Self-Coupled Spiking Nets

Chris Fritz

July 2020

Contents

1 The self-coupled SNN Model

1.1 Problem Statement

Given:

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

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

1.2 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 maintains numerical accuracy over time.

2. ***Tuning Curve Rotation*** To most efficiently use N neurons, we use orthogonal bases to choose tuning cures. The decoder and dynamics matrices D, A are rotated to a common orthonormal basis in d-dimensional space via singular value decomposition (SVD).

   The rotation eliminates off-diagonal elements of the network connectivity matrices. This decoupling prevents numerical instability when spikes simultaneously occur. Multiple neurons may now spike in the same simulation time step whereas Deneve networks forbid simultaneous spiking.

   Two neurons per dimension are required since voltage thresholds are strictly positive. 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 stochastic thinning of the post-synaptic potential (PSP) at every synaptic connection. The pre-synaptic neuron's membrane potential is still reset by an action potential.

4. ***Dimensionless Time*** We describe both the network and target system in dimensionless time. Time is normalized by the neuron's synaptic time constant, τ_s. This dimensionless representation ensures consistent numerical simulation independent of simulation timestep. Furthermore, the PSP decay rate is implicitly specified as 1, reducing the required simulation parameters.

1.3 Derivation: Basic Model

1. Let τ_s be the synaptic time constant of each neuron 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

$$\frac{dx}{d\xi} = Ax(\xi) + Bc(\xi). \tag{1}$$

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

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

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

$$\hat{x}(\xi) \triangleq Dr(\xi), \tag{2}$$

where

$$\frac{dr}{d\xi} = -r + o(\xi). \tag{3}$$

,
where r_j is the total received post-synaptic current (PSC) of neuron j. Define the network error as

$$e(\xi) \triangleq x(\xi) - \hat{x}(\xi) \tag{4}$$

2. From equations (3) and (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 ξ.

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

$$\begin{aligned}
\dot{e} &= \dot{x} - \dot{\hat{x}} \\
&= (Ax + Bc) - (Do - \hat{x}) \\
&= A\left(e + \hat{x}\right) + Bc - Do + \hat{x} \\
&= Ae + (A + I)\hat{x} + Bc - Do \\
&= Ae + (A + I)\left(Dr\right) + Bc - Do \\
\implies A^{-1}\dot{e} &= e + (I + A^{-1})Dr + A^{-1}Bc - A^{-1}Do \\
\implies D^T A^{-1}\dot{e} &= D^T e + D^T(I + A^{-1})Dr + D^T A^{-1}Bc - D^T A^{-1}Do
\end{aligned} \tag{5}$$

where the third equality follows from equation (4).

3. Assuming both D and A are full rank, diagonalize A and D to a common left basis:

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

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

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

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

To express equation (5) with the $\mathcal{U}$ and V bases, first note

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

and

$$\begin{aligned} D^T A^{-1} D &= V \begin{bmatrix} S \\ 0 \end{bmatrix} \mathcal{U}^T \mathcal{U} \Lambda^{-1} \mathcal{U}^T \mathcal{U}\,[S\ 0]\,V^T \\ &= V \begin{bmatrix} S \\ 0 \end{bmatrix} \Lambda^{-1}\,[S\ 0]\,V^T \\ &= \sum_{j=1}^{d} \frac{S_j^2}{\Lambda_j} V_j V_j^T. \end{aligned}$$

Consequently,

$$\sum_{j=1}^{d} \frac{S_j}{\Lambda_j} V_j \mathcal{U}_j^T \dot{e} = \sum_{j=1}^{d} S_j V_j \mathcal{U}_j^T e + \sum_{j=1}^{d} S_j^2 (1 + \Lambda_j^{-1}) V_j V_j^T r + \sum_{j=1}^{d} \frac{S_j}{\Lambda_j} V_j \mathcal{U}_j^T Bc - \sum_{j=1}^{d} \frac{S_j^2}{\Lambda_j} V_j V_j^T o. \tag{6}$$

