Supplementary Information

Learning and Controlling Multi-scale Dynamics in Spiking Neural Networks using Recursive Least Square Modifications

SUPPLEMENTARY NOTE 1: DERIVATION OF DDP ALGORITHM

Without loss of generality, we consider the following time-invariant system

$$x(t+1) = Ax(t) + B_{\mathcal{K}} u_{\mathcal{K}}(t), \tag{S1}$$

where $x(t) \in \mathbb{R}^k$ and $u_{\mathcal{K}}(t) \in \mathbb{R}^m$ denote, respectively, the state and input of the system at time t. In modern control theory, $A \in \mathbb{R}^{k\times k}$ is called system matrix and is constant and $B_{\mathcal{K}} \in \mathbb{R}^{k\times m}$ is input matrix (For convenience, we will abbreviate $B_{\mathcal{K}}$ as B and $u_{\mathcal{K}}$ as u in the following narrative). Fixed time steps T, initial state $x(0) = x_0$ and final state $x(T) = x_f$, the performance function is defined as

$$\begin{cases} V[x_0] & = \sum_{k=0}^{T-1} \left[x^{\mathsf{T}}(k) Q x(k) + u^{\mathsf{T}}(k) R u(k) \right] \\ V[x_f] & = 0, \end{cases} \tag{S2}$$

where $Q \in \mathbb{R}^{k\times k}$ and $R \in \mathbb{R}^{m\times m}$, denote semidefinite and positive definite matrices, respectively.

According to Bellman optimality principle, we have the recursive relationships

$$V[x(j)] = \min_{u} \left\{ x^{\mathsf{T}}(j) Q x(j) + u^{\mathsf{T}}(j) R u(j) + V[x(j+1)] \right\}, \quad j = 0, \cdots, T-1. \tag{S3}$$

Now, we deduce the optimal control backward.

Step 1. At time step $T-1$, we have

$$\begin{cases} V[x(T-1)] = \min\limits_{u(T-1)} \{ x^{\mathsf{T}}(T-1) Q x(T-1) \\ \qquad\qquad + u^{\mathsf{T}}(T-1) R u(T-1) + V[x_f] \} \\ B^{\mathsf{T}} \left[B u(T-1) + A x(T-1) - x_f \right] = 0 \\ K_{u_{T-1}} = B^{\mathsf{T}} B, \quad K_{x_{T-1}} = B^{\mathsf{T}} A, \quad K^f_{T-1} = B^{\mathsf{T}} \\ K^{u_{T-1}}_{x_{T-1}} = -K^{\mathsf{T}}_{x_{T-1}} (K^{-1}_{u_{T-1}})^{\mathsf{T}} \end{cases} \tag{S4}$$

where $K_{u_{T-1}}$, $K_{x_{T-1}}$, K^f_{T-1}, $K^{u_{T-1}}_{x_{T-1}}$ are the coefficients of $u(T-1)$, $x(T-1)$, x_f, $\frac{\partial u^{\mathsf{T}}(T-1)}{x(T-1)}$. Thus, we have

$$u^*(T-1) = {K^{u_{T-1}}_{x_{T-1}}}^{\mathsf{T}} \cdot x(T-1) + K^{-1}_{u_{T-1}} \cdot K^f_{T-1} \cdot x_f. \tag{S5}$$

We take the partial derivative of $V[x(T-1)]$ with respect to $x(T-1)$

$$\begin{aligned}\frac{\partial V(x(T-1))}{\partial x(T-1)} &= 2Q\cdot x(T-1) + \frac{u^{\mathsf{T}}(T-1)}{x(T-1)}\cdot\frac{\partial(u^{\mathsf{T}}(T-1)Ru(T-1)))}{u(T-1)} \\ &= 2Q\cdot x(T-1) + K_{x_{T-1}}^{u_{T-1}}\cdot 2\cdot R\cdot u(T-1) \\ &= 2Q\cdot x(T-1) + 2K_{x_{T-1}}^{u_{T-1}}\cdot R\cdot[K_{x_{T-1}}^{u_{T-1}\mathsf{T}}\cdot x(T-1) + K_{u_{T-1}}^{-1}\cdot K_{T-1}^{f}\cdot x_f] \\ &= 2[Q + K_{x_{T-1}}^{u_{T-1}}\cdot R\cdot K_{x_{T-1}}^{u_{T-1}\mathsf{T}}]x(T-1) + 2[k_{x_{T-1}}^{u_{T-1}}\cdot R\cdot K_{u_{T-1}}^{-1}\cdot K_{T-1}^{f}]x_f \\ &= 2V_{x_{T-1}}^{T-1}\cdot x(T-1) + 2V_{x_{T-1}}^{f}\cdot x_f \end{aligned} \tag{S6}$$

