 shows an analysis of the accuracy of the local and population-level Poisson frameworks with synaptic delays. For both models, a 5ms synaptic delay does not have pronounced effects on the spiking activity or the quality of the read-out (Fig 7A and 7B). Fig 7D shows the R^2 values as a function of time delay for both frameworks. We compare it against a theoretical upper bound on the coding accuracy (in black), which is a consequence of the exponential Euler approximation (see Methods for details).

For the local framework, the R^2 value is lower than for the simulation shown in Fig 6 because the parameters were chosen to make the network more robust to synaptic delays. Otherwise, in the high-precision (high α) regime with synaptic delays, ping-ponging may result, as shown by the higher error regime in Fig 5B. By contrast, the population framework can maintain high levels of accuracy for a large range of synaptic delays.

These revised dynamics could also be used to increase accuracy of a standard BSN with synaptic delays, for example as defined in [36]. We attempted to incorporate synaptic delays into our implementation of the standard BSN, but it led to ping-ponging, after which the model was no longer able to track the target dynamics. The resulting R^2 values were negative, so they are not included in Fig 7D.

Network performance

Lastly, we analyzed the performance of our local and population frameworks as compared to the original BSN model. Fig 8 shows the cross-correlations of spike trains generated by a network of forty neurons implementing a one-dimensional integrator, $\dot{x}(t) = c(t)$, with a white-noise stimulus $c(t)$, for the local, population and original BSN models. Local and population Poisson BSN models both enforced a synaptic delay of 1 ms.

To compute cross-correlations, we divided the neurons into positive-output and negative-output groups. We then computed average within-group (positive-positive and negative-negative) and across-group (positive-negative) cross-correlations. These curves show substantial differences between the original BSN model and the two Poisson models. First, cross-correlations of the original BSN model are 0 at lag zero, due to the rule that only one neuron can spike in a single time bin. More importantly, the within-group correlations for the BSN model exhibit a trough at time zero, meaning that neurons with the same output weight are anti-correlated. Conversely, across-group correlations exhibit an increase at small lags, meaning that neurons with opposite sign output weights are more likely to fire together in a small time window. This relationship is at odds with correlations in both retina and visual cortex, where studies have reported that correlations are highest for neurons with similar tuning, and lowest for neurons with dissimilar tuning [34, 41–43]. By contrast, the local and population Poisson models successfully recapitulate this pattern of correlations, with a peak in the cross-correlations between pairs of neurons with the same sign weights, and a trough for pairs of opposite-sign neurons. Cross-correlations from these models also exhibit no trough at zero due to the lack of a rule prohibiting simultaneous spiking. Thus, cross-correlations represent an additional dimension of biological plausibility of the proposed Poisson frameworks.

Fig 9A shows the relationship between root-mean-square error (RMSE) and network size (N). The RMSE values shown are *relative* values, normalized in order to best compare the trend in error with respect to N. The original BSN model has a decoding error that scales

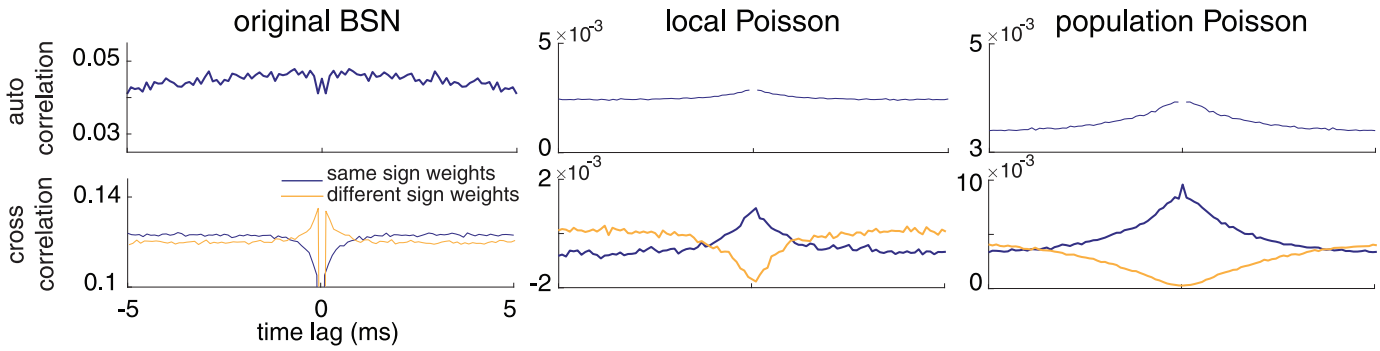

Fig 8. Cross- and auto-correlations for the original BSN, local Poisson and population Poisson BSN models with synaptic delay. The top row shows average auto-correlations across both populations of neurons. The bottom row shows average cross-correlations for pairs of neurons with the same sign output weight (i.e., both positive or both negative, in purple) and for pairs of neurons with opposite-sign output weights (e.g., one positive and one negative neuron, in yellow). The original BSN network exhibits negative correlations between neurons with the same sign, and positive correlations between neurons with opposite sign. The local and Population Poisson models show the opposite pattern, which more closely resembles correlations found in neural populations in (e.g.) visual cortex.