Multiply both sides of the preceding equation by V_j^T to arrive at the system of equations

$$\frac{S_j}{\Lambda_j} \mathcal{U}_j^T \dot{e} = S_j \mathcal{U}_j^T e + S_j^2 (1 + \Lambda_j^{-1}) V_j^T r + S_j \Lambda_j^{-1} \mathcal{U}_j^T Bc - S_j^2 \Lambda_j^{-1} V_j^T o$$

$$\implies S_j \mathcal{U}_j^T \dot{e} = S_j \Lambda_j \mathcal{U}_j^T e + S_j^2 (\Lambda_j + 1) V_j^T r + S_j \mathcal{U}_j^T Bc - S_j^2 V_j^T o,$$

for $j = 1, \ldots, d$.

4. To simplify notation, make the following substitutions which define our rotated neuron's spike train, membrane voltage, PSC, and input matrix respectively:

$$\begin{aligned} \Omega_j &\triangleq S_j^2 V_j^T o \\ v_j &\triangleq S_j \mathcal{U}_j^T e, \\ \rho_j &\triangleq S_j^2 V_j^T r \\ \beta_j &\triangleq S_j \mathcal{U}_j^T B. \end{aligned} \tag{7}$$

The system of equations simplifies to the membrane voltage dynamics

$$\dot{v}_j = \Lambda_j v_j + (\Lambda_j + 1)\rho_j + \beta_j c - \Omega_j,$$

or in matrix form,

$$\dot{v} = \Lambda v + (\Lambda + I)\rho + \beta c - \Omega. \tag{8}$$

Here, v is a d vector which describes the dynamics of the d-neurons needed to implement the dynamical system. The remaining $N - d$ neurons are unused and do not contribute to the network readout at present.
From equation (3) the PSC dynamics are

$$\dot{\rho} = -\rho + \Omega. \tag{9}$$

Similar to equation (8), ρ describes a d-vector.

5. The spike trains $\Omega(\xi)$ are chosen minimize the network estimation error

$$\mathcal{L}(\xi) = ||x(\xi + d\xi) - \hat{x}(\xi + d\xi)||. \tag{10}$$

The network greedily minimizes $\mathcal{L}$ an instant $d\xi$ ahead in time. Writing $\hat{x}$ in terms of Ω and ρ, equations (2) and (7) imply

$$
\begin{aligned}
\hat{x} &= Dr \\
&= \sum_{j=1}^{d} S_j \mathcal{U}_j V_j^T r \\
&= \sum_{j=1}^{d} (S_j^{-1} S_j^2) \mathcal{U}_j V_j^T r \\
&= \sum_{j=1}^{d} (S_j^{-1} \mathcal{U}_j)(S_j^2 V_j^T r) \\
&= \sum_{j=1}^{d} (S_j^{-1} \mathcal{U}_j) \rho_j \\
&= \mathcal{U} S^{-1} \rho \\
&= \Delta \rho,
\end{aligned}
$$

Where

$$
\Delta \stackrel{\Delta}{=} \mathcal{U} S^{-1}. \tag{11}
$$

If neuron j does not spike, the objective is

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

If neuron j spikes at time ξ, then $\Omega \leftarrow \Omega + \hat{e}_j$. The estimate $\hat{x}$ is updated so that the objective is now

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

where Δ_j is the the j^{th} column of Δ. A spike occurs when it lowers the objective more than not spiking. Our spiking condition is therefore

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

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

$$
\implies \Delta_j^T (\hat{x} - x) > \frac{\Delta_j^T \Delta_j}{2}
$$

$$
\implies \Delta_j^T e > \frac{\Delta_j^T \Delta_j}{2}.
$$

Note $\Delta_j = \mathcal{U}_j S_j^{-1}$ so that

$$\begin{aligned}
\Delta_j^T \Delta_j &= S_j^{-1} \mathcal{U}_j^T \mathcal{U}_j S_j^{-1} \\
&= S_j^{-2} \\
\implies \mathcal{U}_j^T S_j^{-1} e &> \frac{S_j^{-2}}{2} \\
\implies \mathcal{U}_j^T S_j e &> \frac{1}{2} \\
\implies v_j &> \frac{1}{2},
\end{aligned}$$

where the last inequality follows from applying the voltage definition from equation (7). Thus neuron j spikes when its membrane voltage v_j exceeds the threshold of $\frac{1}{2}$. Consequently the spiking behavior of each neuron in the network is given by

$$\begin{aligned}
&v^{th} = \frac{1}{2} 1_N, \\
&\text{if } v_j > v_j^{th}, \\
&\text{then } v_j \leftarrow v_j - 1, \\
&\text{and } \rho_j \leftarrow \rho_j + 1
\end{aligned} \tag{12}$$

where 1_N is the N-vector with entries 1.