where

$$\begin{cases} V_{x_{T-1}}^{T-1} &= Q + K_{x_{T-1}}^{u_{T-1}}\cdot R\cdot K_{x_{T-1}}^{u_{T-1}\mathsf{T}}, \\ V_{x_{T-1}}^{f} &= K_{x_{T-1}}^{u_{T-1}}\cdot R\cdot K_{u_{T-1}}^{-1}\cdot K_{T-1}^{f}. \end{cases} \tag{S7}$$

Step 2. For $j = T-2, \cdots, 0$, we can obtain

$$\begin{cases} V[x(j)] = \min\limits_{u(j)}\{x^{\mathsf{T}}(j)Qx(j) + u^{\mathsf{T}}(j)Ru(j) + V[x(j+1)]\} \\ Bu(j) + Ax(j) = x(j+1). \end{cases} \tag{S8}$$

Due to the principle of optimality, the optimal control $u^*(j)$ is the solution according to $\frac{\partial V[x(j)]}{\partial u(j)} = 0$, which can be expressed as

$$\begin{aligned}\frac{\partial V[x(j)]}{\partial u(j)} &= 2R\cdot u(j) + \frac{\partial V[x(j+1)]}{\partial u(j)} \\ &= 2R\cdot u(j) + \frac{\partial x^{\mathsf{T}}(j+1)}{u(j)}\frac{\partial V[x(j+1)]}{\partial x(j+1)} \\ &= 2R\cdot u(j) + B^{\mathsf{T}}[2V_{x_{j+1}}^{j+1}\cdot x(j+1) + 2V_{x_{j+1}}^{f}\cdot x_f] \\ &= 2R\cdot u(j) + 2B^{\mathsf{T}}\cdot V_{x_{j+1}}^{j+1}[A\cdot x(j) + B\cdot u(j)] + 2B^{\mathsf{T}}\cdot V_{x_{j+1}}^{f}\cdot x_f \\ &= 2[R + B^{\mathsf{T}}\cdot V_{x_{j+1}}^{j+1}\cdot B]u(j) + 2B^{\mathsf{T}}\cdot V_{x_{j+1}}^{j+1}\cdot A\cdot x(j) + 2B^{\mathsf{T}}V_{x_{j+1}}^{f}x_f \\ &= 0. \end{aligned} \tag{S9}$$

Thus, we have

$$\begin{cases} [R + B^{\mathsf{T}}\cdot V_{x_{j+1}}^{j+1}\cdot B]u(j) + B^{\mathsf{T}}\cdot V_{x_{j+1}}^{j+1}\cdot A\cdot x(j) = -B^{\mathsf{T}}\cdot V_{x_{j+1}}^{f}\cdot x_f \\ K_{u_j} = R + B^{\mathsf{T}}\cdot V_{x_{j+1}}^{j+1}\cdot B \\ K_{x_j} = B^{\mathsf{T}}\cdot V_{x_{j+1}}^{j+1}\cdot A \\ K_j^f = -B^{\mathsf{T}}\cdot V_{x_{j+1}}^{f} \\ K_{x_j}^{u_j} = -K_{x_j}^{\mathsf{T}}\cdot(K_{u_j}^{-1})^{\mathsf{T}} \end{cases} \tag{S10}$$

where K_{u_j}, K_{x_j}, K_j^f, $K_{x_j}^{u_j}$ are the coefficients of $u(j)$, $x(j)$, x_f, $\frac{\partial u^\mathsf{T}(j)}{x(j)}$. The optimal $u^*(j)$ is

$$u^*(j) = {K_{x_j}^{u_j}}^\mathsf{T} \cdot x(j) + K_{u_j}^{-1} \cdot K_j^f \cdot x_f \tag{S11}$$