https://doi.org/10.1371/journal.pcbi.1008261.g008

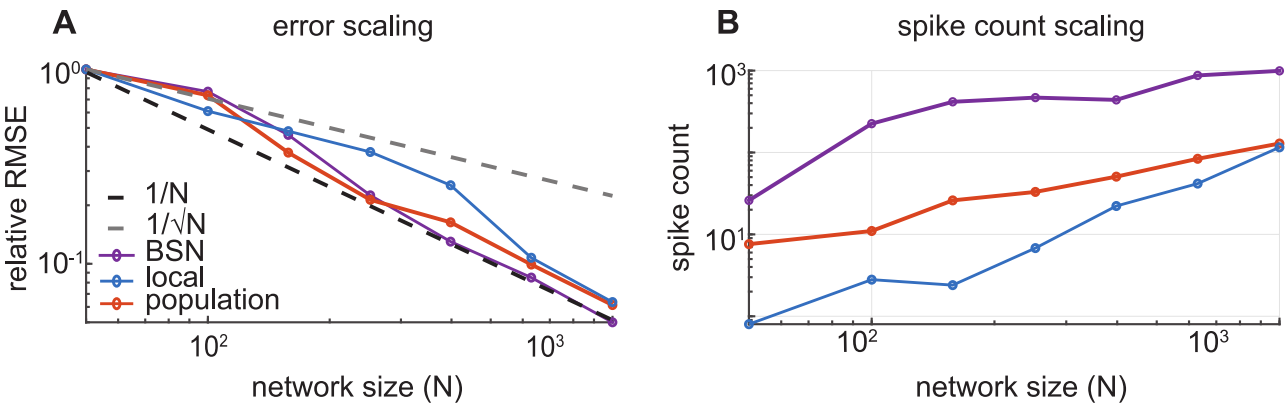

Fig 9. Scaling of error and spike count with population size (weights held fixed). (A) Relative root-mean-square error (RMSE) decreases approximately linearly with the network size for all three models. R^2 values for the fit to $1/N$ were .98, .91, and .82 for the BSN, local and population models, respectively. **(B)** Total spike count as a function of network size. The results shown are the average of five simulations of the networks performing exact integration of a signal formed by a sum of two sinusoids.

https://doi.org/10.1371/journal.pcbi.1008261.g009

with $\frac{1}{N}$, provided that the weights are likewise scaled by $\frac{1}{N}$, and we find similar scaling for the local and population frameworks, although the BSN has a tighter bound. Poisson rate codes typically scale as $\frac{1}{\sqrt{N}}$ due to the Cramér-Rao bound, which places a lower bound on the variance (and therefore the mean-squared-error) of an unbiased estimator. We might expect the local and population networks to scale similarly.

However, the $\frac{1}{\sqrt{N}}$ scaling applies under the assumption of a constant firing rate. Our local and population models have rates that are a function of voltage dynamics, or the coding error. Although the spiking *mechanism* is Poisson, the conditional intensity is still *driven* by the coding error. So even if the Poisson spikes over- or under-shoot the target in a particular time window, on average the network corrects it in the following bins. In sum, this makes our frameworks' coding capabilities scale better than $\frac{1}{\sqrt{N}}$.

We also looked at how spike counts varied with network size, shown in Fig 9B. In general, as the weights get smaller, the read-out is more precise, but more spikes are needed to reconstruct the target. The local framework initially has a low spike rate because of the large weights and higher error tolerance ($\alpha = 1000$). However, as the weights get smaller, spiking increases and the local framework exhibits ping-ponging at very small weight sizes. Likewise in the BSN, for small weight sizes the decoding error increases because weights become too small to encode the target given the one-spike rule.

Finally, we looked at how robust both frameworks are to neuron loss. The authors of [31] show in Fig 7G that the original BSN is robust to sudden inactivation of neurons. We ran a similar simulation, shown in Fig 10, silencing 50% of the negatively weighted and positively weighted neurons for.6s at a time by setting the probability of spiking in that time bin to 0. Like in the original BSN, our networks maintain coding accuracy despite large neuron "loss" by increasing the firing rates of remaining neurons.

Discussion

In this paper, we have highlighted a shortcoming of the balanced spiking network (BSN) paradigm, namely the requirement of near-instantaneous communication between neurons, which arises from the fact that a spike in any neuron causes an instantaneous reset of membrane potential in all other neurons. In practice, the BSN model is often implemented with the additional rule that only one neuron can spike in a single time bin. When synaptic delays are

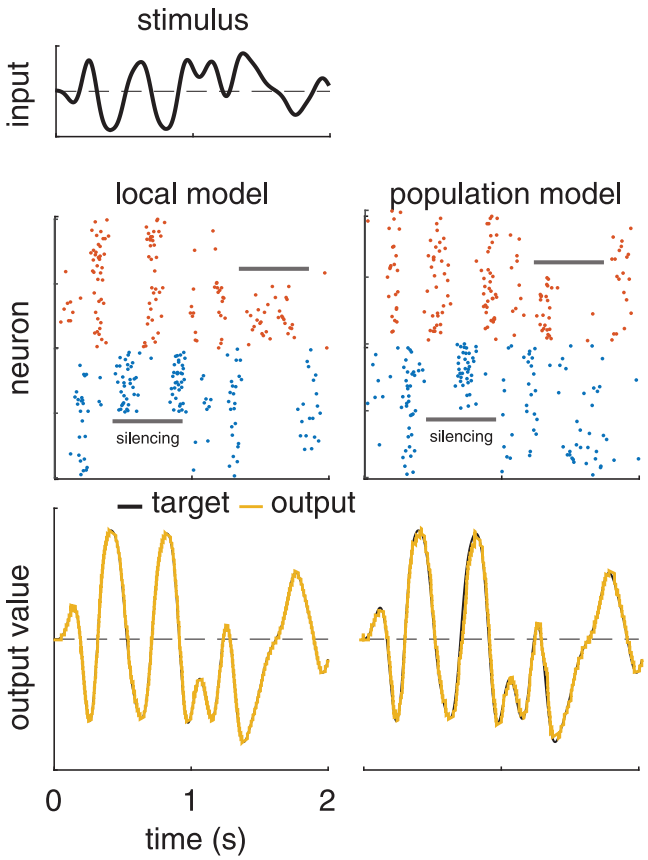

Fig 10. Simulation showing robustness of local and population Poisson models to silencing of a subset of neurons. **Left**: We created a local Poisson BSN model with 400 neurons, with weights set to perform exact integration, and presented it with a slowly varying 1D stimulus (top). From time $t = .4$ to $t = 1$s we artificially silenced 50% of neurons in the negative-weight (blue) population, preventing them from spiking by setting $p(\text{spike}) = 0$ (silencing period indicated by the grey bar). From time $t = 1.2$ to $t = 1.8$s we did the same for 50% of neurons in the positive-weight (red) population. **Right**: Likewise for the population framework.

https://doi.org/10.1371/journal.pcbi.1008261.g010

introduced, or multiple spikes are allowed per bin, the model easily enters a ping-ponging regime in which the network output overshoots and undershoots the target output on alternating time bins.

