

# Research Notes on Self-Coupled Spiking Neural Networks

Chris Fritz

today

## Contents

<b>1</b>	<b>The self-coupled SNN Model</b>	<b>2</b>
<b>2</b>	<b>Derivation of the Basic Model</b>	<b>3</b>
2.1	Predecessor: PCF Voltage Dynamics . . . . .	3
2.2	The Self-Coupled Model is PCF in an Orthonormal Basis . . . . .	6
2.3	Optimizing Spike-Timing: From PCF to Self-Coupled . . . . .	10
2.4	Consequences of Positive Unit-Area Spikes . . . . .	11
2.5	Simulation of Basic Equations . . . . .	23
<b>3</b>	<b>RMSE vs Spike Rate for Constant Driving Force</b>	<b>26</b>
3.1	Ideal Network . . . . .	26
3.2	Neuron Trajectories . . . . .	26
3.3	Solution for the Periodically Spiking Network . . . . .	26
3.4	RMSE for Periodic Spiking . . . . .	27
<b>4</b>	<b>Derivation: The Predictive Coding Framework and Gap-Junction Network</b>	<b>31</b>
4.1	Identical Initial Derivation for All Three Models (PCF, GJ, SC): . . . . .	31
4.2	PCF Derivation: . . . . .	32
4.3	GJ Derivation: . . . . .	33
<b>5</b>	<b>PCF and Gap-Junction Response to Constant Stimulus</b>	<b>34</b>
5.1	PCF Network Response to Constant Stimulus: . . . . .	34
5.2	Gap-Junction Network Response to Constant Stimulus: . . . . .	37
5.3	Comparison of Self-Coupled, Gap-Junction, and PCF Networks for a Constant Stimulus . . . . .	39

# 1 The self-coupled SNN Model

## *Problem Statement*

Given:

- A Linear Dynamical System  $\frac{dx}{dt} = Ax(t) + Bc(t)$ ,  $x \in \mathbf{R}^d$
- A Decoder Matrix  $D \in \mathbf{R}^d \times N$  specifying the preferred directions of  $N$  neurons in  $d$ -dimensional space,

synthesize a spiking neural network that implements the linear dynamical system.

## *Features*

1. ***Long-Term Network Accuracy*** The Deneve network assumes  $\hat{x} = x$ . We show this assumption produces estimation error between the network and its target system that increases with time. By avoiding this assumption, the self-coupled network remains accurate over time.
2. ***Tuning Curve Rotation*** To most efficiently use neurons, we use orthogonal coding directions via SVD. The dynamics matrix  $A$  is diagonalized by an orthonormal basis  $\mathcal{U}$  in  $d$ -dimensional space, while the decoder matrix  $D$  is chosen such that  $\mathcal{U}$  gives its left singular vectors. This choice of coding directions eliminates connectivity between neurons with orthogonal encoding directions.  
At least two neurons per dimension (2d in total) are required since spikes, the encoding quanta, have positive unit-area.  $N$ -neuron ensembles can thus represent systems with  $\frac{N}{2}$  dimensions or less.
3. ***Post-synaptic Spike Dropping*** At each synapse, neurotransmitter release due to an action potential is probabilistic. We incorporate probabilistic spike transmission by thinning at every synaptic connection. The pre-synaptic neuron's membrane potential is still deterministically reset by an action potential.
4. ***Dimensionless Time*** We describe both the network and target system in dimensionless time. Time is normalized by the synapses' time constant,  $\tau_s$ . This dimensionless representation ensures consistent numerical simulation independent of simulation timestep. Furthermore,  $\tau_s$  is implicitly specified as 1, reducing the model's parameters by one.

## 2 Derivation of the Basic Model

### 2.1 Predecessor: PCF Voltage Dynamics

1. Let  $\tau_s$  be the synaptic time constant of each synapse in the network. Define dimensionless time as:

$$\xi \triangleq \frac{t}{\tau_s}.$$

We now assume our Linear Dynamical System is expressed in dimensionless time, i.e

$$\frac{dx}{d\xi} = Ax(\xi) + Bc(\xi). \quad (2.1)$$

To describe the neuron dynamics in dimensionless time, let  $o(\xi) \in \mathbf{R}^N$  be the spike trains of N neurons composing the network with components

$$o_j(\xi) = \sum_{k=1}^{n_j \text{ spikes}} \delta(\xi - \xi_j^k),$$

where  $\xi_j^k$  is the time at which neuron  $j$  makes its  $k^{th}$  spike. Define the network's estimate of the state variable as

$$\hat{x}(\xi) \triangleq Dr(\xi), \quad (2.2)$$

where  $D \in \mathbf{R}^{d \times N}$  and

$$\frac{dr}{d\xi} = -r + o(\xi). \quad (2.3)$$

When the probability of synaptic transmission is 1, component  $r_j$  is the total received post-synaptic current (PSC) from neuron  $j$  by the network estimator. Define the network error as

$$e(\xi) \triangleq x(\xi) - \hat{x}(\xi). \quad (2.4)$$

2. From equations (2.3) and (2.2), we have

$$D\dot{r} + Dr = Do$$

$$\implies \dot{\hat{x}} + \hat{x} = Do,$$

where the dot denotes derivative w.r.t dimensionless time  $\xi$ .

Subtract  $\dot{\hat{x}}$  from  $\dot{x}$  to get  $\dot{e}$ :

$$\begin{aligned} \dot{e} &= \dot{x} - \dot{\hat{x}} \\ &= (Ax + Bc) - (Do - \hat{x}) \\ &= A(e + \hat{x}) + Bc - Do + \hat{x} \\ &= Ae + (A + I)\hat{x} + Bc - Do \\ &= Ae + (A + I)(Dr) + Bc - Do \\ \implies D^T \dot{e} &= D^T Ae + D^T(A + I)(Dr) + D^T Bc - D^T Do. \end{aligned}$$

The quantity  $D^T e$  defines the membrane voltage of the predictive coding framework (PCF), a precursor to this model:

$$v_{pcf} \triangleq D^T e.$$

Note that the definition implies  $e = D^{T\dagger} v_{pcf}$ . The voltage dynamics are thus

$$\dot{v}_{pcf} = D^T A D^{T\dagger} v_{pcf} + D^T (A + I) (Dr) + D^T Bc - D^T Do, \quad (2.5)$$

where  $D^{T\dagger}$  is the left pseudo-inverse of  $D^T \in \mathbf{R}^{N \times d}$ . The PCF thus defines a mapping between two vector spaces: the d-dimensional state space of the target system, and the N-dimensional voltage space of the spiking neural network. This mapping is visualized in figure (1).

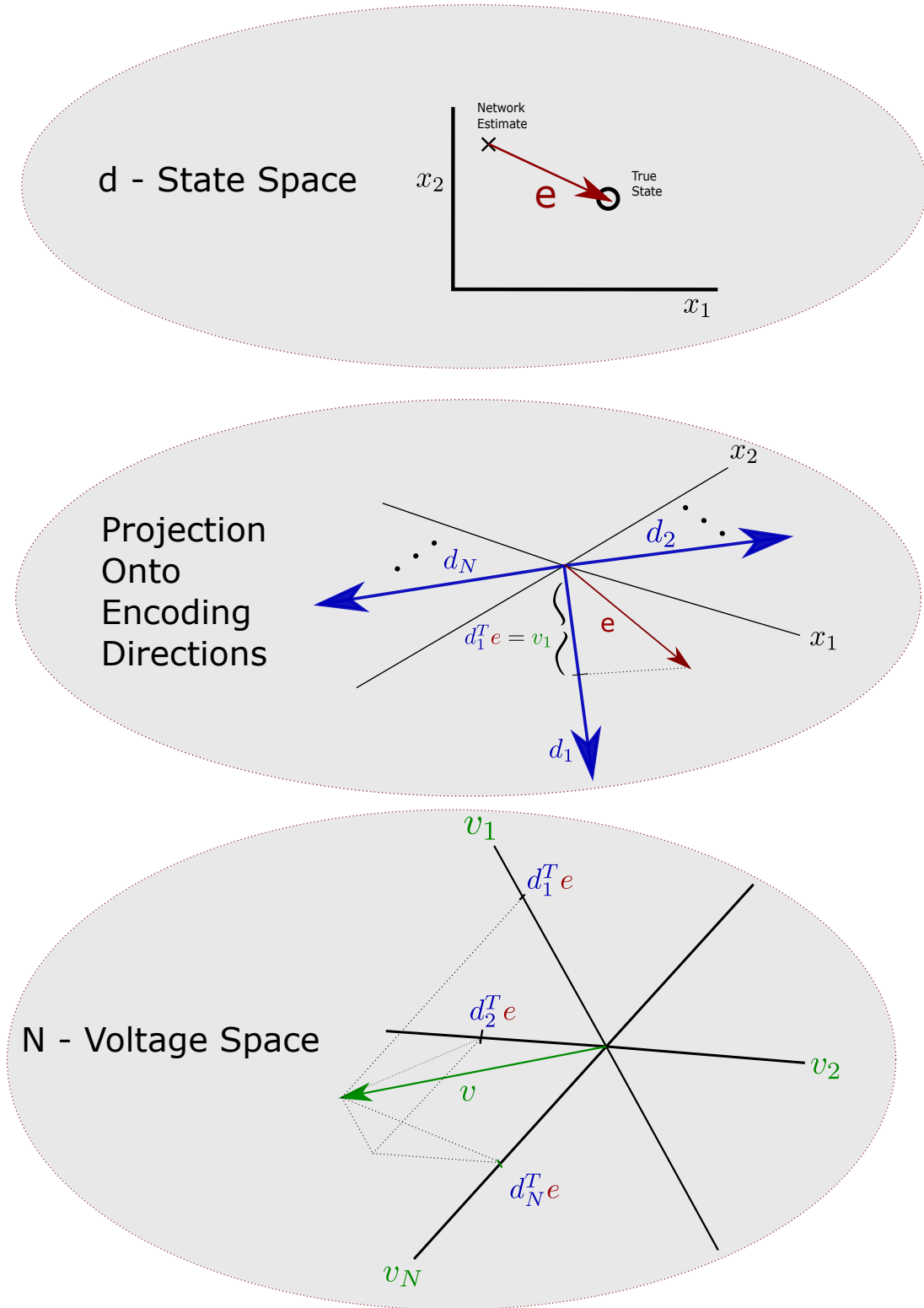


Figure 1: Mapping Between State and Voltage Spaces: **Top:** The estimation error  $e$  is computed by comparing the decoded network estimate to the true state of the target dynamical system. **Middle:** The  $e$  is projected onto the encoding directions of the neurons composing the network. The projection of error onto encoding direction  $j$  gives the membrane voltage of neuron  $j$ ,  $v_j = d_j^T e$ . **Bottom:** The voltages form a  $N$ -dimensional vector contained in voltage space. 5

## 2.2 The Self-Coupled Model is PCF in an Orthonormal Basis

1. The self-coupled network is derived from the PCF via a change of bases. Assuming both  $D$  and  $A$  are full rank, diagonalize each to a common left basis:

$$A = \mathcal{U} \Lambda \mathcal{U}^T = \sum_{j=1}^d \Lambda_j \mathcal{U}_j \mathcal{U}_j^T,$$

$$D = \mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T = \sum_{j=1}^d S_j \mathcal{U}_j V_j^T,$$

$$D^T = V \begin{bmatrix} S \\ 0 \end{bmatrix} \mathcal{U}^T = \sum_{j=1}^d S_j V_j \mathcal{U}_j^T,$$

$$D^T D = V \begin{bmatrix} S \\ 0 \end{bmatrix} \begin{bmatrix} S & 0 \end{bmatrix} V^T = \sum_{j=1}^d S_j^2 V_j V_j^T,$$

with  $\mathcal{U} \in \mathbf{R}^{d \times d}$  and  $V \in \mathbf{R}^{N \times N}$ , and  $S \in \mathbf{R}^{d \times d}$ .