Now, we take the partial derivative of $V[x(j)]$ with respect to $x(j)$

$$\begin{aligned}
\frac{\partial V(x(j))}{\partial x(j)} &= 2Q \cdot x(j) + 2K_{x_j}^{u_j} \cdot R \cdot u(j) + \left[\frac{x^\mathsf{T}(j+1)}{x(j)} + \frac{u^\mathsf{T}(j)}{x(j)}\frac{x^\mathsf{T}(j+1)}{u(j)}\right]\frac{\partial V[x(j+1)]}{x(j+1)} \\
&= 2Q \cdot x(j) + 2K_{x_j}^{u_j} \cdot R \cdot u(j) + [A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}] \cdot [2V_{x_{j+1}}^{j+1} \cdot x(j+1) + 2V_{x_{j+1}}^f \cdot x_f] \\
&= 2Q \cdot x(j) + 2K_{x_j}^{u_j} \cdot R \cdot u(j) + 2[A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}]V_{x_{j+1}}^{j+1} \cdot x(j+1) + 2[A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}]V_{x_{j+1}}^f \cdot x_f \\
&= 2Q \cdot x(j) + 2K_{x_j}^{u_j} \cdot R \cdot u(j) + 2[A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}]V_{x_{j+1}}^{j+1} \cdot [Ax(j) + Bu(j)] + \\
&\qquad\qquad 2[A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}]V_{x_{j+1}}^f \cdot x_f \\
&= 2\left[Q + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^{j+1} A\right] x(j) + 2\left[K_{x_j}^{u_j} \cdot R + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^{j+1} \cdot B\right] u(j) \\
&\qquad\qquad + 2[A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}]V_{x_{j+1}}^f \cdot x_f \\
&= 2\left[Q + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^{j+1} \cdot A\right] x(j) \\
&\quad + 2\left[K_{x_j}^{u_j} \cdot R + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^{j+1} \cdot B\right] [{K_{x_j}^{u_j}}^\mathsf{T} \cdot x(j) + K_{u_j}^{-1} \cdot K_j^f \cdot x_f] \\
&\quad + 2[A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}]V_{x_{j+1}}^f \cdot x_f \\
&= 2\Big[Q + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^{j+1} \cdot A + K_{x_j}^{u_j} \cdot R \cdot {K_{x_j}^{u_j}}^\mathsf{T} \\
&\qquad\qquad + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^{j+1} B \cdot {K_{x_j}^{u_j}}^\mathsf{T}\Big] x(j) \\
&\quad + 2\Big[K_{x_j}^{u_j} \cdot R \cdot K_{u_j}^{-1} \cdot K_j^f + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^{j+1} \cdot B \cdot K_{u_j}^{-1} \cdot K_j^f \\
&\qquad\qquad + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^f\Big] x_f \\
&= 2V_{x_j}^j \cdot x(j) + 2V_{x_j}^f \cdot x_f
\end{aligned} \tag{S12}$$

,where

$$\begin{cases} V_{x_j}^j &= Q + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T})V_{x_{j+1}}^{j+1} \cdot A + K_{x_j}^{u_j} \cdot R \cdot {K_{x_j}^{u_j}}^\mathsf{T} + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T}) \cdot V_{x_{j+1}}^{j+1} \cdot B \cdot {K_{x_j}^{u_j}}^\mathsf{T} \\ V_{x_j}^f &= K_{x_j}^{u_j} \cdot R \cdot K_{u_j}^{-1} \cdot K_j^f + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T})V_{x_{j+1}}^f + (A^\mathsf{T} + K_{x_j}^{u_j} \cdot B^\mathsf{T})V_{x_{j+1}}^{j+1} \cdot B \cdot K_{u_j}^{-1} \cdot K_j^f. \end{cases} \tag{S13}$$

Next, we can obtain $x_0 \to u^*(0) \to x^*(1) \to u^*(1) \to \cdots \to x^*(T-1) \to u^*(T-1) \to x_f$ sequentially, according to Equations (S1), (S5) and (S11), namely:

$$\begin{cases} u^*(j) = {K_{x_j}^{u_j}}^\mathsf{T} \cdot x(j) + K_{u_j}^{-1} \cdot K_j^f \cdot x_f \\ x^*(j+1) = A \cdot x(j) + B \cdot u(j),\ j = 0, \cdots, T-1. \end{cases} \tag{S14}$$

Toward a better understanding and using this method, we establish the Algorithm **??** to describe it specifically.