To address this problem, we proposed two extensions to the BSN model that incorporate conditionally Poisson spiking. Our proposed models both preserve the readout structure of the original BSN, in which a linear combination of exponentially filtered spike trains approximates a linear dynamical system of interest. However, they both replace of "hard threshold" integrate-and-fire spiking of the original BSN with a spiking process governed by an instantaneous spike rate or conditional intensity. We note that although conditioning spiking allows for more stable network activity and for synaptic delays, we do so at the expense of the single time bin precision of the original BSN (see, e.g., Fig 6C). The BSN spikes deterministically to correct the coding error in a given time bin, while the local and Poisson frameworks may fire spikes earlier or later than optimal. As such, we relax the guarantee that spikes fired will precisely compensate for the coding error.

In the local Poisson BSN framework, the conditional intensity arises from passing the membrane potential through a sigmoidal nonlinearity. The accelerating phase of this nonlinearity is consistent with nonlinearities observed in neural data [44–47] and closely resembles the

exponential nonlinearity commonly used in generalized linear modeling analyses [33, 34], while the saturating phase is consistent with saturation in real neural firing rates.

In the "population" Poisson BSN framework, the conditional intensity is obtained by setting the vector of expected spike counts to the least-squares solution for the total output error. This model differs from the original BSN in that the encoding weights, the linear mapping from output error to membrane potential, uses the pseudo-inverse of the decoding weights, $\tilde{W}$, whereas the original BSN used the transpose $W^{\top}$. This change ensures that the spike rate of each neuron takes account of how many other neurons in the population have similar decoding weights, so that the expected spike count across the entire population in some finite time window compensates optimally for the output error. These modifications make both frameworks robust to both parameter settings and synaptic delays on realistic time scales (1-3ms). They also allow the network to have realistic auto- and cross-correlations, while preserving the decoding accuracy and the robustness of the original BSN.

Related work

Recent literature has explored a variety of other extensions and applications of the BSN framework, including nonlinear dynamical systems and the learning of synaptic weights [48, 49], synaptic plasticity rules [50], and biological extensions like finite timescale synapses [51] and synaptic delays [52]. The BSN framework has also been adapted to other computational problems such as probabilistic computation [53] and sensory adaptation [54].

Our paper is not the first to address the issue of instability in the BSN. Recent work from [52] examined the use of penalties on spiking to reduce ping-ponging (referred to in that paper as "up states"). We found that this strategy required fine-tuning and succeeded in a relatively narrow parameter regime compared to the solutions we proposed here. Other work by by Chalk et al. [36] has argued that oscillations in the brain activity may arise from BSNs with synaptic delays, suggesting that a substantially damped form of ping-ponging may be a signature of efficient computation in neural circuits. This work used the finite timescale dynamics of [51] to implement the BSN with synaptic delays. Similarly to our work, spiking activity in the Chalk model is asynchronous and sparse, and neurons are not prevented from firing synchronously. However, the Chalk model avoids ping-ponging behavior by injecting a carefully tuned amount of noise into membrane potential and using slow membrane time constants (100ms). The added membrane noise degrades the network representation of coding error, forcing a trade-off between stability and accuracy of representation. By contrast, our models do not require tuning membrane potential noise to achieve stability; although firing is stochastic, the membrane potential maintains an accurate representation of the error over a wide range of input and network sizes. Our network also uses membrane time constants with more biophysically plausible ranges (10-20ms).

The topic of balanced networks has also received considerable attention outside the specific BSN framework introduced by [31]. Balanced networks have been proposed as a substrate for working memory [55, 56], probabilistic inference [57, 58], and the control of complex movements [59]. Excitatory-inhibitory balance is also a key topic in the mathematical theory of neural circuit dynamics, where it has been proposed as an explanation for the correlations found in large-scale population activity [60–62]. Finally, a rich literature has focused on the training of spiking neural networks in more general supervised and reinforcement learning settings, where the objective involves task performance or can only be evaluated at the end of a trial [63–67].

Our population Poisson model is similar to work by Eliasmith and Anderson [68] in that the network objective is to minimize the expected squared error between a target function and

a network estimate using a spiking neural network. However, their model does so by optimizing for the decoding weights instead of network activity and uses a spiking mechanism that is driven linearly by the magnitude of the input, instead of the error-correcting computational principle used in the BSN. Later work by Eliasmith [69] extended this framework to allow for embedding of the desired dynamics (i.e., the A matrix) into the population activity.

Our work also connects to a rich literature on point process models of neural spike trains. The local Poisson framework draws direct inspiration from the work of [32], which sought to approximate a noisy integrate-and-fire model with an inhomogeneous Poisson process via the so-called "escape-rate approximation", which refers to the instantaneous probability of noisy membrane potential crossing threshold in a small time window. Subsequent work on the spike response model [39, 70–73] and Poisson generalized linear model [33–35, 74–76] further explored the connection between integrate-and-fire and conditionally Poisson spike train models. The latter are sometimes referred to as "soft-threshold" integrate-and-fire model [77], making the local Poisson model a natural extension of the original BSN model.