In the original basis, the state is  $x$ . In the rotated basis we denote this quantity as  $y$ . It is the projection of  $x$  onto the  $d$ -dimensional  $\mathcal{U}$  basis:

$$y \triangleq \mathcal{U}^T x \tag{2.6}$$

The rotated target dynamics are thus

$$\begin{aligned} \dot{y} &= \mathcal{U}^T \dot{x} \\ &= \Lambda y(\xi) + \mathcal{U}^T B c(\xi) \\ &= \Lambda y(\xi) + \mathcal{U}^T B \mathcal{U} \mathcal{U}^T c(\xi) \\ &= \Lambda y(\xi) + \beta \tilde{c}(\xi) \end{aligned} \tag{2.7}$$

where

$$\beta \triangleq \mathcal{U}^T B \mathcal{U},$$

and

$$\tilde{c} \triangleq \mathcal{U}^T c,$$

give the projections of  $B$  and  $c$  respectively. The network estimate in the rotated basis is

$$\hat{y} \triangleq \mathcal{U}^T \hat{x}.$$

From equation (2.2),

$$\begin{aligned}
\hat{y} &= \mathcal{U}^T \hat{x} \\
&= \mathcal{U}^T D r \\
&= [S \quad 0] V^T r \\
&= [S \quad 0] \rho \\
\implies \dot{\hat{y}} &= [S \quad 0] V^T \dot{r} \\
&= [S \quad 0] (-V^T r + V^T o) .
\end{aligned}$$

Note that  $V^T r$  and  $V^T o$  are projections of the N-neuron network's post-synaptic current and spike train respectively onto the rotated basis, denoted by

$$\rho \triangleq V^T r, \quad (2.8)$$

$$\tilde{o} \triangleq V^T o. \quad (2.9)$$

The preceding equality also gives  $\hat{y}$  in terms of  $\rho$ :

$$\hat{y} = [S \quad 0] \rho. \quad (2.10)$$

With these definitions, the last equality above also implies

$$\dot{\rho} = -\rho + \tilde{o}. \quad (2.11)$$

To finish describing the basic network quantities in terms of the rotated basis, let  $\epsilon$  be the error in the rotated basis:

$$\begin{aligned}
\epsilon &\triangleq y - \hat{y} \\
&= \mathcal{U}^T e.
\end{aligned} \quad (2.12)$$

2. Repeat the derivation of equation (2.5) but with  $y$ ,  $\hat{y}$ , and  $\epsilon$ :

$$\begin{aligned}
\dot{\epsilon} &= \dot{y} - \dot{\hat{y}} \\
&= \Lambda y + \beta c - [S \ 0] (-\rho + \tilde{o}) \\
&= \Lambda (\epsilon + [S \ 0] \rho) + \beta \tilde{c} - [S \ 0] (-\rho + \tilde{o}) \\
&= \Lambda \epsilon + (\Lambda + I) [S \ 0] \rho + \beta \tilde{c} - [S \ 0] \tilde{o} \\
\Rightarrow \begin{bmatrix} S \\ 0 \end{bmatrix} \dot{\epsilon} &= \begin{bmatrix} S \\ 0 \end{bmatrix} \Lambda \epsilon + \begin{bmatrix} S \\ 0 \end{bmatrix} (\Lambda + I) [S \ 0] \rho + \begin{bmatrix} S \\ 0 \end{bmatrix} \beta \tilde{c} - \begin{bmatrix} S \\ 0 \end{bmatrix} [S \ 0] \tilde{o}.
\end{aligned}$$

The last equality gives a system of  $N$  equations of which only  $d$  are nontrivial. A comparison with equation (2.5) suggests the  $N$ -dimensional rotated membrane potential  $v$  is best defined as:

$$v \triangleq \begin{bmatrix} S \\ 0 \end{bmatrix} \epsilon \in \mathbf{R}^N. \quad (2.13)$$

This mapping is invertible from  $v$  to  $\epsilon$  if we write

$$\epsilon = [S^{-1} \ 0].$$

This gives a well-defined  $d$ -vector, whose first  $d$  elements are well defined, and the remaining components of  $v$  are assumed to be zero. Using a similar abuse for the  $\rho$  and  $\tilde{o}$  terms, we arrive at the system of  $N$  equations describing the network voltage dynamics:

$$\dot{v} = \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} v + \begin{bmatrix} S(\Lambda + I_d)S & 0 \\ 0 & 0 \end{bmatrix} \rho + \begin{bmatrix} S \\ 0 \end{bmatrix} \beta \tilde{c} - \begin{bmatrix} S^2 & 0 \\ 0 & 0 \end{bmatrix} \tilde{o}. \quad (2.14)$$

To summarize conceptually, there are 4 vector spaces in total: the error space which tracks the dynamical system and the network estimate, the voltage space which tracks the membrane potentials, and the transformed counterparts of each in the  $\mathcal{U} - V$  bases. Figure (2) shows the relationships derived between these subspaces.



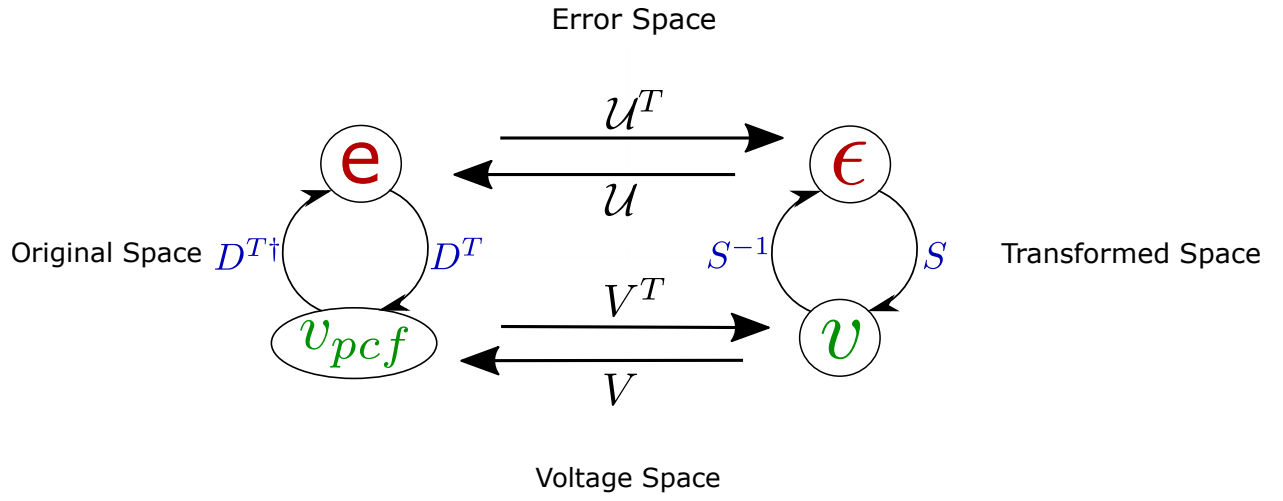


Figure 2: Depiction of the relationship between original and transformed spaces and their respective error and voltage spaces. An arrow represents left multiplication by the given matrix. The zeros in the full  $N \times N$  matrices mapping between  $v$  and  $\epsilon$  are omitted for clarity.

## 2.3 Optimizing Spike-Timing: From PCF to Self-Coupled

1. In PCF, the spike trains  $o$  are chosen minimize the network estimation error

$$\mathcal{L}(\xi) = \|x(\xi + d\xi) - \hat{x}(\xi + d\xi)\|^2. \quad (2.15)$$

The network greedily minimizes  $\mathcal{L}(\xi)$  an instant  $d\xi$  ahead in time. If no spike occurs at time  $\xi$ , then the objective is given above. If neuron  $j$  spikes, the estimate  $\hat{x} \leftarrow \hat{x} + d_j$ , where  $d_j$  is column  $j$  of  $D$ . The objective is now

$$\begin{aligned} \mathcal{L}_{sp}(\xi) &= \|x - (\hat{x} + d_j)\|^2 \\ &= x^T x - 2x^T \hat{x} - 2x^T d_j + \hat{x}^T \hat{x} + 2\hat{x}^T d_j + d_j^T d_j \\ &= x^T x - 2x^T \hat{x} + \hat{x}^T \hat{x} - 2d_j^T (x - \hat{x}) + d_j^T d_j \\ &= \|x - \hat{x}\|^2 - 2d_j^T (x - \hat{x}) + d_j^T d_j \\ &= \mathcal{L}_{ns}(\xi) - 2d_j^T (x - \hat{x}) + d_j^T d_j, \end{aligned}$$

where  $\mathcal{L}_{ns}(\xi)$  is the objective if no spike occurs. Spiking occurs when the objective decreases or

$$\mathcal{L}_{sp} < \mathcal{L}_{ns}$$

$$\implies -2d_j^T (x - \hat{x}) + d_j^T d_j < 0$$

$$\implies d_j^T (x - \hat{x}) > \frac{\|d_j\|^2}{2}.$$

Since  $d_j^T (x - \hat{x}) = d_j^T e$  is already defined as membrane voltage, the right hand side gives neuron  $j$ 's spike threshold voltage  $v_{th}$ ,

$$v_{th}^{pcf} = \frac{1}{2} \begin{bmatrix} d_1^T d_1 \\ \vdots \\ d_N^T d_N \end{bmatrix}.$$

2. For the rotated network, note  $\mathcal{U}^T$  is an orthonormal matrix by definition. Thus it is norm-preserving:

$$\begin{aligned} \mathcal{L}_{sp}(\xi) &= \|x - \hat{x}\|^2 \\ &= \|\mathcal{U}^T (x - \hat{x})\|^2 \\ &= \|y - \hat{y}\|^2. \end{aligned}$$