SUPPLEMENTARY NOTE 2: FAILURE CASE OF DDP ALGORITHM

When matrix $B_{\mathcal{K}}$ is column full rank rather than row full rank, the optimal control sequence calculated by DDP Algorithm fails to transfer initial state x_0 to fixed final state x_f within T time steps. The case is given as follows. Let initial state be $x_0 = [0.4360\ 0.0259\ 0.5497\ 0.4353]^\mathsf{T}$, final state be $x_f = [6\ 5\ 8\ 9]^\mathsf{T}$ and time steps be $T = 3$. Matrices Q and R are identity matrices of corresponding dimensions. The matrix A and $B_{\mathcal{K}}$ are given

$$A = \begin{bmatrix} 5 & 3 & 3 & 4 \\ 3 & 1 & 3 & 2.5 \\ 3 & 3 & 4 & 1 \\ 4 & 2.5 & 1 & 3 \end{bmatrix} \tag{S15}$$

and

$$B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{S16}$$

According to DDP Algorithm, it is easy to get $x^*(2) = [0.2992\ 4.7771\ 1.8964\ -3.1611]^\mathsf{T}$ and

$$\begin{aligned} u^*(2) &= (B_{\mathcal{K}}^\mathsf{T} B_{\mathcal{K}})^{-1} B_{\mathcal{K}}^\mathsf{T} (x_f - A \cdot x^*(2)) \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -3.7534 \\ 1.6493 \\ -21.6767 \\ 3.6015 \end{bmatrix} \\ &= \begin{bmatrix} -3.7534 \\ 1.6493 \\ 3.6015 \end{bmatrix}. \end{aligned} \tag{S17}$$

Hence, the optimal final state $x_f^* \neq x_f$, namely

$$\begin{aligned} x_f^* &= A \cdot x^*(2) + B_{\mathcal{K}} \cdot u^*(2) \\ &= \begin{bmatrix} 5 & 3 & 3 & 4 \\ 3 & 1 & 3 & 2.5 \\ 3 & 3 & 4 & 1 \\ 4 & 2.5 & 1 & 3 \end{bmatrix} \begin{bmatrix} 0.8123 \\ 7.0020 \\ 3.0961 \\ -6.1506 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -3.7534 \\ 1.6493 \\ 3.6015 \end{bmatrix} \\ &= \begin{bmatrix} 6 \\ 5 \\ 29.6767 \\ 9 \end{bmatrix} \neq x_f \,. \end{aligned} \tag{S18}$$

SUPPLEMENTARY NOTE 3: DERIVATION OF EQUATION (29) EQUIVALENT TO EQUATION (30) IN THE MAINTEXT

For $T = n_k$, $n_k \in \mathbb{N}$, through induction, it is easy to verify that

$$K_{x_t}^{u_t\mathsf{T}} = -B^{\mathsf{T}} \left[A^{n_k-1-t}\right]^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0, n_k - t] A^{n_k-t} \tag{S19}$$

and

$$K_{u_t}^{-1} \cdot K_t^f = B^{\mathsf{T}} \left[A^{n_k-1-t}\right]^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0, n_k - t],\ t = 0, 1, \cdots, n_k - 1. \tag{S20}$$

According to system (S1), for $t = 0, 1, \cdots, n_k - 1$, the Equation (**??**) can be rewritten as

$$\begin{aligned} u_{\mathcal{K}}^*(t) &= K_{x_t}^{u_t\mathsf{T}} x(t) + K_{u_t}^{-1} \cdot K_t^f \cdot x_f \\ &= -K_{u_t}^{-1} \cdot K_t^f \cdot A^{n_k-t} \left[\sum_{i=0}^{t-1} A^{t-1-i} B u_{\mathcal{K}}^*(i)\right] + K_{u_t}^{-1} \cdot K_t^f \cdot x_f. \end{aligned} \tag{S21}$$