Future challenges

Although our proposed frameworks are a step in the direction of biological plausibility, there remain a variety of open challenges. A straightforward way of extending the biological realism of our proposed network is to reformulate it with a separate inhibitory population to comply with Dale's law, as was done in [36]. Another extension is the development of neurally plausible learning rules and weight patterns. The network we proposed has a static weight matrix with all-to-all connectivity. A more realistic model would allow for sparse connectivity, sign constraints forcing neurons to be purely excitatory or inhibitory, and plausible learning rules that allow weights to change over time as a function of reward signals. There have been successes learning the fast, slow and feed-forward weights through non-local, supervised, control theoretic approaches [48, 49] or, much more recently, through local, Hebbian plasticity rules [78]. Our probabilistic formulation of the Poisson BSN frameworks makes implementing local, Hebbian plasticity rules more tractable, as it opens up the possibility of applying unsupervised learning techniques from traditional machine learning methodology.

We note that our implementation of synaptic delays merely shifts the arrival time of spikes but still involves an instantaneous jump in the filtered spike train dynamics. In future work, we hope to implement time delays with a finite rise time, as was done in [51]. We suspect that since our network is stable with instantaneous jumps in $\mathbf{r}(t)$, it will also be stable with smoothly increasing spiking dynamics. Finally, at the level of implementation, we recognize that although our model does not rely on additional spiking conditions or spike-time interpolation, using Poisson firing dynamics instead of the integrate-and-fire approximation represents a slight step away from biological plausibility. We also hope to address the biological plausibility of 'anti-neurons' in the population framework.

Another main challenge is the incorporation of nonlinear dynamics. Although the original BSN model was designed to implement linear dynamical systems, it is well known that a wide variety of neural computations are nonlinear. Recent work has proposed an extension of the BSN framework to nonlinear dynamics [30, 48]; combining this approach with conditionally Poisson spiking therefore represents a promising avenue for future work.

Finally, the conditionally Poisson extensions we have proposed provide new opportunities for applying the BSN framework to the interpretation and analysis of real neural data sets. Both the original BSN model and ours assume access to the precise spiking patterns of all the neurons in a population, but real neural recordings typically record only a small fraction of the neurons in a population. Previous work has shown that latent BSN dynamics can be recovered

from spike trains in the fully observed case [53]. Other work has discussed the recovery of Poisson generalized linear models from partial recordings [79, 80]. This motivates the development of new methods for identifying balanced network dynamics and computations from partially observed data sets, which may offer fundamental insights into spike-based computation in the brain.

Methods

Exponential integrator

Exponential integrators are a class of numerical methods for solving first-order differential equations of the form

$$\dot{y}_t = -Ay_t + g(y_t) + c \tag{28}$$

where $-Ay_t$ is a linear term with a dynamics matrix A, c is a constant, and the nonlinear terms are grouped in $g(y_t)$. We use this method of integration throughout our simulations because it is much more numerically stable than explicit Euler methods. For equations without nonlinear terms, it above can be solved exactly from time 0 to a later time t as

$$y_t = y_0 e^{At} + \frac{c}{A}(e^{At} - 1) \tag{29}$$

For $A = 0$,

$$y_t = y_0 + ct \tag{30}$$

Using exponential integrators allows us to implement a time delay of magnitude d by having the network spike on the basis of an extrapolated future error at a time $t + d$. Specifically, we have the network voltage track the error between a predicted $\mathbf{x}(t + d)$ and $\hat{\mathbf{x}}(t + d)$. Eq 16 then becomes

$$\begin{aligned} \mathbf{v}_t &= W^\top(\mathbf{z}_{t+d} - \hat{\mathbf{x}}_{t+d}) \\ &= W^\top(e^{Ad}\mathbf{z}_t + \frac{c}{A}(e^{Ad} - 1) - e^{-d/\tau}W\mathbf{r}_t) \end{aligned} \tag{31}$$

where we have assumed that the input to the network, $c(t)$, stays at constant value, c, from t to $t + d$ and that the spike rate evolves passively in the absence of spiking. In the case that $A = 0$,

$$\mathbf{v}_t = W^\top(\mathbf{z}_t + cd - e^{-d/\tau}W\mathbf{r}_t) \tag{32}$$

In the case of the population framework, the $W^\top$ above is replaced by $\tilde{W}^\top$.

Theoretical minimum error. In Fig 7D we compared the R^2 values for the local and population models as a function of delay to a theoretical upper bound. This bound comes from the expected mismatch between the predicted $\mathbf{x}_{t+d} = e^{Ad}y_t + \frac{c}{A}(e^{Ad} - 1)$ and the actual value of $\mathbf{x}_{t+d}$. To determine this bound, we integrated the target dynamics for the length of the simulation with exponential Euler integration for a range of delays (d). We then compared it to the target dynamics integrated without a time delay to get an R^2 for each delay length d.

Simulation parameters

For our simulations, all parameters are non-dimensionalized. Time is measured in seconds and $dt = 0.1$ms. All other simulation parameters are shown in Table 1, below.

Fig 2 was generated using the parameter settings described in Boerlin *et al*, fig. 1C (included below for comparison). The cost terms are $\mu = 10^{-6}$ and $\nu = 10^{-5}$ and the voltage decay constant

Table 1. Simulation parameters.

Figure	N	W	α	F_{max}	F_{min}	κ	A	τ	d
Fig 2	400	± 0.1	N/A	N/A	N/A	N/A	0	10	0
Fig 4D, BSN	1	1	N/A	N/A	N/A	N/A	0	20	0
Fig 4D, GLM	1	1	Varies	50	0	N/A	0	20	0
Fig 5	400	±.1 + noise (μ = .01)	1e4	1	0	N/A	0	10	0
Fig 6A and 6B	400	±.1 + noise (μ = 1)	1000	1	0	200	0	1	0
Fig 6C and 6D	400	±.1 + noise (μ = 1)	1000	1	0	200	$\begin{pmatrix} -1 & -10 \\ 10 & -1 \end{pmatrix}$	1	0
Fig 7A and 7B	200	± 0.02 + noise (μ = .01)	800	100	0	30	-50	0.2	0 or 5ms
Fig 8	40	± 0.0025 + noise (μ = .01)	2000	5000	5	20	0	0.5	1ms
Fig 9	varies (50-1000)	$\pm\frac{300}{N}$	1000	2	0.01	1	-100	20	0
Fig 10	400	± 0.2	5000	3	0	50	-100	1	0

https://doi.org/10.1371/journal.pcbi.1008261.t001

is $\tau_v = 20$. The noise added to the voltage dynamics and the stimulus was Gaussian with $\sigma_v = 10^{-3}$ and $\sigma_c = 0.01$, respectively. When enforcing the constraint that one neuron should spike per time bin, we selected the neuron with the highest voltage above threshold. Another option with similar results would be to select randomly between the ones above threshold.