If we define the rotated network objective as

$$\tilde{\mathcal{L}}(\xi) \triangleq \|y(\xi + d\xi) - \hat{y}(\xi + d\xi)\|^2,$$

it is equal to the original network objective when no spike occurs. However, a spike alters the readout by  $\hat{y} \leftarrow \hat{y} + S_l$ , where  $S_l$  is the  $l^{th}$  column of  $[S \ 0]$ . With the same approach as above, the objective when neuron  $l$  spikes is

$$\begin{aligned}\tilde{L}_{sp} &= \tilde{L}_{ns} + 2S_l^T \epsilon + S_l^T S_l \\ \implies v_l &> \frac{\|S_l\|^2}{2}.\end{aligned}$$

This leads to voltage thresholds

$$v_{th} = \frac{1}{2} \begin{bmatrix} S_1^T S_1 \\ \vdots \\ S_d^T S_d \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

## 2.4 Consequences of Positive Unit-Area Spikes

1. The voltage thresholds are nonnegative, such that neuron  $j$  will only spike if

$$v_l = S_l^T \epsilon > v_{th} > 0.$$

The spike from neuron  $l$  corrects the network error  $\epsilon$  by adding  $S_l$  to the estimate  $\hat{y}$ . The nonnegative voltage implies it is impossible for neuron  $l$  to correct antiparallel errors ( $-S_l$ ), since

$$S_l^T(\epsilon) = S_l^T(-S_l) = -\|S_l\|^2 < 0 < v_{th}.$$

To illustrate consider the space of errors  $\epsilon \in \mathbf{R}^d$  which satisfy the voltage threshold of neuron  $l$ , i.e

$$\epsilon_{sp} = \{\epsilon \in \mathbf{R}^d \mid S_l^T \epsilon > v_{th}\}.$$

In  $\mathbf{R}^2$ ,  $\epsilon_{sp}$  is the half-plane formed by the line normal to  $S_l$  shifted  $v_{th}$  from the origin along  $S_l$  as in figure (3). This excludes  $-S_l$ . The optimization thus tells us that neuron  $l$  spikes when the projection  $S_l^T \epsilon$  exceeds  $v_{th}$ .

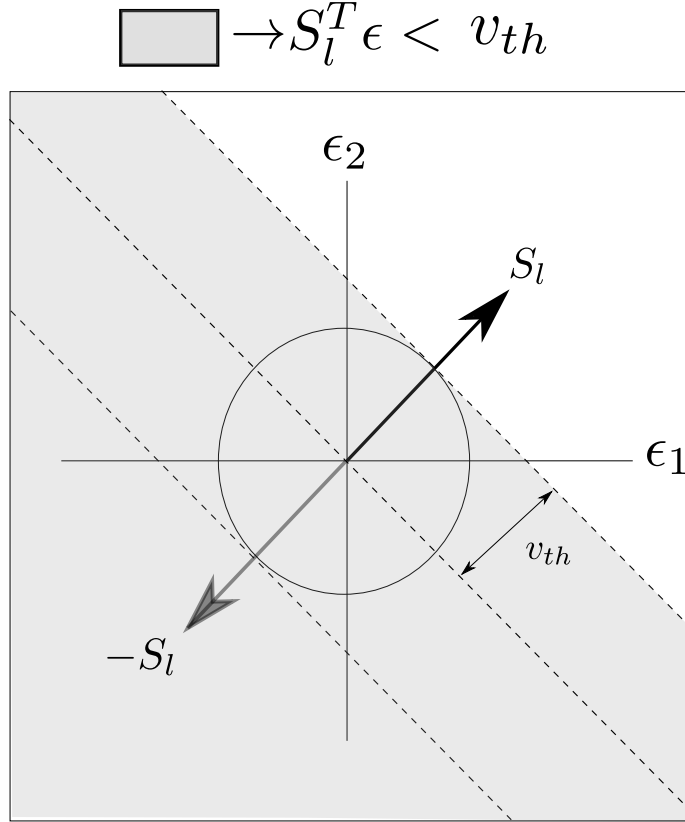


Figure 3: Sketch of the Encoding space  $\epsilon_{sp}$  of neuron  $l$  with direction  $S_l \in \mathbf{R}^2$ . The radius of the circle is  $v_{th} = \frac{\|S_l\|^2}{2}$ . (Vector  $S_l$  not drawn to scale).

2. Equations (2.14) and (2.11) describe how we implement a network with  $d$  neurons that produces an estimate  $\hat{y}$  of the given target system. As written, the network can only encode vectors  $y$  with strictly nonnegative elements. To see why we return to the optimization procedure performed by the network. The network optimizes the objective

$$\mathcal{L} = \|y - \hat{y}\|,$$

by choosing spike times  $\tilde{o}$ . The spikes are integrated into a post-synaptic feedback  $\rho$ . This feedback vector is scaled by  $S$  to generate the estimate  $\hat{y}$  that minimizes the objective at time  $\xi$ . In other words, the network performs the optimization

$$\min_{\rho \in \mathbf{R}^{d+}} \|y - S\rho\|^2,$$

where  $\mathbf{R}^{d+}$  denotes the real  $d$ -vectors with nonnegative components. The components must be non-negative because the spikes  $\tilde{o}$  have nonnegative area when integrated, and their dynamics will not decay below zero otherwise. In using a greedy approach with only one spike at a given time step, the network more specifically performs

$$\min_{x \in \mathbf{Z}^{d+} : \sum_j Z_j = 1} \|y - S(\rho + x)\|^2.$$

I.e, it must choose one neuron to spike with unit area 1, which adds precisely one column of  $S$  to the estimate  $\hat{y}$ .

It is easier to analyze the former optimization of  $\rho \in \mathbf{R}^{d+}$ , and we do so here. Because  $\{x \in \mathbf{Z}^{d+} : \sum_j Z_j = 1\} \subset \mathbf{R}^{d+}$ , it is always the case that

$$\min_{\rho \in \mathbf{R}^{d+}} \|y - S\rho\|^2 \leq \min_{x \in \mathbf{Z}^{d+} : \sum_j Z_j = 1} \|y - S(\rho + x)\|^2,$$

i.e optimizing over arbitrary  $\rho \in \mathbf{R}^{d+}$  will always give just as low or lower objectives than under the single-greedy spike optimization.

We're interested in the range of vectors representable by the network. That is the set

$$X^* = \{x \in \mathbf{R}^{d+} : Sx = y\}.$$

Over this set, the objective function is 0, i.e.

$$X^* = \{x \in \mathbf{R}^{d+} : \mathcal{L} = \|y - Sx\|^2 = 0\}.$$

Let  $x \in X^*$ , and consider its negative  $-x$ . It follows that

$$\begin{aligned} x &= S\rho \\ \implies -x &= -S\rho \\ &= S(-\rho). \end{aligned}$$

However if  $\rho \in \mathbf{R}^{d+}$ , then it is impossible for  $-\rho \in \mathbf{R}^{d+}$  to also be true. Thus  $-\rho \notin X^*$  so that  $\mathcal{L} > 0$ . We conclude that for any vector  $\hat{y}$ , the network can represent with  $\mathcal{L} = 0$ , there exists a

negative vector that the network cannot represent with  $\mathcal{L} = 0$ . This is undesirable as it restricts the set of vectors the network can reconstruct within a given error tolerance. In  $\mathbf{R}^2$  for example, network representation where  $\mathcal{L} = 0$  is restricted to the first quadrant. This restriction applies equally to the greedy single-spike optimization.

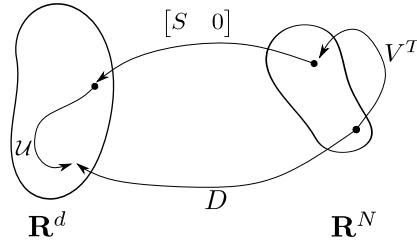
This issue is unique to the self-coupled network and does not occur in the original PCF network even though its spikes must also have positive unit area. The difference arises when we take the SVD of the decoder matrix.

$$D = \mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T.$$

The SVD decomposes  $D$  into orthonormal bases  $\mathcal{U}$  and  $V$  which are mapped to one another by singular values  $S$ , as in figure (4). By rotating into the  $\mathcal{U} - V$  bases, we preemptively perform the first and last mappings, leaving only multiplication by a diagonal matrix. This eliminates linearly dependent encoding vectors, keeping only the orthonormal. For example, suppose  $x = Dy = \mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T y$ . For an orthonormal basis,  $-x$  is obtainable by  $\mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T (-y)$ . However the constraint that spikes have positive unit area prevents a vector  $\rho = -y$  from being reachable by the network as written. We consider two options to address this below.

$$D : \mathbf{R}^N \rightarrow \mathbf{R}^d = \mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T$$

Original Matrix



Rotate to  $\mathcal{U} - V$  basis

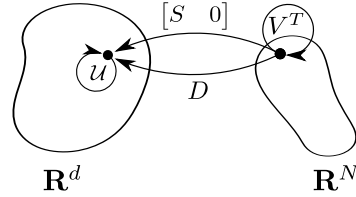


Figure 4: Visualizing  $D$  as a sequence of linear maps between subspaces. **Top:** The matrix  $D \in \mathbf{R}^{d \times N}$  is decomposed via SVD into a sequence of 3 linear maps (matrices). The rightmost matrix  $V^T \in \mathbf{R}^{N \times N}$  projects a vector  $x$  to give coefficients for the expansion in the basis  $V$ . The center matrix  $\begin{bmatrix} S & 0 \end{bmatrix} \in \mathbf{R}^{d \times N}$  maps vectors from the  $V$  basis to a vector in  $\mathbf{R}^d$  by scaling and truncation. The leftmost matrix  $\mathcal{U} \in \mathbf{R}^{d \times d}$  gives the resultant vector  $Dx \in \mathbf{R}^d$  by using the scaled vector  $\begin{bmatrix} S & 0 \end{bmatrix} V^T$  as coefficients for a basis expansion in  $\mathcal{U}$ . **Bottom:** We rotate the basis for vectors in  $\mathbf{R}^N$  and  $\mathbf{R}^d$  to the  $\mathcal{U}$  and  $V$  bases respectively. This negates the need of  $D$  to preemptively project and afterward rotate a vector, leaving only scaling by a diagonal matrix. The mapping  $D$  performs on a vector  $y$  simplifies to multiplication by a diagonal matrix  $S$  of  $y$ 's first  $d$  components.

3. For the network decoder to fully span the state space of interest, it must have anti-parallel encoding directions even in the rotated orthonormal basis. To do so we augment the rotated bases by adding an antiparallel set so they are no longer orthonormal. We consider two methods of adding antiparallel bases:

**Method A: Separate Antiparallel Networks** One solution is to form a separate network of  $d$  neurons whose encoding directions are the antiparallel set  $-\mathcal{U}$ . That is, we form an identical network except the decode matrix is

$$-D = -\mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T.$$

We then add the output of the two networks to recover the encoded state.