To further simplify the Equation (S21), induction is used. When $t = 0, 1$, we have

$$\begin{aligned} u_{\mathcal{K}}^*(0) &= K_{u_0}^{-1} \cdot K_0^f \cdot x_f \\ &= B^{\mathsf{T}} \left[A^{n_k-1}\right]^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0, n_k] \cdot x_f, \end{aligned} \tag{S22}$$

and

$$
\begin{aligned}
u^*_{\mathcal{K}}(1) &= -K_{u_1}^{-1}\cdot K_1^f \cdot A^{n_k-1}B\cdot u^*_{\mathcal{K}}(0) + K_{u_1}^{-1}\cdot K_1^f \cdot x_f \\
&= B^{\mathsf{T}}\left[A^{n_k-2}\right]^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k-1]\cdot\left\{I_N - A^{n_k-1}BB^{\mathsf{T}}\left[A^{n_k-1}\right]^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]x_f\right\} \\
&= B^{\mathsf{T}}\left[A^{n_k-2}\right]^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k-1]\cdot\left(W_{\mathcal{K}}[0,n_k] - A^{n_k-1}BB^{\mathsf{T}}\left[A^{n_k-1}\right]^{\mathsf{T}}\right)\cdot W_{\mathcal{K}}^{-1}[0,n_k]\cdot x_f \\
&= B^{\mathsf{T}}\left[A^{n_k-2}\right]^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]\cdot x_f. \qquad \text{(S23)}
\end{aligned}
$$

Owing to induction, we can obtain $u^*_{\mathcal{K}}(i) = B^{\mathsf{T}}\left[A^{n_k-1-i}\right]^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]x_f, i = 2,3,\cdots,t-1$, and

$$
\begin{aligned}
u^*_{\mathcal{K}}(t) &= -B^{\mathsf{T}}\left(A^{n_k-1-t}\right)^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]\cdot\left[\sum_{i=0}^{t-1} A^{n_k-1-i}BB^{\mathsf{T}}\left(A^{n_k-1-i}\right)^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]\cdot x_f\right] \\
&\quad + B^{\mathsf{T}}\left(A^{n_k-1-t}\right)^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]\; x_f \\
&= B^{\mathsf{T}}\left(A^{n_k-1-t}\right)^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]\cdot\left[I_N - \sum_{i=0}^{t-1} A^{n_k-1-i}BB^{\mathsf{T}}\left(A^{n_k-1-i}\right)^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]\right] x_f \\
&= B^{\mathsf{T}}\left(A^{n_k-1-t}\right)^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]\cdot\left[W_{\mathcal{K}}[0,n_k] - \sum_{i=0}^{t-1} A^{n_k-1-i}BB^{\mathsf{T}}\left(A^{n_k-1-i}\right)^{\mathsf{T}}\right]\cdot W_{\mathcal{K}}^{-1}[0,n_k]\cdot x_f \\
&= B^{\mathsf{T}}\left(A^{n_k-1-t}\right)^{\mathsf{T}} W_{\mathcal{K}}^{-1}[0,n_k]\cdot x_f. \qquad \text{(S24)}
\end{aligned}
$$

The proof of Equation (29) equivalent to Equation (30) in the maintext is completed.

SUPPLEMENTARY NOTE 4: COMPARISON BETWEEN IZHIKEVICH MODEL AND LEAKY INTERGRATE AND FIRE MODEL

To evaluate the performance of different SNN methods in processing multiscale discrete signals, we compare the learning effects of the Izhikevich and LIF models on tasks of low and high dimensions. In Fig. S1(a), we present the results of the Izhikevich model learning macroscopic sine waves and emitting microscopic spikes, as discussed in the main text. On the other hand, Fig. S1(b) shows the macroscopic and microscopic effects of the LIF model in the same low-dimensional task.

While the Izhikevich model is a second-order dynamic model, the LIF model is a first-order dynamic model. Therefore, the results simulated by the Izhikevich model are more precise than those of the LIF model, both intuitively and theoretically. As shown in Fig. S1(a)–(b), the RLS is turned on at 1 second and turned off at 4.5 seconds. After RLS is turned off, the simulated results of the Izhikevich model are much better than those of the LIF model. At the microscopic level, the spikes emitted by the Izhikevich model are more regular than those of the LIF model, possibly because the former has better biological plausibility.

In Fig. S1(c), we present the effects of the two models in learning the high-dimensional Lorenz curve, with the Izhikevich model on the left and the LIF model on the right. The learning effect of the Izhikevich model is much better than that of the LIF model.