As a reminder, N is the number of neurons in the network and W is the vector of read-out weights. For the local framework, α is the slope of the exponential nonlinearity, F_{max} is the saturation, and F_{min} is the minimum firing rate. For the population framework, κ is the time window over which the network minimizes the error. Finally, A is the dynamics matrix of the linear dynamical system, τ is the decay time constant of the filtered spike trains $\mathbf{r}$, and d is the time delay.

A software implementation of Poisson BSNs, along with code to re-generate figures shown in the manuscript, is available at https://github.com/pillowlab/PoissonBalancedNets.

Cost terms

The original BSN model objective function incorporated two additional cost terms to penalize spiking: a quadratic cost term μ and the linear cost term v. These terms encouraged the network to use fewer spikes and to distribute spiking more evenly across neurons with large and small output weights. We did not incorporate these in our derivation for clarity, but they are included in our simulations of the BSN.

Including both cost terms into the derivation in the Results section, Eq (4) becomes

$$E_t = ||\mathbf{x}_t - \hat{\mathbf{x}}_t||_2^2 + v||\mathbf{r}_t||_1 + \mu||\mathbf{r}_t||_2^2 \tag{33}$$

and the voltage and threshold equations become

$$\mathbf{v}_t = W^\top(\mathbf{x}_t - \hat{\mathbf{x}}_t) - \frac{\mu}{\tau}\mathbf{r}_t \tag{34}$$

$$\dot{\mathbf{v}}_t = -\frac{1}{\tau_v}\mathbf{v}_{t-1} + \Omega\mathbf{r}_t - (W^\top W + \frac{\mu}{\tau^2})\mathbf{s}_t + W^\top\mathbf{c}_t \tag{35}$$

$$T_i = \frac{\frac{v}{\tau} + \frac{\mu}{\tau^2} + ||W_i||^2}{2} \tag{36}$$

The linear cost term (ν) is proportional to the L1 norm of $\mathbf{r}(t)$, or $||\mathbf{r}||_1 = \sum_{i=1}^{N} r_{[i]t}$. This cost term penalizes the network's total activity. The quadratic cost term (μ) limits individual neuron firing rates, forcing a spread of activity across all neurons in a population. Over time, the network transfers activity from precise, costly neurons with high firing rates to imprecise, larger weighted neurons to maintain a compromise between efficiency and accuracy of the read-out. Boerlin *et al* also include a voltage leak term for biological realism.

In our Poisson models, we did not observe the ping-pong effects described in Boerlin *et al* for the range of parameters we considered, so we don't need cost terms for network stability. For the local Poisson framework, the cost terms can be included when α and F_{max} are high enough to cause ping-ponging.

Performance metrics

We use R^2 as a measure for how well the network read-out is approximating the target variable. The formula for calculating these is

$$R^2 = 1 - \frac{\sum_{t=0}^{T} (\hat{\mathbf{x}}_t - \mathbf{x}_t)^2}{\sum_{t=0}^{T} (\hat{\mathbf{x}}_t - \hat{\bar{\mathbf{x}}})^2} \tag{37}$$

for a simulation of time length T, where $\hat{\bar{\mathbf{x}}}$ is the mean of $\hat{\mathbf{x}}$ over the entire simulation. The root-mean-squared error (RMSE) is simply

$$RMSE = \frac{1}{T} \sum_{t=0}^{T} (\hat{\mathbf{x}}_t - \mathbf{x}_t)^2 \tag{38}$$

The cross- and auto-correlations between spike trains were calculated as unbiased estimates $\hat{r}$ with a maximum lag of $l = 50$ time bins according to

$$\hat{r} = \frac{1}{T-l} \left(\sum_{n=0}^{T-l-1} x_n y_{n+l} \right) \tag{39}$$

where x and y are spike trains from two different neurons.

Author Contributions

Conceptualization: Camille E. Rullán Buxó, Jonathan W. Pillow.

Formal analysis: Camille E. Rullán Buxó, Jonathan W. Pillow.

Funding acquisition: Jonathan W. Pillow.

Software: Camille E. Rullán Buxó.

Supervision: Jonathan W. Pillow.

Validation: Camille E. Rullán Buxó.

Visualization: Camille E. Rullán Buxó, Jonathan W. Pillow.

Writing – original draft: Camille E. Rullán Buxó, Jonathan W. Pillow.

Writing – review & editing: Camille E. Rullán Buxó, Jonathan W. Pillow.

References

1. Gold JI, Shadlen MN. Neural computations that underlie decisions about sensory stimuli. Trends Cogn Sci. 2001 Jan; 5(1):10–16. https://doi.org/10.1016/S1364-6613(00)01567-9 PMID: 11164731

2. Wong KF, Huk AC, Shadlen MN, Wang XJ. Neural circuit dynamics underlying accumulation of time-varying evidence during perceptual decision making. Front Comput Neurosci. 2007; 1:6. https://doi.org/10.3389/neuro.10.006.2007 PMID: 18946528

3. Brunton BW, Botvinick MM, Brody CD. Rats and Humans Can Optimally Accumulate Evidence for Decision-Making. Science. 2013; 340(6128):95–98. https://doi.org/10.1126/science.1233912 PMID: 23559254

4. Mante V, Sussillo D, Shenoy KV, Newsome WT. Context-dependent computation by recurrent dynamics in prefrontal cortex. Nature. 2013; 503(7474):78–84. https://doi.org/10.1038/nature12742 PMID: 24201281