6. Equations (8), (9), and (12) describe how we implement a network with d neurons that produces an accurate estimate $\hat{x}$ of the given target system.

   When neuron j spikes, a vector $\Delta_j = S_j^{-1} \mathcal{U}_j$ is added to the network estimate, $\hat{x}$. A spike has a strictly positive area so that the network is only able to modify its estimate by adding from a fixed set of vectors. This restricts the space representable by the network to strictly positive state-space, or only $\frac{1}{2^d}$ of the desired state-space. To remove this restriction, we add an additional d neurons whose tuning curves $\mathcal{U}_j$ are anti-parallel to neurons j for $j = 1, \ldots, d$. Such vectors are required in order to allow subtraction, defined as addition of the additive inverse. Thus the number of neurons required to represent a d-dimensional system is $2d$. We update U, S, Λ and v_{th} to reflect the additional neurons:

$$\begin{aligned}
U &\leftarrow [U \quad -U] \in \mathbf{R}^{d \text{ x } 2d}, \\
S &\leftarrow \begin{bmatrix} S & 0 \\ 0 & S \end{bmatrix} \in \mathbf{R}^{2d \text{ x } 2d}, \\
\Lambda &\leftarrow \begin{bmatrix} \Lambda & 0 \\ 0 & \Lambda \end{bmatrix} \in \mathbf{R}^{2d \text{ x } 2d}, \\
v_{th} &\leftarrow \begin{bmatrix} v_{th} \\ v_{th} \end{bmatrix} \in \mathbf{R}^{2d},
\end{aligned}$$

   and afterward recompute $\beta \in \mathbf{R}^{2d \text{ x } d}$ and $\Delta \in \mathbf{R}^{d \text{ x } 2d}$.

1.4 Simulation of Basic Equations

Here we simulate the above equations (8), (9), and (12) with the $N = 2d$ neurons. The parameters are

$$A = -\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

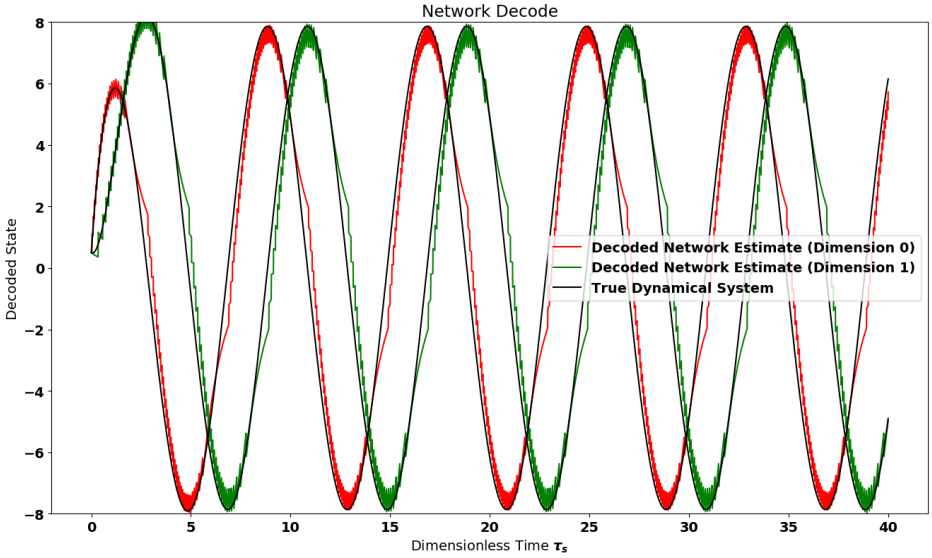
$$c(\xi) = 10\begin{bmatrix} cos(\frac{\pi}{4}\xi) \\ sin(\frac{\pi}{4}\xi) \end{bmatrix} \tag{13}$$