We divide the error into its positive and negative components and encode each in a separate network. Let

$$\epsilon^+ = \epsilon \geq 0,$$

be the nonnegative component of  $\epsilon$ . Note that the original estimate,  $e$  may contain negative or positive components, but the projection  $\epsilon^+$  does not. Similarly define the negative error by

$$\epsilon^- = -\epsilon < 0.$$

Note

$$\epsilon^- \neq -\epsilon^+.$$

Rather,

$$\epsilon = \epsilon^+ - \epsilon^-.$$

The relationship between  $\epsilon^+$  and  $\epsilon^-$  resembles two orthogonal subspaces. The preceding relation is analogous to a direct sum.

Let

$$v^+ = S^T \epsilon^+,$$

be the voltage induced by projecting  $\epsilon^+$  onto the orthonormal bases given by  $D = \mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T$ ,

and

$$v^- = S^T \epsilon^-$$

be the respective projection onto the antiparallel orthonormal bases,  $D = \mathcal{U} \begin{bmatrix} -S & 0 \end{bmatrix} V^T$ .

Note that

$$v^- \neq -v^+.$$

Rather,

$$v = v^+ - v^-,$$

where  $v$  is the voltage of an idealized neuron capable of positive and negative area spikes. Note that both  $v_j^+$  and  $v_j^-$  are bounded by thresholds  $v_{th} = \frac{\|S_j\|^2}{2}$ , so that the idealized neuron is always within the voltage range  $v \in [-v_{th}, v_{th}]$ . This is equivalent to asserting the error along each encoding direction  $S_j$  is contained within the polytope  $S_j^T \epsilon \leq \frac{\|S_j\|^2}{2}$ .

Let  $\rho^+$  and  $\tilde{o}^+$  be the slow synaptic feedback and spike trains of the positive neurons, with  $\rho^-$  and  $\tilde{o}^-$  defined similarly for the negative neurons. Finally split  $\tilde{c} = \tilde{c}^+ - \tilde{c}^-$  into positive and negative components as with  $\epsilon$ .



We now have two  $d$ -dimensional systems of equations.

$$\dot{v}^+ = \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} v^+ + \begin{bmatrix} S(\Lambda + I_d)S & 0 \\ 0 & 0 \end{bmatrix} \rho^+ + \begin{bmatrix} S \\ 0 \end{bmatrix} \beta \tilde{c}^+ - \begin{bmatrix} S^2 & 0 \\ 0 & 0 \end{bmatrix} \tilde{o}^+,$$

$$\dot{v}^- = \begin{bmatrix} \Lambda & 0 \\ 0 & 0 \end{bmatrix} v^- + \begin{bmatrix} S(\Lambda + I_d)S & 0 \\ 0 & 0 \end{bmatrix} \rho^- + \begin{bmatrix} S \\ 0 \end{bmatrix} \beta \tilde{c}^- - \begin{bmatrix} S^2 & 0 \\ 0 & 0 \end{bmatrix} \tilde{o}^-.$$

These equations each produce estimates

$$\hat{y}^+ = S\rho^+,$$

$$\hat{y}^- = S\rho^-,$$

which give the network estimate

$$\hat{y} = \hat{y}^+ - \hat{y}^-.$$

Writing the above as a single network, assume  $N = 2d$  so we need not fill with zeros:

$$\begin{bmatrix} \dot{v}^+ \\ \dot{v}^- \end{bmatrix} = \begin{bmatrix} \Lambda & 0 \\ 0 & \Lambda \end{bmatrix} \begin{bmatrix} v^+ \\ v^- \end{bmatrix} + \begin{bmatrix} S(\Lambda + I_d)S & 0 \\ 0 & S(\Lambda + I_d)S \end{bmatrix} \begin{bmatrix} \rho^+ \\ \rho^- \end{bmatrix} + \begin{bmatrix} S & 0 \\ 0 & S \end{bmatrix} \beta \begin{bmatrix} \tilde{c}^+ \\ \tilde{c}^- \end{bmatrix} - \begin{bmatrix} S^2 & 0 \\ 0 & S^2 \end{bmatrix} \begin{bmatrix} \tilde{o}^+ \\ \tilde{o}^- \end{bmatrix}.$$

We simplify this by writing

$$\dot{v} = \Lambda v + S(\Lambda + I_{2d})S\rho + S\beta\tilde{c} - S^2\tilde{o},$$

where we have made the following substitutions:

$$v \leftarrow \begin{bmatrix} v^+ \\ v^- \end{bmatrix} \in \mathbf{R}^{2d},$$

$$\Lambda \leftarrow \begin{bmatrix} \Lambda & 0 \\ 0 & \Lambda \end{bmatrix} \in \mathbf{R}^{2d \times 2d},$$

$$S \leftarrow \begin{bmatrix} S & 0 \\ 0 & S \end{bmatrix} \in \mathbf{R}^{2d \times 2d},$$

$$\rho \leftarrow \begin{bmatrix} \rho^+ \\ \rho^- \end{bmatrix} \in \mathbf{R}^{2d},$$

$$\tilde{o} \leftarrow \begin{bmatrix} \tilde{o}^+ \\ \tilde{o}^- \end{bmatrix} \in \mathbf{R}^{2d},$$

$$\beta \leftarrow \begin{bmatrix} \beta & 0 \\ 0 & \beta \end{bmatrix} \in \mathbf{R}^{2d \times 2d},$$

$$\tilde{c} \leftarrow \begin{bmatrix} \tilde{c}^+ \\ \tilde{c}^- \end{bmatrix} \in \mathbf{R}^{2d},$$

$$v_{th} \leftarrow \begin{bmatrix} v_{th} \\ v_{th} \end{bmatrix} \in \mathbf{R}^{2d}.$$

To decode from the network to the  $d$ -dimensional estimate, we multiply by  $\begin{bmatrix} \mathcal{U} & -\mathcal{U} \end{bmatrix} \in \mathbf{R}^{d \times 2d}$ , i.e

$$\hat{y} = \hat{y}^+ - \hat{y}^- = \begin{bmatrix} \mathcal{U} & -\mathcal{U} \end{bmatrix} \rho.$$

This approach suggests a balance between excitatory and inhibitory neurons when coding an oscillatory signal that inhabits the full  $d$ -dimensional state space. To illustrate, consider a 2 neuron network as in figure (5). An input  $\tilde{c}(\xi) = \sin(\omega\xi)$  drives the neurons which encode  $v^+$  and  $v^-$  respectively. A readout neuron performs leaky integration of the spike trains from the two driven neurons. Consider how a received spike changes the voltage of the readout neuron. A spike from one neuron will increase the readout neuron’s membrane potential (excitatory input), while a spike from the other neuron must symmetrically decrease the readout neuron’s potential (inhibitory input). In this case, we observe equal levels of excitatory (and inhibitory input from the two neurons, suggesting a tight balance. Note also this ensures consistency with Dale’s law, which states that a neuron cannot both excite and inhibit other neurons.

Note also that the fast coupling matrix preceding  $\tilde{o}$  is diagonal. This implies that when a neuron  $j$  spikes, its antiparallel neuron is unchanged. In the PCF, a neuron spike resets its threshold to  $-v_{th}$ , but likewise sets a neuron antiparallel to it to  $v_{th}$ . Next, the antiparallel neuron spikes and likewise resets the neuron. This cycle repeats itself causing the network estimate to oscillate uncontrollably in a catastrophic network failure termed "ping-ponging". The PCF addresses this through regularization terms applied to the network objective  $\mathcal{L}$  and added noise, (Boerlin 2013) both of which are unnecessary in this case.

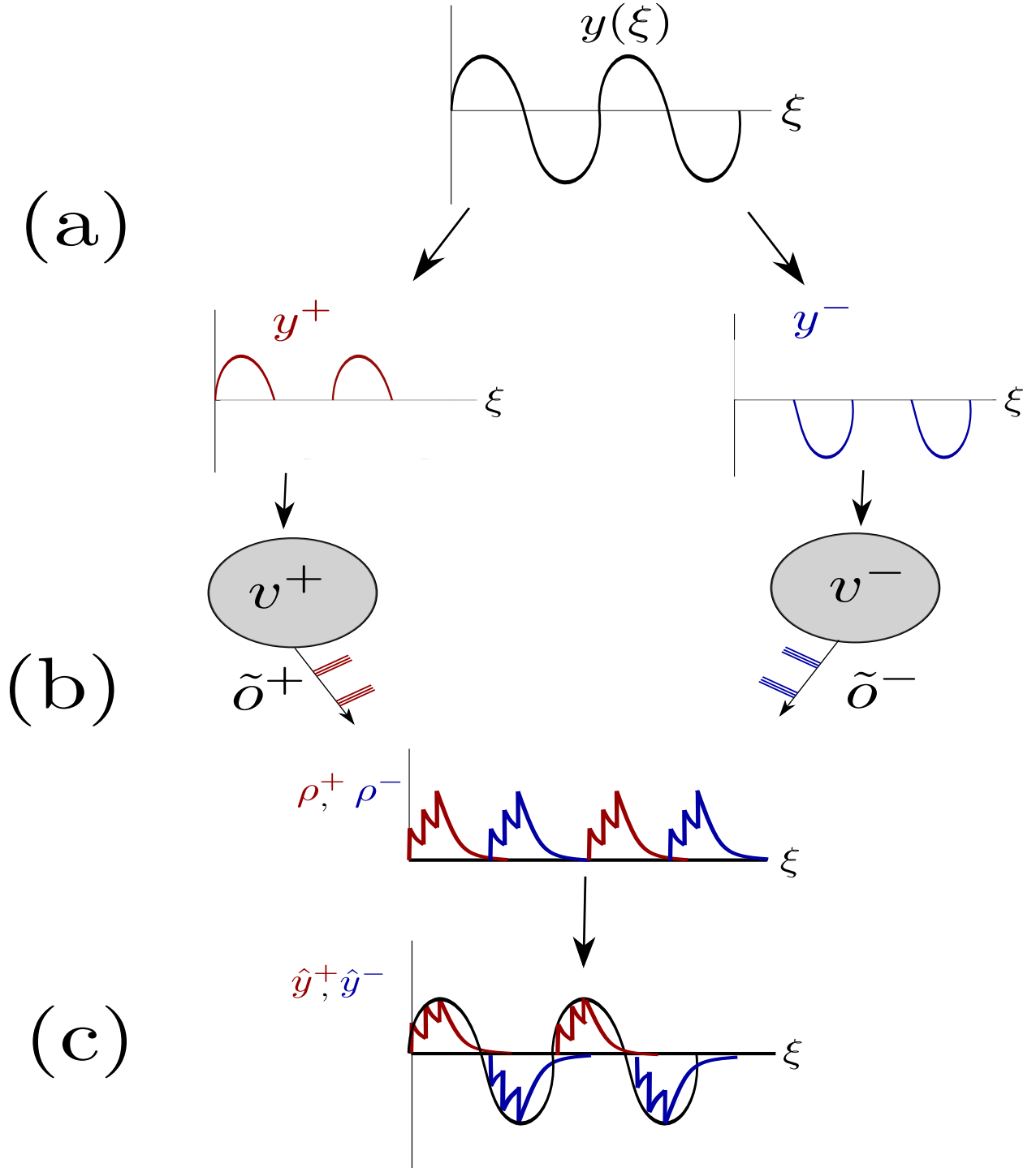


Figure 5: Balance Between Excitatory and Inhibitory Activity for Oscillatory Input. *(a)*: A sinusoidal input  $y$  is divided into its positive (excitatory) and negative (inhibitory) components. *(b)*: Each neuron encodes its respective input by spiking to produce a nonnegative filtered spike train  $\rho$ . *(c)*:

The network estimate is the sum of activity from excitatory and inhibitory filtered spike trains. For oscillatory input, both neurons must spike in equal amounts so that the amplitude of oscillation remains bounded.