5. Machens CK, Romo R, Brody CD. Flexible control of mutual inhibition: a neural model of two-interval discrimination. Science. 2005; 307(5712):1121–1124. https://doi.org/10.1126/science.1104171 PMID: 15718474

6. Goldman MS. Memory without Feedback in a Neural Network. Neuron. 2009; 61(4):621–634. https://doi.org/10.1016/j.neuron.2008.12.012 PMID: 19249281

7. Barak O, Sussillo D, Romo R, Tsodyks M, Abbott LF. From fixed points to chaos: Three models of delayed discrimination. Progress in Neurobiology. 2013; 103(0):214–222. Conversion of Sensory Signals into Perceptions, Memories and Decisions. https://doi.org/10.1016/j.pneurobio.2013.02.002 PMID: 23438479

8. Churchland MM, Cunningham JP, Kaufman MT, Ryu SI, Shenoy KV. Cortical preparatory activity: representation of movement or first cog in a dynamical machine. Neuron. 2010; 68(3):387–400. https://doi.org/10.1016/j.neuron.2010.09.015 PMID: 21040842

9. Churchland MM, Cunningham JP, Kaufman MT, Foster JD, Nuyujukian P, Ryu SI, et al. Neural population dynamics during reaching. Nature. 2012; 487(7405):51–56. https://doi.org/10.1038/nature11129 PMID: 22722855

10. Pandarinath C, O'Shea DJ, Collins J, Jozefowicz R, Stavisky SD, Kao JC, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. Nature Methods. 2018; 15(10):805–815. https://doi.org/10.1038/s41592-018-0109-9 PMID: 30224673

11. Macke JH, Buesing L, Cunningham JP, Byron MY, Shenoy KV, Sahani M. Empirical models of spiking in neural populations. In: Advances in neural information processing systems. vol. 24; 2011. p. 1350–1358.

12. Petreska B, Byron MY, Cunningham JP, Santhanam G, Ryu SI, Shenoy KV, et al. Dynamical segmentation of single trials from population neural data. In: Advances in neural information processing systems; 2011. p. 756–764.

13. Buesing L, Macke J, Sahani M. Spectral learning of linear dynamics from generalised-linear observations with application to neural population data. In: Advances in Neural Information Processing Systems 25; 2012. p. 1691–1699.

14. Archer EW, Koster U, Pillow JW, Macke JH. Low-dimensional models of neural population activity in sensory cortical circuits. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, editors. Advances in Neural Information Processing Systems 27. Curran Associates, Inc.; 2014. p. 343–351.

15. Archer E, Park IM, Buesing L, Cunningham J, Paninski L. Black box variational inference for state space models. arXiv preprint arXiv:151107367. 2015;.

16. Gao Y, Archer EW, Paninski L, Cunningham JP. Linear dynamical neural population models through nonlinear embeddings. In: Advances in Neural Information Processing Systems; 2016. p. 163–171.

17. Zhao Y, Park IM. Variational Latent Gaussian Process for Recovering Single-Trial Dynamics from Population Spike Trains. Neural Computation. 2017; 29(5):1293–1316. https://doi.org/10.1162/NECO_a_00953 PMID: 28333587

18. Linderman S, Johnson M, Miller A, Adams R, Blei D, Paninski L. Bayesian Learning and Inference in Recurrent Switching Linear Dynamical Systems. In: Singh A, Zhu J, editors. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics. vol. 54; 2017. p. 914–922.

19. Hernandez D, Moretti AK, Wei Z, Saxena S, Cunningham J, Paninski L. A Novel Variational Family for Hidden Nonlinear Markov Models. arXiv preprint arXiv:181102459. 2018;.

20. Zhao Y, Park IM. Variational joint filtering. arXiv preprint arXiv:170709049. 2019;Memming's latest on nonlinear latent dynamics w/ Poisson noise.

21. Duncker L, Bohner G, Boussard J, Sahani M. Learning interpretable continuous-time models of latent stochastic dynamical systems. arXiv preprint arXiv:190204420. 2019;.

22. Sompolinsky H, Crisanti A, Sommers HJ. Chaos in Random Neural Networks. Phys Rev Lett. 1988 Jul; 61:259–262. https://doi.org/10.1103/PhysRevLett.61.259 PMID: 10039285

23. Vreeswijk C, Sompolinsky H. Chaos in neuronal networks with balanced excitatory and inhibitory activity. Science. 1996; 274(5293):1724. https://doi.org/10.1126/science.274.5293.1724 PMID: 8939866

24. Brunel N. Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons. Journal of computational neuroscience. 2000; 8(3):183–208. https://doi.org/10.1023/A:1008925309027 PMID: 10809012

25. Denève S, Latham PE, Pouget A. Efficient computation and cue integration with noisy population codes. Nat Neurosci. 2001 Aug; 4(8):826–831. https://doi.org/10.1038/90541

26. Ganguli S, Huh D, Sompolinsky H. Memory traces in dynamical systems. Proceedings of the National Academy of Sciences. 2008; 105(48):18970–18975. https://doi.org/10.1073/pnas.0804451105 PMID: 19020074

27. Sussillo D, Abbott LF. Generating coherent patterns of activity from chaotic neural networks. Neuron. 2009 Aug; 63(4):544–557. https://doi.org/10.1016/j.neuron.2009.07.018 PMID: 19709635

28. Eliasmith C, Stewart TC, Choo X, Bekolay T, DeWolf T, Tang C, et al. A large-scale model of the functioning brain. science. 2012; 338(6111):1202–1205. https://doi.org/10.1126/science.1225266 PMID: 23197532

29. Fisher D, Olasagasti I, Tank DW, Aksay ER, Goldman MS. A modeling framework for deriving the structural and functional architecture of a short-term memory microcircuit. Neuron. 2013; 79(5):987–1000. https://doi.org/10.1016/j.neuron.2013.06.041 PMID: 24012010