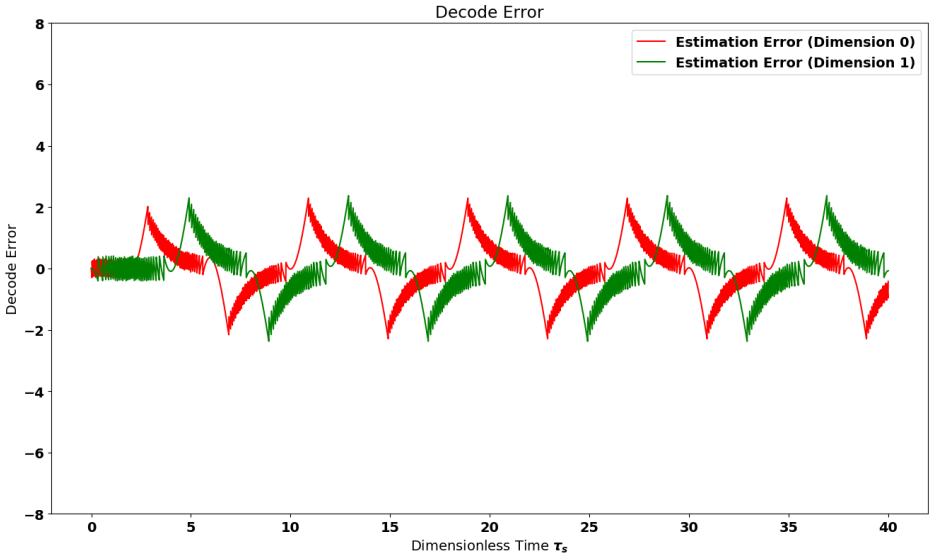
$$D_{\text{ij}} \sim \mathcal{N}(0,1) \text{ Columns Normalized to Unit Length}$$

$$d\xi = 10\mu s,$$

$$N = 4,$$

$$x(0) = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix}.$$


Network Decode
Decoded State
Dimensionless Time τ_s
Decoded Network Estimate (Dimension 0)
Decoded Network Estimate (Dimension 1)
True Dynamical System


Decode Error
Decode Error
Dimensionless Time τ_s
Estimation Error (Dimension 0)
Estimation Error (Dimension 1)

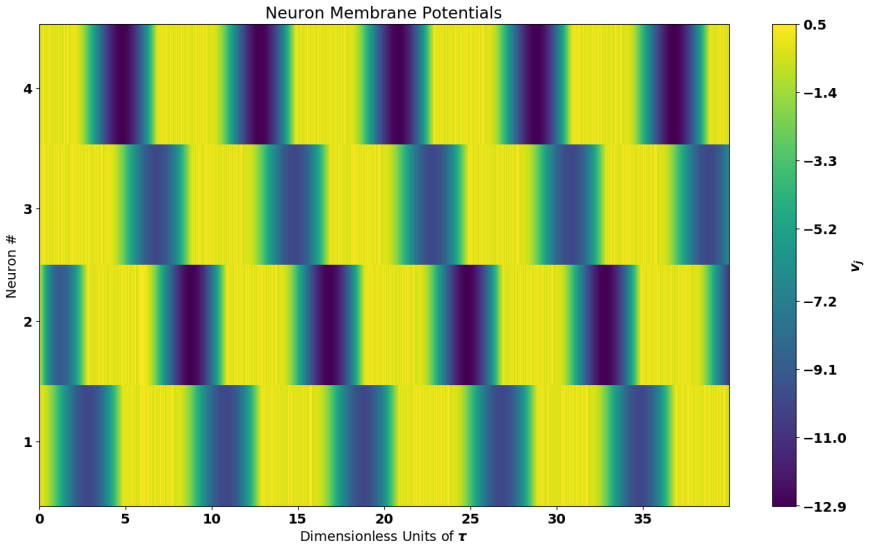

Neuron Membrane Potentials
Neuron #
Dimensionless Units of τ
v_j

Figure 1: Simulation of equations (8), (9), and (12), with parameters listed in equation (13). ***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 spike trains Ω handled separately. After integrating over a timestep, all neurons above threshold were manually reset according to the spiking rule (12). The matrix exponential was computed using a Padé approximation via the Python package Scipy: *scipy.linalg.expm()*.