**Method B: Fast-Coupling Between Antiparallel Neurons** The other method we consider maintains the fast-coupling relationship between a neuron and its opposite as described in PCF. Fast-coupling implies that when a neuron  $i$  with encoding direction  $d_i$  spikes, neuron  $j$ 's voltage changes by  $-d_j^T d_i$ . This is visible in the spiking ( $D^T D o$ ) term in PCF's voltage equation (2.5). If  $d_j = -d_i$ , then a spike in  $d_j$  decreases  $j$ 's voltage by  $\|d_j\|^2$ , while simultaneously increasing neuron  $i$ 's voltage by the same.

1. Previously we used a subset of  $d$  neurons from the decoder matrix,

$$D = \mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T \in \mathbf{R}^{d \times N}.$$

In adding  $d$  antiparallel neurons, we assume  $N \geq 2d$  and instead use  $2d$  neurons, i.e. we write

$$D = [\mathcal{U} \quad -\mathcal{U}]^T \begin{bmatrix} S & 0 & 0 \\ 0 & S & 0 \end{bmatrix} V^T \in \mathbf{R}^{d \times N}.$$

The first and second  $d$  neurons have encoding directions  $\mathcal{U}$  and  $-\mathcal{U}$  respectively. Before the first  $d$  right eigenvectors  $V_j$  were scaled by  $\sigma_j$  and mapped to  $\mathcal{U}_j$ , with the remaining  $V_j$  mapped to 0. In adding  $d$  antiparallel vectors, we scale an additional  $d$  right eigenvectors  $V_{j+d}$  by  $\sigma_j$  and map them to  $-\mathcal{U}_j$ .

The dynamical system remains the same, i.e.

$$\dot{y} = \Lambda y + \beta \tilde{c}.$$

2. The network estimation

$$\hat{y} = \mathcal{U}^T D \hat{x} = \begin{bmatrix} S & 0 \end{bmatrix} \rho$$

is now

$$\begin{aligned} \hat{y} &= \mathcal{U}^T [\mathcal{U} \quad -\mathcal{U}] \begin{bmatrix} S & 0 & 0 \\ 0 & S & 0 \end{bmatrix} \rho \\ &= \begin{bmatrix} S & -S & 0 \end{bmatrix} \rho. \end{aligned}$$

We rederive the error dynamics

$$\begin{aligned} \dot{\epsilon} &= \dot{y} - \dot{\hat{y}} \\ &= \Lambda y + \beta \tilde{c} - \begin{bmatrix} S & -S & 0 \end{bmatrix} (-\rho + \tilde{o}) \\ &= \Lambda (\epsilon + \hat{y}) + \begin{bmatrix} S & -S & 0 \end{bmatrix} \rho + \beta \tilde{c} - \begin{bmatrix} S & -S & 0 \end{bmatrix} \tilde{o} \\ &= \Lambda \epsilon + (I + \Lambda) \begin{bmatrix} S & -S & 0 \end{bmatrix} \rho + \beta \tilde{c} - \begin{bmatrix} S & -S & 0 \end{bmatrix} \tilde{o} \end{aligned}$$

3. We previously obtained voltage by optimizing the objective

$$\mathcal{L} = \|y - \hat{y}\|^2.$$

For the first  $d$  neurons this was

$$v_j = S_j^T \epsilon \quad \text{if } j \in [1, \dots, d].$$

We now add an addition  $d$  neurons with antiparallel directions. For each neuron  $j$  we add an additional neuron  $j + d$  with opposite encoding direction, and therefore opposite voltage. The voltage is now

$$\begin{cases} v_j = S_j^T \epsilon & \text{for } j \in [1, \dots, d] \\ -S_{j-d}^T \epsilon & \text{for } j \in [d+1, \dots, 2d] \end{cases}.$$

In matrix form,

$$v = \begin{bmatrix} S \\ -S \end{bmatrix} \epsilon.$$

From the error dynamics above, we get

$$\dot{v} = \begin{bmatrix} \Lambda & 0 \\ 0 & \Lambda \end{bmatrix} v + \begin{bmatrix} S \\ -S \end{bmatrix} (I + \Lambda) \begin{bmatrix} S & -S & 0 \end{bmatrix} \rho + \begin{bmatrix} S \\ -S \end{bmatrix} \beta \tilde{c} - \begin{bmatrix} S^2 & -S^2 & 0 \\ -S^2 & S^2 & 0 \end{bmatrix} \tilde{o}. \quad (2.16)$$

Equation (2.16) gives the voltage dynamics of  $2d$  neurons.

4. To facilitate comparison between the self-coupled network and original PCF model, we will use this method (B) hereafter. Method A is a viable alternative, however it is fundamentally different than the original PCF model because antiparallel neurons do not interact in contrast with PCF and method B. We investigate method A later.

## 2.5 Simulation of Basic Equations

Here we simulate the above equations (2.14) and (2.11) with the  $N = 2d$  neurons. The parameters are

$$\begin{aligned}
 A &= - \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \mathcal{U} \Lambda \mathcal{U}^T, \\
 B &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \\
 c(\xi) &= \begin{bmatrix} \cos(\frac{\pi}{4}\xi) \\ \sin(\frac{\pi}{4}\xi) \end{bmatrix} \\
 D &= \mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T = \mathcal{U} \begin{bmatrix} .1 I_d & 0 \end{bmatrix} I_N, \\
 d\xi &= 10^{-6}, \\
 N &= 4, \\
 x(0) &= \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix}.
 \end{aligned} \tag{2.17}$$

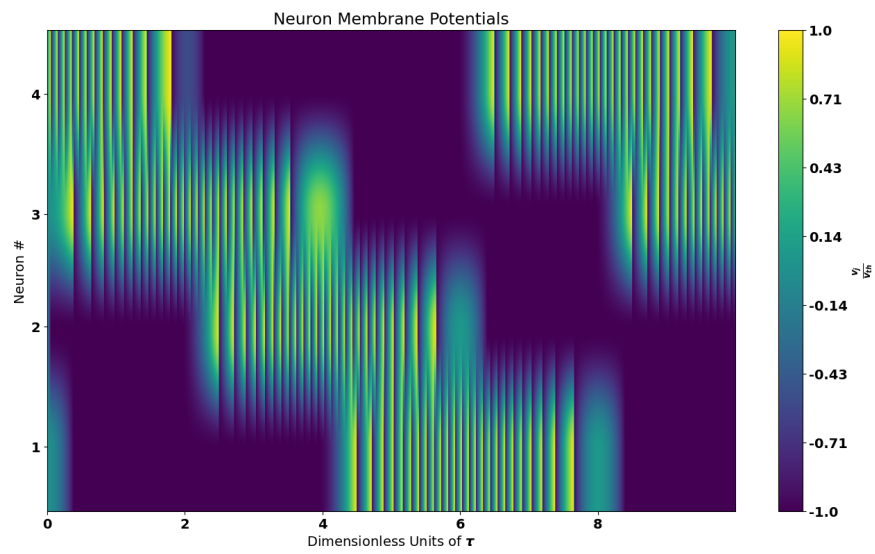
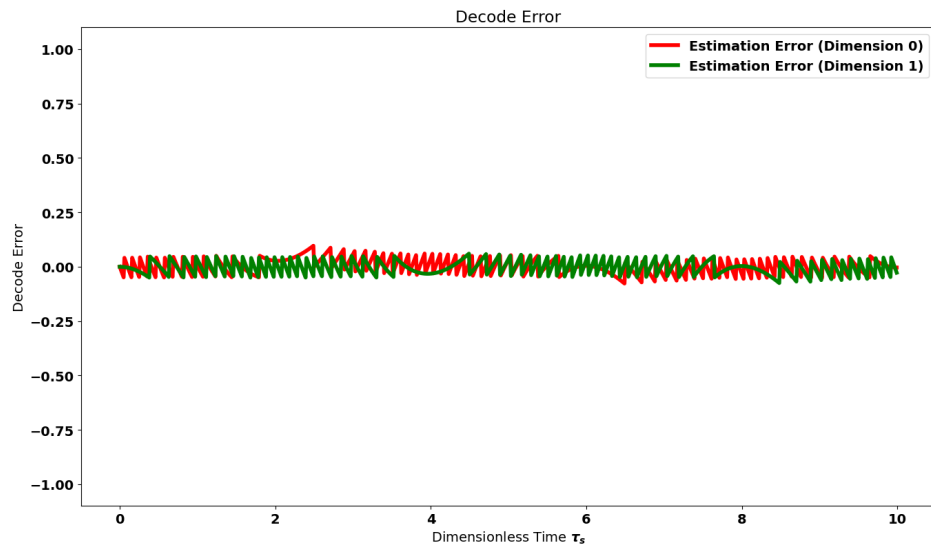
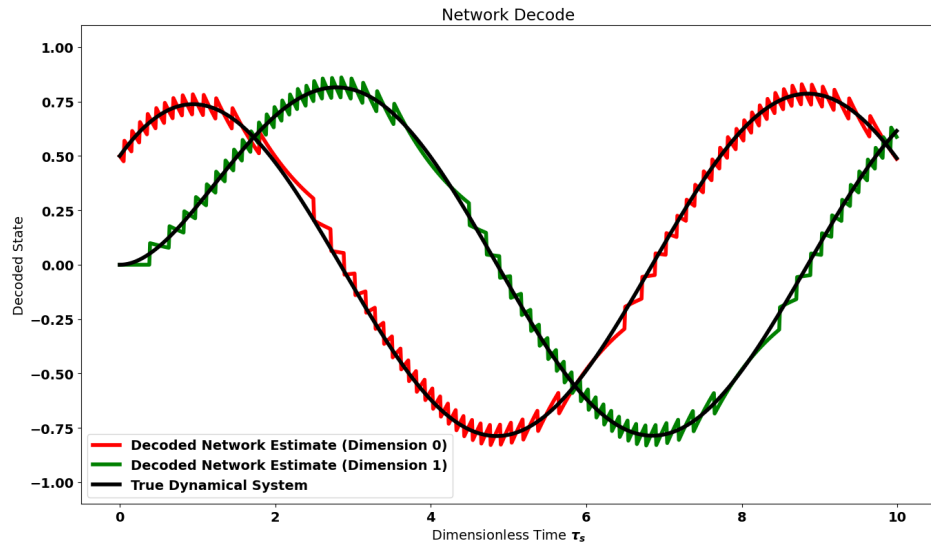




Figure 6: Simulation of equations (2.14) and (2.11) with parameters listed in equation (2.17). **Top:** The decoded network estimate plotted alongside the target dynamical system. **Middle:** The estimation error along each state-space dimension. **Bottom:** The membrane potentials of the 4 neurons during the same time period.