30. Abbott LF, DePasquale B, Memmesheimer RM. Building functional networks of spiking model neurons. Nature neuroscience. 2016; 19(3):350. https://doi.org/10.1038/nn.4241 PMID: 26906501

31. Boerlin M, Machens CK, Denève S. Predictive Coding of Dynamical Variables in Balanced Spiking Networks. PLoS Comput Biol. 2013 11; 9(11):e1003258. https://doi.org/10.1371/journal.pcbi.1003258 PMID: 24244113

32. Plesser HE, Gerstner W. Noise in integrate-and-fire neurons: from stochastic input to escape rates. Neural Comput. 2000 Feb; 12(2):367–384. https://doi.org/10.1162/089976600300015835 PMID: 10636947

33. Truccolo W, Eden UT, Fellows MR, Donoghue JP, Brown EN. A Point Process Framework for Relating Neural Spiking Activity to Spiking History, Neural Ensemble and Extrinsic Covariate Effects. J Neurophysiol. 2005; 93(2):1074–1089. https://doi.org/10.1152/jn.00697.2004 PMID: 15356183

34. Pillow JW, Shlens J, Paninski L, Sher A, Litke AM, Chichilnisky EJ, Simoncelli EP. Spatio-temporal correlations and visual signaling in a complete neuronal population. Nature. 2008; 454:995–999. https://doi.org/10.1038/nature07140 PMID: 18650810

35. Weber AI, Pillow JW. Capturing the Dynamical Repertoire of Single Neurons with Generalized Linear Models. Neural Computation. 2017; 29(12):3260–3289. https://doi.org/10.1162/neco_a_01021 PMID: 28957020

36. Chalk M, Gutkin B, Denève S. Neural oscillations as a signature of efficient coding in the presence of synaptic delays. Elife. 2016; 5:e13824. https://doi.org/10.7554/eLife.13824 PMID: 27383272

37. Hansel D, Mato G, C M, Neltner L. On numerical simulations of integrate-and-fire neural networks. Neural Computation. 1998; 10(2):467–483. https://doi.org/10.1162/089976698300017845 PMID: 9472491

38. Shelley M, Tao L. Efficient and accurate time-stepping schemes for integrate-and-fire neuronal networks. Journal of Computational Neuroscience. 2001;p. 111–119. https://doi.org/10.1023/A:1012885314187 PMID: 11717528

39. Jolivet R, Lewis T, Gerstner W. The Spike Response Model: a Framework to Predict Neuronal Spike Trains. Springer Lecture notes in computer science. 2003; 2714:846–853. https://doi.org/10.1007/3-540-44989-2_101

40. Latimer KW, Chichilnisky EJ, Rieke F, Pillow JW. Inferring synaptic conductances from spike trains with a biophysically inspired point process model. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, editors. Advances in Neural Information Processing Systems 27. Curran Associates, Inc.; 2014. p. 954–962.

41. Shlens J, Rieke F, Chichilnisky E. Synchronized firing in the retina. Curr Opin Neurobiol. 2008 Aug; 18(4):396–402. https://doi.org/10.1016/j.conb.2008.09.010 PMID: 18832034

42. Kohn A, Smith MA. Stimulus dependence of neuronal correlation in primary visual cortex of the macaque. J Neurosci. 2005 Apr; 25(14):3661–3673. https://doi.org/10.1523/JNEUROSCI.5106-04.2005 PMID: 15814797

43. Smith MA, Kohn A. Spatial and temporal scales of neuronal correlation in primary visual cortex. The Journal of Neuroscience. 2008; 28(48):12591–12603. https://doi.org/10.1523/JNEUROSCI.2929-08.2008 PMID: 19036953

44. Heeger DJ. Normalization of cell responses in cat striate cortex. Vis Neurosci. 1992 Aug; 9(2):181–197. https://doi.org/10.1017/S0952523800009640 PMID: 1504027

45. Britten KH, Shadlen MN, Newsome WT, Movshon JA. Responses of neurons in macaque MT to stochastic motion signals. Visual neuroscience. 1993; 10(6):1157–1169. https://doi.org/10.1017/S0952523800010269 PMID: 8257671

46. Geisler WS, Albrecht DG. Bayesian analysis of identification performance in monkey visual cortex: nonlinear mechanisms and stimulus certainty. Vision research. 1995; 35(19):2723–2730. https://doi.org/10.1016/0042-6989(95)00029-Y PMID: 7483312

47. Chichilnisky EJ. A simple white noise analysis of neuronal light responses. Network: Computation in Neural Systems. 2001; 12:199–213. PMID: 11405422

48. Alemi A, Machens C, Denève S, Slotine JJ. Learning arbitrary dynamics in efficient, balanced spiking networks using local plasticity rules. arXiv preprint arXiv:170508026. 2017;.

49. Denève S, Alemi A, Bourdoukan R. The brain as an efficient and robust adaptive learner. Neuron. 2017; 94(5):969–977. https://doi.org/10.1016/j.neuron.2017.05.016

50. Bourdoukan R, Barrett D, Denève S, Machens CK. Learning optimal spike-based representations. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ, editors. Advances in Neural Information Processing Systems 25. Curran Associates, Inc.; 2012. p. 2285–2293.

51. Schwemmer MA, Fairhall AL, Denéve S, Shea-Brown ET. Constructing precisely computing networks with biophysical spiking neurons. Journal of Neuroscience. 2015; 35(28):10112–10134. https://doi.org/10.1523/JNEUROSCI.4951-14.2015 PMID: 26180189

52. Koren V, Denève S. Computational account of spontaneous activity as a signature of predictive coding. PLoS computational biology. 2017; 13(1):e1005355. https://doi.org/10.1371/journal.pcbi.1005355 PMID: 28114353

53. Savin C, Denève S. Spatio-temporal representations of uncertainty in spiking neural networks. In: Advances in Neural Information Processing Systems; 2014. p. 2024–2032.