1.5 Analysis: RMSE vs Spike Rate for Constant Driving Force

We analyse the network described by equations (8), (9), and (12) for the case of a constant driving force $c(\xi) = k\mathcal{U}_j$.

Let

$$A = -\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$c(\xi) = k\mathcal{U}_1$$

$$d\xi = 10\mu s,$$

$$N = 4,$$

$$x(0) = \begin{bmatrix} \frac{1}{2} & 0 \end{bmatrix}.$$

With the given initial conditions, $v_j = 0$ for $j \neq 1$ for all ξ. The dynamics simplify to

$$\dot{v_1} = \Lambda_1 v_1 + (\Lambda_1 + 1)\rho_1 + S_1 - \Omega_1.$$

We assume that the decoding matrix D is chosen such that $S_1 = 1$. Because A is the negative identity matrix, it is also clear that $\Lambda_1 = -1$. The preceding equation simplifies to

$$\dot{v_1} = -v_1 + k - \Omega_1, \tag{14}$$

which is a form of the well-known Leaky Integrate-and-Fire (LIF) model. With initial condition $v_1(0) = 0$, and neglecting spiking (Ω_1), the neuron's trajectory is readily integrated as

$$v(\xi) = k(1 - e^{-\xi}).$$

Neglecting any spike reset, the voltage will asymptotically approach $v_1 = k$. Thus for any spiking to occur, we must have $k > v_{th}$. In this case, the time required to reach a spike threshold v_{th} is

$$v_{th} = k(1 - e^{-\xi_{spike}})$$

$$\implies \xi_{spike} = ln\left(\frac{1}{1 - \frac{v_{th}}{k}}\right)$$

$$\implies \frac{1}{\xi_{spike}} = ln\left(1 - \frac{v_{th}}{k}\right),$$

which determines the frequency at which the LIF neuron spikes. Denote this frequency as a function of driving strength k by $\phi(k)$:

$$\phi(k) = ln\left(1 - \frac{v_{th}}{k}\right). \tag{15}$$

The network will encode the constant driving force by spiking at a fixed rate determined by equation (15). Similar to membrane voltage, the resulting PSC and readout dynamics are reduced to one neuron periodically spiking:

$$\dot{\rho_1} = -\rho_1 + \Omega_1$$

$$\implies \dot{\hat{x}} = -\Delta_1\rho_1 + \Delta_1\Omega_1$$

$$= -\hat{x} + \mathcal{U}_1\Omega_1.$$

The spike train Ω_1 is a periodic sequence of impulses spaced in time by $\frac{1}{\phi k}$. Hence $\Omega_1(\xi) = \sum_{l=0}^{\infty} \delta\left(\xi - \frac{l}{\phi(k)}\right)$. The network estimate therefore has dynamics

$$\dot{\hat{x}} = -\hat{x} + \mathcal{U}_1 \sum_{l=0}^{\infty} \delta\left(\xi - \frac{l}{\phi(k)}\right)$$

To compute the RMSE, note the target dynamical system is:

$$\dot{x} = -x + k\mathcal{U}_1$$
$$x(0) = \begin{bmatrix}\frac{1}{2} & 0\end{bmatrix}.$$

Therefore the network estimation error has dynamics

$$\dot{e} = \dot{x} - \dot{\hat{x}}$$

$$= -(x - \hat{x}) + \mathcal{U}_1(k - \sum_{l=0}^{\infty} \delta\left(\xi - \frac{l}{\phi(k)}\right)$$

$$\implies \dot{e} = -e + \mathcal{U}_1(k - \sum_{l=0}^{\infty} \delta\left(\xi - \frac{l}{\phi(k)}\right) \tag{16}$$