For the numerical implementation, the matrix exponential was used to integrate the continuous terms over a simulation time step. Continuous terms include all equation terms excepting the delta functions  $\omega$  handled separately. After integrating over a timestep, any neuron above threshold was manually reset (action of fast inhibition). If multiple neurons are above threshold, the system is integrated backwards in time until only one neuron is above threshold before spiking. The matrix exponential was computed using a Padé approximation via the Python package Scipy: `scipy.linalg.expm()`.

### 3 RMSE vs Spike Rate for Constant Driving Force

We analyse the network described by equations (2.14) and (2.11) for the case of a constant (in time) driving force  $\tilde{c}(\xi) = \tilde{c}$ .

#### 3.1 Ideal Network

We assume the network

#### 3.2 Neuron Trajectories

Consider the dynamical system

$$\dot{y} = \Lambda y + \beta \tilde{c}, \quad y(0) = y_0$$

as implemented by the network of  $N$  neurons whose encoding directions are

$$\begin{bmatrix} S & 0 \end{bmatrix} \in \mathbf{R}^{d \times N},$$

where

$$S = \begin{bmatrix} S_1 & \dots & S_d \end{bmatrix} \in \mathbf{R}^{d \times d}$$

is a diagonal matrix.

Let

$$\tilde{c} = k \hat{u} \in \mathbf{R}^d$$

such that

$$\begin{aligned} -\Lambda^{-1} \beta (k \hat{u}) &= k \frac{S_j}{\|S_j\|} \\ \implies \tilde{c} &= -\frac{k}{\|S_j\|} \beta^{-1} \Lambda S_j. \end{aligned}$$

With this choice of  $\tilde{c}$ , if we assume neuron  $j$  spikes periodically at a rate  $\phi_j$ .

#### 3.3 Solution for the Periodically Spiking Network

Assume the spike train  $\tilde{o}_j$  is a periodic sequence of impulses spaced in time by  $\frac{1}{\phi_j}$ . If the first spike occurs at  $\xi_j^0 = 0$ , then  $\tilde{o}_j(\xi) = \sum_{l=0}^{\infty} \delta\left(\xi - \frac{l}{\phi_j}\right)$ . By our choice of  $k$ , only neuron  $j$  will spike so that

$$\dot{\hat{y}}(\xi) = -\hat{y} + \sum_{l=0}^{\infty} \delta\left(\xi_j^k - \frac{l}{\phi_j}\right) S_j. \quad (3.1)$$

Equation (3.1) implies that the network estimate  $\hat{y}$  will decay until  $j$ 's first spike occurs at  $\xi_j^1 = \frac{1}{\phi_j}$ .

$$\hat{y}(\xi) = \hat{y}(0) e^{-\xi}, \quad 0 \leq \xi < \frac{1}{\phi_j}.$$

At this instant,  $S_j$  is added to neuron  $j$ 's component of the estimate.

$$\hat{y}\left(\frac{1}{\phi_j}\right) = \hat{y}(0) e^{-\frac{1}{\phi_j}} + S_j.$$

Decay again occurs until the next spike

$$\begin{aligned}
\hat{y}(\xi) &= \hat{y}\left(\frac{1}{\phi_j}\right) e^{-(\xi - \frac{1}{\phi_j})}, \\
&= \left(\hat{y}(0) e^{-\frac{1}{\phi_j}} + S_j\right) e^{-(\xi - \frac{1}{\phi_j})}, \quad \frac{1}{\phi_j} \leq \xi < \frac{2}{\phi_j} \\
\implies \hat{y}\left(\frac{2}{\phi_j}\right) &= \left(\hat{y}(0) e^{-\frac{1}{\phi_j}} + S_j\right) e^{-\frac{1}{\phi_j}} + S_j \\
&= x(0) e^{-\frac{2}{\phi_j}} + S_j e^{-\frac{1}{\phi_j}} + S_j.
\end{aligned}$$

The third spike more clearly shows the recursive behavior

$$\begin{aligned}
\hat{y}\left(\frac{3}{\phi_j}\right) &= \left[\hat{y}(0) e^{-\frac{2}{\phi_j}} + S_j e^{-\frac{1}{\phi_j}} + S_j\right] e^{-\frac{1}{\phi_j}} + S_j \\
&= \hat{y}(0) e^{-\frac{3}{\phi_j}} + S_j e^{-\frac{2}{\phi_j}} + S_j e^{-\frac{1}{\phi_j}} + S_j
\end{aligned}$$

Let us consider the  $n^{th}$  spike sufficiently far from  $\xi = 0$  such that the transient term  $\hat{y}(0) e^{-\frac{n}{\phi_j}}$  can be neglected. This leads to the expression

$$\begin{aligned}
\hat{y}\left(\frac{n}{\phi_j}\right) &= \sum_{l=0}^{n-1} S_j e^{-\frac{l}{\phi_j}} \\
&= S_j \frac{1 - e^{-\frac{n}{\phi_j}}}{1 - e^{-\frac{1}{\phi_j}}}.
\end{aligned}$$

For sufficiently large  $n$ , this converges to

$$\hat{y}(\xi_1^n) = \frac{S_j}{1 - e^{-\frac{1}{\phi_j}}}.$$

We know from equation (3.1) that the estimate will decay exponentially from this value over an interval  $\frac{1}{\phi_j}$  until a spike returns it to the initial value. Thus we have an explicit expression for the long-term behavior given by

$$\hat{y}(\xi) = \frac{S_j}{1 - e^{-\frac{1}{\phi_j}}} e^{-(\xi) \bmod \frac{1}{\phi_j}}, \quad (3.2)$$

where  $x \bmod y$  denotes the fractional remainder of  $x$  after division by  $y$ .

### 3.4 RMSE for Periodic Spiking

From equation (3.2) the error  $\epsilon = y - \hat{y}$  is a periodic function of  $\xi$  with period  $\frac{1}{\phi_j}$ .

Assume

$$\dot{y} = 0$$

$$\implies y(\xi) = k \frac{S_j}{\|S_j\|}.$$

We compute the RMSE of the error signal  $\epsilon$  by

$$RMSE = \sqrt{\phi_j \int_0^{\frac{1}{\phi_j}} \|\epsilon(\tau)\|^2 d\tau}. \quad (3.3)$$

The integrand simplifies to

$$\begin{aligned} \|\epsilon\|^2 &= (y_j - \hat{y})^2 \\ &= \|y\|^2 - 2y^T \hat{y} + \|\hat{y}\|^2 \\ &= k^2 - 2 \frac{k \|S_j\|}{1 - e^{-\frac{1}{\phi_j}}} e^{-\tau} + \frac{\|S_j\|^2}{\left(1 - e^{-\frac{1}{\phi_j}}\right)^2} e^{-2\tau} \end{aligned}$$

Note that

$$\int_0^{\frac{1}{\phi_j}} e^{-\tau} d\tau = 1 - e^{-\frac{1}{\phi_j}},$$

while

$$\begin{aligned} \int_0^{\frac{1}{\phi_j}} (e^{-\tau})^2 d\tau &= \frac{1 - e^{-\frac{2}{\phi_j}}}{2} \\ &= \frac{1}{2} \left(1 - e^{-\frac{1}{\phi_j}}\right) \left(1 + e^{-\frac{1}{\phi_j}}\right). \end{aligned}$$

Therefore the integral is

$$\phi \int_0^{\frac{1}{\phi}} \epsilon_j(\tau)^2 d\tau = k^2 - 2\phi k \|S_j\| + \phi_j \frac{\|S_j\|^2}{2} \frac{1 + e^{-\frac{1}{\phi_j}}}{1 - e^{-\frac{1}{\phi_j}}}.$$

The RMSE of the network estimate is thus

$$RMSE = \sqrt{k^2 - 2\phi k \|S_j\| + \phi_j \frac{\|S_j\|^2}{2} \frac{1 + e^{-\frac{1}{\phi_j}}}{1 - e^{-\frac{1}{\phi_j}}}}. \quad (3.4)$$

Note  $\phi_j$  is dependent on the remaining parameters. We wish to reduce equation (3.4) to a function of independent variables. The rate  $\phi_j$  had no analytic solution from the voltage trajectory, but can be deduced from equation (3.2). Consider the network estimate immediately before a spike occurs, i.e

$$\hat{y}(\xi^-) = \frac{S_j}{1 - e^{-\frac{1}{\phi}}} e^{-\frac{1}{\phi}}.$$

At this point, the error induces a spike in neuron  $j$ , i.e.

$$v(\xi) = S_j \epsilon = v_{th} = \frac{\|S_j\|^2}{2}.$$

This implies

$$\begin{aligned} \frac{\|S_j\|^2}{2} &= S_j^T (y - \hat{y}) \\ &= \|S_j\|^2 \left( \frac{k}{\|S_j\|} - \frac{e^{-\frac{1}{\phi_j}}}{1 - e^{-\frac{1}{\phi_j}}} \right). \\ &= \|S_j\|^2 \left( \frac{k}{\|S_j\|} - \frac{1}{e^{\frac{1}{\phi_j}} - 1} \right). \\ \implies e^{\frac{1}{\phi}} - 1 &= \frac{1}{\frac{k}{\|S_j\|} - \frac{1}{2}} \\ \implies \frac{1 + e^{-\frac{1}{\phi}}}{1 - e^{-\frac{1}{\phi}}} &= \frac{2k}{\|S_j\|}. \end{aligned}$$

Substitute this last expression into equation (3.4) to get

$$\begin{aligned} RMSE_j &= \sqrt{k^2 - 4\phi k^2 \frac{1 - e^{-\frac{1}{\phi_j}}}{1 + e^{-\frac{1}{\phi_j}}} + 2\phi_j k^2 \frac{1 - e^{-\frac{1}{\phi_j}}}{1 + e^{-\frac{1}{\phi_j}}}} \\ &= k \sqrt{1 - 2\phi_j \tanh\left(\frac{1}{2\phi_j}\right)}. \end{aligned}$$

Finally, we can normalize by  $k$  to obtain the NRMSE that is both dimensionless and solely dependent on the firing rate.

$$NRMSE_j(\phi_j) = \sqrt{1 - 2\phi_j \tanh\left(\frac{1}{2\phi_j}\right)}. \quad (3.5)$$

Equation (3.5) is plotted in figure (7).