54. Gutierrez GJ, Denève S. Population adaptation in efficient balanced networks. eLife. 2019; 8. https://doi.org/10.7554/eLife.46926 PMID: 31550233

55. Lim S, Goldman MS. Balanced cortical microcircuitry for maintaining information in working memory. Nature neuroscience. 2013; 16(9):1306–1314. https://doi.org/10.1038/nn.3492 PMID: 23955560

56. Lim S, Goldman MS. Balanced cortical microcircuitry for spatial working memory based on corrective feedback control. Journal of Neuroscience. 2014; 34(20):6790–6806. https://doi.org/10.1523/JNEUROSCI.4602-13.2014 PMID: 24828633

57. Boerlin M, Denève S. Spike-Based Population Coding and Working Memory. PLoS Comput Biol. 2011 2; 7(2):e1001080. https://doi.org/10.1371/journal.pcbi.1001080 PMID: 21379319

58. Hennequin G, Aitchison L, Lengyel M. Fast sampling-based inference in balanced neuronal networks. In: Advances in neural information processing systems; 2014. p. 2240–2248.

59. Hennequin G, Vogels TP, Gerstner W. Optimal control of transient dynamics in balanced networks supports generation of complex movements. Neuron. 2014; 82(6):1394–1406. https://doi.org/10.1016/j.neuron.2014.04.045 PMID: 24945778

60. Haider B, Duque A, Hasenstaub AR, McCormick DA. Neocortical Network Activity In Vivo Is Generated through a Dynamic Balance of Excitation and Inhibition. Journal of Neuroscience. 2006; 26(17):4535–4545. https://doi.org/10.1523/JNEUROSCI.5297-05.2006 PMID: 16641233

61. Litwin-Kumar A, Doiron B. Slow dynamics and high variability in balanced cortical networks with clustered connections. Nature neuroscience. 2012; 15(11):1498. https://doi.org/10.1038/nn.3220 PMID: 23001062

62. Doiron B, Litwin-Kumar A, Rosenbaum R, Ocker GK, Josić K. The mechanics of state-dependent neural correlations. Nature neuroscience. 2016; 19(3):383. https://doi.org/10.1038/nn.4242 PMID: 26906505

63. Seung HS. Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. Neuron. 2003; 40(6):1063–1073. https://doi.org/10.1016/S0896-6273(03)00761-X PMID: 14687542

64. Sporea I, Grüning A. Supervised learning in multilayer spiking neural networks. Neural computation. 2013; 25(2):473–509. https://doi.org/10.1162/NECO_a_00396 PMID: 23148411

65. Nicola W, Clopath C. Supervised learning in spiking neural networks with FORCE training. Nature communications. 2017; 8(1):2208. https://doi.org/10.1038/s41467-017-01827-3 PMID: 29263361

66. DePasquale B, Cueva CJ, Rajan K, Escola GS, Abbott LF. full-FORCE: A target-based method for training recurrent networks. PLOS ONE. 2018 2; 13(2):1–18. https://doi.org/10.1371/journal.pone.0191527 PMID: 29415041

67. Zenke F, Ganguli S. Superspike: Supervised learning in multilayer spiking neural networks. Neural computation. 2018; 30(6):1514–1541. https://doi.org/10.1162/neco_a_01086 PMID: 29652587

68. Eliasmith C, Anderson CH. Neural Engineering Computation, Representation, and Dynamics in Neurobiological Systems. Cambridge, NY, USA: MIT Press; 2002.

69. Eliasmith C. A unified approach to building and controlling spiking attractor networks. Neural Comput. 2005; 17(6):1276–1314. https://doi.org/10.1162/0899766053630332 PMID: 15901399

70. Gerstner W. A Framework for Spiking Neuron Models: The Spike Response Model. In: Moss F, Gielen S, editors. The Handbook of Biological Physics. vol. 4; 2001. p. 469–516.

71. Paninski L, Pillow JW, Simoncelli EP. Maximum Likelihood Estimation of a Stochastic Integrate-and-Fire Neural Model. Neural Computation. 2004; 16:2533–2561. https://doi.org/10.1162/0899766042321797 PMID: 15516273

72. Pillow JW, Paninski L, Uzzell VJ, Simoncelli EP, Chichilnisky EJ. Prediction and Decoding of Retinal Ganglion Cell Responses with a Probabilistic Spiking Model. The Journal of Neuroscience. 2005; 25:11003–11013. https://doi.org/10.1523/JNEUROSCI.3305-05.2005 PMID: 16306413

73. Mensi S, Naud R, Gerstner W. From Stochastic Nonlinear Integrate-and-Fire to Generalized Linear Models. In: NIPS; 2011. p. 1377–1385.

74. Kass RE, Ventura V. A spike-train probability model. Neural computation. 2001; 13(8):1713–1720. https://doi.org/10.1162/08997660152469314 PMID: 11506667

75. Aguera y Arcas B, Fairhall AL. What causes a neuron to spike? Neural Computation. 2003; 15(8):1789–1807. https://doi.org/10.1162/08997660360675044 PMID: 14511513

76. Latimer K, Rieke F, Pillow JW. Inferring synaptic inputs from spikes with a conductance-based neural encoding model. bioRxiv. 2018;p. 281089.

77. Gerstner W, Kistler WM, Naud R, Paninski L. Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition. New York, NY, USA: Cambridge University Press; 2014.

78. Brendel W, Bourdoukan R, P V, Machens CK, Denève S. Learning to represent signals spike by spike. PLoS Computational Biology. 2020;.

79. Pillow JW, Latham P. Neural characterization in partially observed populations of spiking neurons. In: Platt JC, Koller D, Singer Y, Roweis S, editors. Advances in Neural Information Processing Systems 20. Cambridge, MA: MIT Press; 2008. p. 1161–1168.

80. Rezende DJ, Gerstner W. Stochastic variational learning in recurrent spiking networks. Frontiers in Computational Neuroscience. 2014; 8.