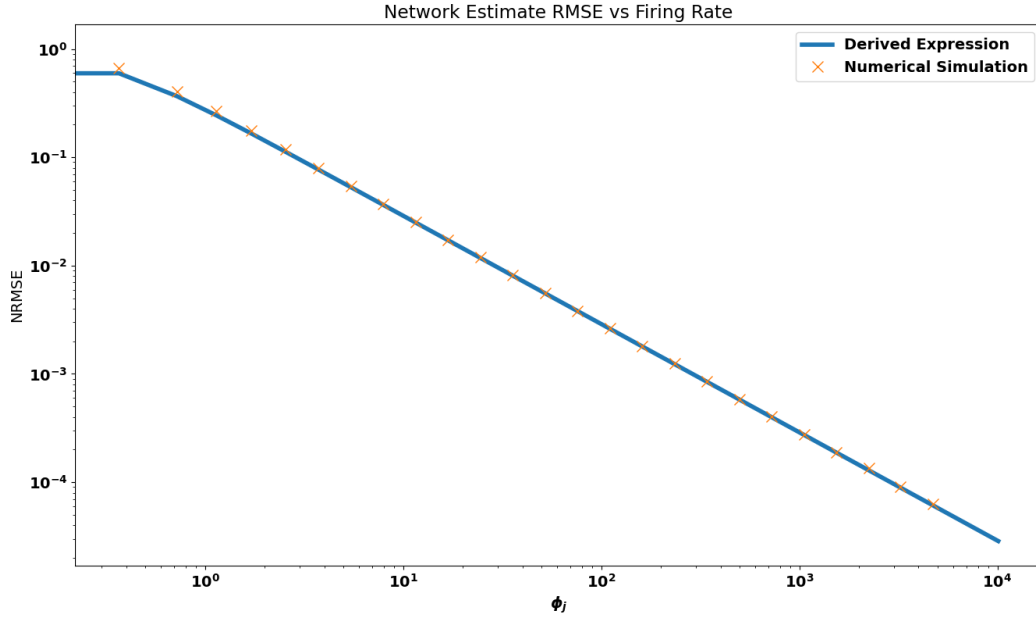


Figure 7: Log-log Plot of Equation (3.5). The RMSE was computed numerically by the discrete integral  $RMSE = \sqrt{\frac{1}{\phi} \sum_i (y - \hat{y})^2 dt}$ , where  $i$  sums over the data points in a spike. The integration is performed over  $n = 100$  interspike intervals and averaged to form a data point. The time step was  $d\xi = 10^{-4}$ .

## 4 Derivation: The Predictive Coding Framework and Gap-Junction Network

Here we derive the the predictive coding framework (PCF) as defined in Boerlin & Deneve, 2013. We highlight an assumption in this model which we later show worsens the network estimate. The correction of this assumption produces a third model that directly coupled membrane voltages. We term this a model the *gap-junction* (GJ) network.

### 4.1 Identical Initial Derivation for All Three Models (PCF, GJ, SC):

The PCF and GJ derivations are initially identical to the SC model in section (2). For all three models:

1. Let  $\tau_s$  be the synaptic time constant of each synapse in the network. Define dimensionless time as:

$$\xi \triangleq \frac{t}{\tau_s}.$$

We now assume our Linear Dynamical System is expressed in dimensionless time, i.e

$$\frac{dx}{d\xi} = Ax(\xi) + Bc(\xi). \quad (4.1)$$

The neuron encoding directions are given by

$$D = [d_1 \quad \dots \quad d_N].$$

To describe the neuron dynamics in dimensionless time, let  $o(\xi) \in \mathbf{R}^N$  be the spike trains of N neurons composing the network with components

$$o_j(\xi) = \sum_{k=1}^{n_j \text{ spikes}} \delta(\xi - \xi_j^k),$$

where  $\xi_j^k$  is the time at which neuron  $j$  makes its  $k^{th}$  spike. Define the network's estimate of the state variable as

$$\hat{x}(\xi) \triangleq Dr(\xi), \quad (4.2)$$

where  $D \in \mathbf{R}^{d \times N}$  and

$$\frac{dr}{d\xi} = -r + o(\xi). \quad (4.3)$$

The network estimation error is

$$e(\xi) \triangleq x(\xi) - \hat{x}(\xi). \quad (4.4)$$

2. Each network greedily minimizes the objective

$$\mathcal{L}(\xi) = \|x(\xi + d\xi) - \hat{x}(\xi + d\xi)\|^2.$$

When neuron  $j$  spikes, the estimate becomes

$$\mathcal{L}_{spike} = \|x - \hat{x} - d_j\|^2.$$

If  $j$  does not spike the error is

$$\mathcal{L}_{ns} = \|x - \hat{x}\|^2.$$

Neuron  $j$  spikes when it decreases the objective, i.e.

$$\mathcal{L}_{sp} < \mathcal{L}_{ns},$$

which gives spiking condition

$$d_j^T e = \frac{\|d_j\|^2}{2}.$$

This leads us to define voltage as

$$v \triangleq D^T e.$$

3. From equations (2.3) and (2.2), we have

$$D\dot{r} + Dr = Do$$

$$\implies \dot{\hat{x}} + \hat{x} = Do,$$

where the dot denotes derivative w.r.t dimensionless time  $\xi$ .

## 4.2 PCF Derivation:

The voltage dynamics  $v_{pcf}$  are given by

$$\begin{aligned} \dot{v}_{pcf} &= D^T \dot{e} \\ &= D^T (\dot{x} - \dot{\hat{x}}) \\ &= D^T (Ax + Bc - Do - \hat{x}). \end{aligned}$$

The PCF argues that when the network functions,  $x = \hat{x}$ . We then have

$$\begin{aligned} \dot{v}_{pcf} &= D^T (A\hat{x} + Bc - Do - \hat{x}) \\ &= D^T (A - I) \hat{x} + D^T Bc - D^T Do. \end{aligned}$$

With  $\hat{x} = Dr$  PCF has voltage dynamics

$$\dot{v}_{pcf} = D^T (A - I) Dr + D^T Bc - D^T Do. \quad (4.5)$$

Derive the spiking condition with an identical method to the SC network in section (2) to get PCF threshold voltages

$$v_{th} = \frac{1}{2} \begin{bmatrix} d_1^T d_1 \\ \vdots \\ d_N^T d_N \end{bmatrix}.$$



### 4.3 GJ Derivation:

The voltage dynamics of the GJ model are

$$\dot{v}_{GJ} = D^T (Ax + Bc - Do - \hat{x}).$$

The GJ model does not assume  $x = \hat{x}$ . Rather, it uses equations (4.4) and (??):

$$v_{gj} = D^T e$$

$$= D^T (x - \hat{x})$$

$$\implies x = D^{T\dagger} v_{GJ} + \hat{x}.$$

Substitute this in the dynamics equation and simplify to get

$$\dot{v}_{GJ} = D^T A D^{T\dagger} v_{GJ} + D^T (A - I) Dr + D^T Bc - D^T Do. \quad (4.6)$$

The addition of the voltage coupling term  $D^T A D^{T\dagger} v_{GJ}$  leads to the name Gap-Junction.

## 5 PCF and Gap-Junction Response to Constant Stimulus

We compute the response of the PCF and GJ models to a constant driving stimulus as we did for the SC model in section (3)

### 5.1 PCF Network Response to Constant Stimulus:

To compare between PCF and SC, we consider the same dynamical system as before, but using unrotated bases i.e.

$$\dot{x} = Ax + Bc, \quad x(0) = x_0$$

where,

$$A = \mathcal{U}\Lambda\mathcal{U}^T,$$

and the neuron encoding directions are

$$D = \mathcal{U} \begin{bmatrix} S & 0 \end{bmatrix} V^T = \begin{bmatrix} d_1 & \dots & d_N \end{bmatrix} \in \mathbf{R}^{d \times N}.$$

We choose  $c$  such that

$$x = k \frac{d_j}{\|d_j\|}$$

is a fixed point, i.e.

$$\dot{x} = 0$$

$$\implies c = -\frac{k}{\|d_j\|} B^{-1} A d_j, \quad k \in \mathbf{R}.$$

The network error is  $e = x - \hat{x}$  consequently grows parallel to  $d_j$ , ensuring only neuron  $j$  spikes periodically. The spike train  $o_j$  becomes a periodic sequence of impulses spaced in time by  $\frac{1}{\phi_j}$ . If the first spike occurs at  $\xi_j^0 = 0$ , then  $\phi_j(\xi) = \sum_{l=0}^{\infty} \delta\left(\xi - \frac{l}{\phi_j}\right)$ . Since only neuron  $j$  spikes, the estimate is

$$\dot{\hat{x}}(\xi) = -\hat{x} + \sum_{l=0}^{\infty} \delta\left(\xi - \frac{l}{\phi_j}\right) d_j.$$

Using an identical inductive approach as before, the PCF network steady state estimate is

$$\hat{x}(\xi) = \frac{d_j}{1 - e^{-\frac{1}{\phi_j}}} e^{-(\xi) \bmod \frac{1}{\phi_j}}, \quad (5.1)$$

where  $x \bmod y$  denotes the fractional remainder of  $x$  after division by  $y$ .

We compute the RMSE of the error  $e$  by

$$RMSE = \sqrt{\phi_j \int_0^{\frac{1}{\phi_j}} \|e(\tau)\|^2 d\tau}.$$

The integrand simplifies to

$$\begin{aligned}
||e||^2 &= (x - \hat{x})^2 \\
&= ||x||^2 - 2x^T \hat{x} + ||\hat{x}||^2 \\
&= k^2 - 2 \frac{k}{1 - e^{-\frac{1}{\phi_j}}} ||d_j|| e^{-\tau} + \left( \frac{||d_j||}{1 - e^{-\frac{1}{\phi_j}}} \right)^2 e^{-2\tau}
\end{aligned}$$

This gives an RMSE of

$$RMSE(k, d_j, \phi_j) = \sqrt{k^2 - 2\phi_j k ||d_j|| + \phi_j ||d_j||^2 \frac{1 + e^{-\frac{1}{\phi_j}}}{1 - e^{-\frac{1}{\phi_j}}}.$$

To simplify, note that immediately before a spike,

$$\begin{aligned}
d_j^T e &= d_j^T \left( k \frac{d_j}{||d_j||} - d_j \frac{e^{-\frac{1}{\phi_j}}}{1 - e^{-\frac{1}{\phi_j}}} \right) \\
&= \frac{||d_j||^2}{2} \\
\Rightarrow \frac{||d_j||^2}{2} &= d_j^T \left( k \frac{d_j}{||d_j||} - d_j \frac{e^{-\frac{1}{\phi_j}}}{1 - e^{-\frac{1}{\phi_j}}} \right) \\
\Rightarrow e^{\frac{1}{\phi_j}} - 1 &= \frac{1}{\frac{k}{||d_j||} - \frac{1}{2}} \\
\Rightarrow \frac{1 + e^{-\frac{1}{\phi_j}}}{1 - e^{-\frac{1}{\phi_j}}} &= \frac{2k}{||d_j||}.
\end{aligned}$$

Combine this with the RMSE expression to get

$$RMSE(\phi_j, k) = k \sqrt{1 - 2\phi_j \tanh\left(\frac{1}{2\phi_j}\right)}.$$

Divide by driving strength  $k$  to get the dimensionless quantity that depends only on spike rate,

$$NRMSE(\phi_j) = \sqrt{1 - 2\phi_j \tanh\left(\frac{1}{2\phi_j}\right)}. \quad (5.2)$$

Equation (5.3) is identical to the self-coupled network RMSE, equation (3.5) and is plotted against numerical simulations of a PCF network in figure (8)

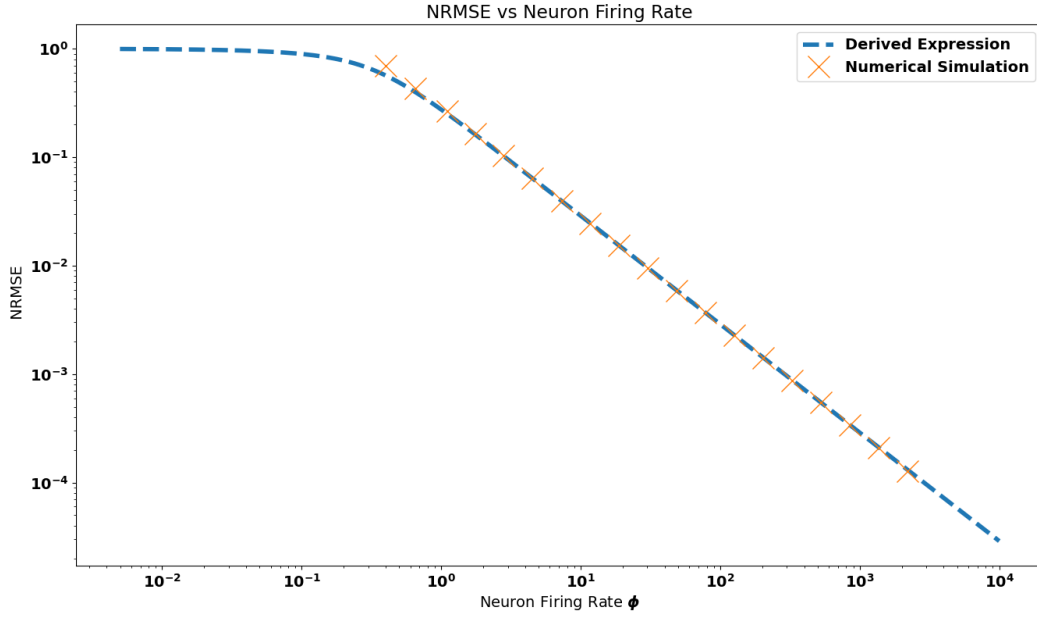


Figure 8: Log-log Plot of Equation (5.3). The RMSE was computed numerically by the discrete integral  $RMSE = \sqrt{\frac{1}{\phi} \sum_i (x - \hat{x})^2 d\xi}$ , where  $i$  sums over the data points in a spike. The integration is performed over  $n = 100$  interspike intervals and averaged to form a data point. The time step was  $d\xi = 10^{-4}$ .

## 5.2 Gap-Junction Network Response to Constant Stimulus:

We now implement the same dynamical system as before with a GJ network. As with the PCF network, we drive the GJ network such that the fixed point is  $x = k \frac{d_j}{\|d_j\|}$  parallel to neuron  $j$ ,

$$\dot{x} = 0 \implies c = -\frac{k}{\|d_j\|} B^{-1} A d_j.$$

This reduces the network to only neuron  $j$  spiking periodically.

The spike train  $o_j$  becomes a periodic sequence of impulses spaced in time by  $\frac{1}{\phi_j}$ . If the first spike occurs at  $\xi_j^0 = 0$ , then  $\phi_j(\xi) = \sum_{l=0}^{\infty} \delta\left(\xi - \frac{l}{\phi_j}\right)$ . Since only neuron  $j$  spikes, the estimate is

$$\dot{\hat{x}}(\xi) = -\hat{x} + \sum_{l=0}^{\infty} \delta\left(\xi_j^k - \frac{l}{\phi_j}\right) d_j.$$

Note that this estimate is identical to the PCF estimate above. Following the same procedure, we arrive at the GJ NRMSE:

$$NRMSE(\phi_j) = \sqrt{1 - 2\phi_j \tanh\left(\frac{1}{2\phi_j}\right)}. \quad (5.3)$$

Equation (5.3) is identical to the self-coupled network RMSE, equation (3.5) and is plotted against numerical simulations of a PCF network in figure (9)

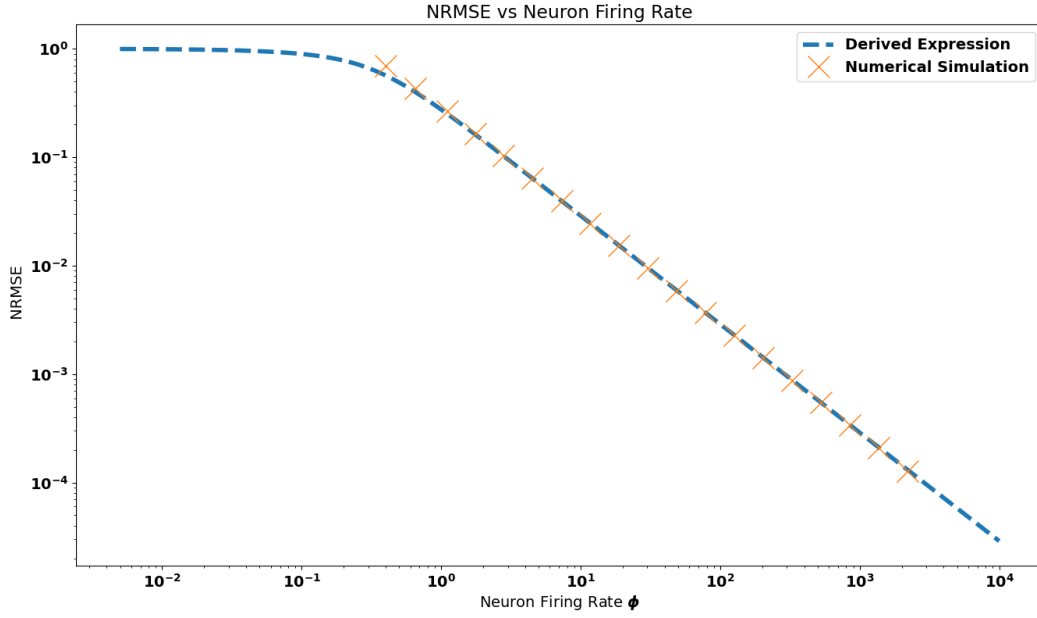


Figure 9: Log-log Plot of Equation (5.3). The RMSE was computed numerically by the discrete integral  $RMSE = \sqrt{\frac{1}{\phi} \sum_i (x - \hat{x})^2 d\xi}$ , where  $i$  sums over the data points in a spike. The integration is performed over  $n = 100$  interspike intervals and averaged to form a data point. The time step was  $d\xi = 10^{-4}$ .

### 5.3 Comparison of Self-Coupled, Gap-Junction, and PCF Networks for a Constant Stimulus

We now compare all three models as they respond to a constant driving stimulus. Each is driven parallel to a single neuron  $j$ . As we showed, each results in an NRMSE given by

$$NRMSE(\phi_j) = \sqrt{1 - 2\phi_j \tanh\left(\frac{1}{2\phi_j}\right)}.$$

To validate numerically, we simulate the SC, PCF, and GJ networks using the same network parameters. Next we compare each NRMSE as predicted by the equation above with numerical measurements. This is shown in figure (10). Note that while the relationship between firing rate and NRMSE is the same, the firing rate under a given set of parameters is not the same between networks.

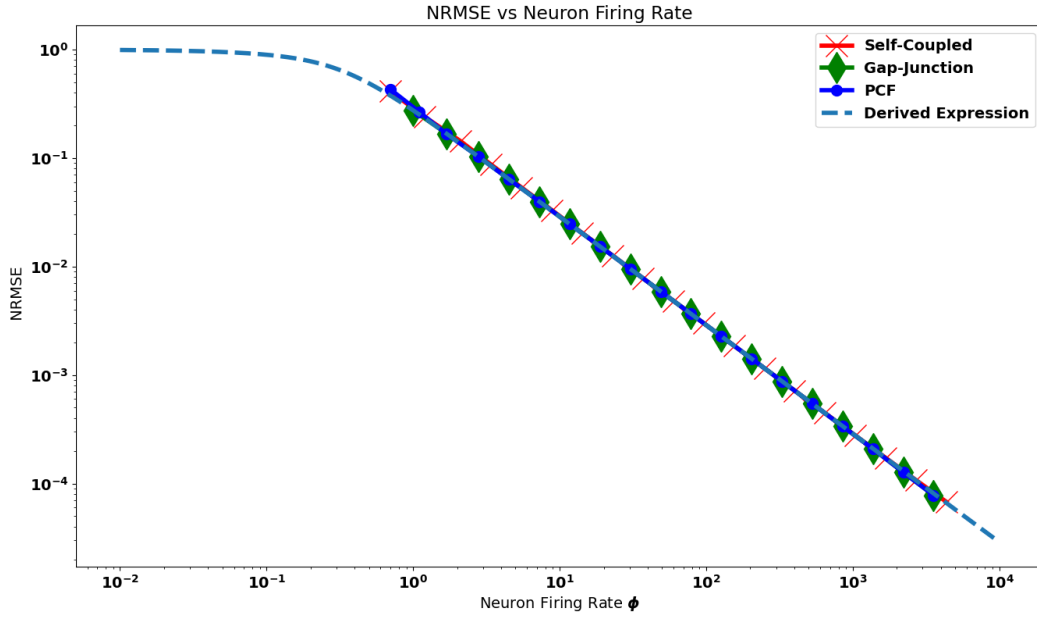


Figure 10: Simulated RMSE for self-coupled, gap-junction, and PCF networks. The dotted line is the derived expression for each model given by given above. Spike rates were estimated numerically by dividing the number of spikes by the simulation length. The RMSE was computed numerically by the discrete integral  $RMSE = \sqrt{\hat{\phi} \sum_{\tau \text{ between spikes}} e(\xi)^T e(\xi) d\xi}$ . All computations used the numerically estimated spike rate.