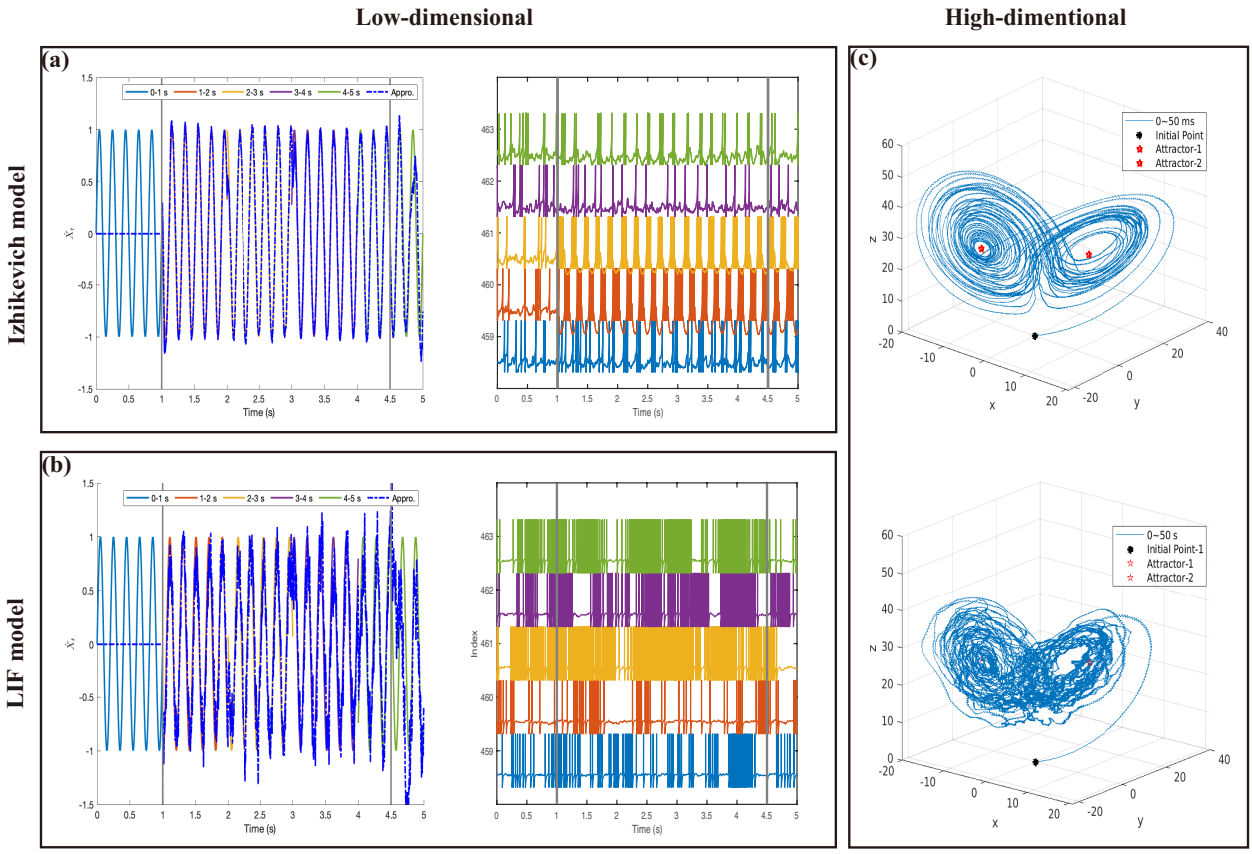

Fig. S1. A comparison between Izhikevich model and leaky Integrate and fired (LIF) model for learning and approximating macroscopic motion trajectories in low and high dimensions. (a): Results of macroscopic sinusoid in low and high dimensions. (a): Results of macroscopic sinusoidal curve and microscopic spikes in low-dimensional task using Izhikevich model. (b): Results of macroscopic sinusoidal curve and microscopic spikes in the low-dimensional task using LIF model. (c): Left: high-dimensional Lorenz curve learned and approximated by Izhikevich model. Right: high-dimensional Lorenz curve learned and approximated by LIF model.

SUPPLEMENTARY NOTE 5: THE LEARNING EFFECTS OF THE LORENZ CURVE WITH DIFFERENT VALUES OF C IN THE SNN MODEL.

We set $C = 250$ uF here with reference to the parameters in [1].

Additionally, we have verified the learning effect of the model under different parameters, as shown in the Fig. S2 below.

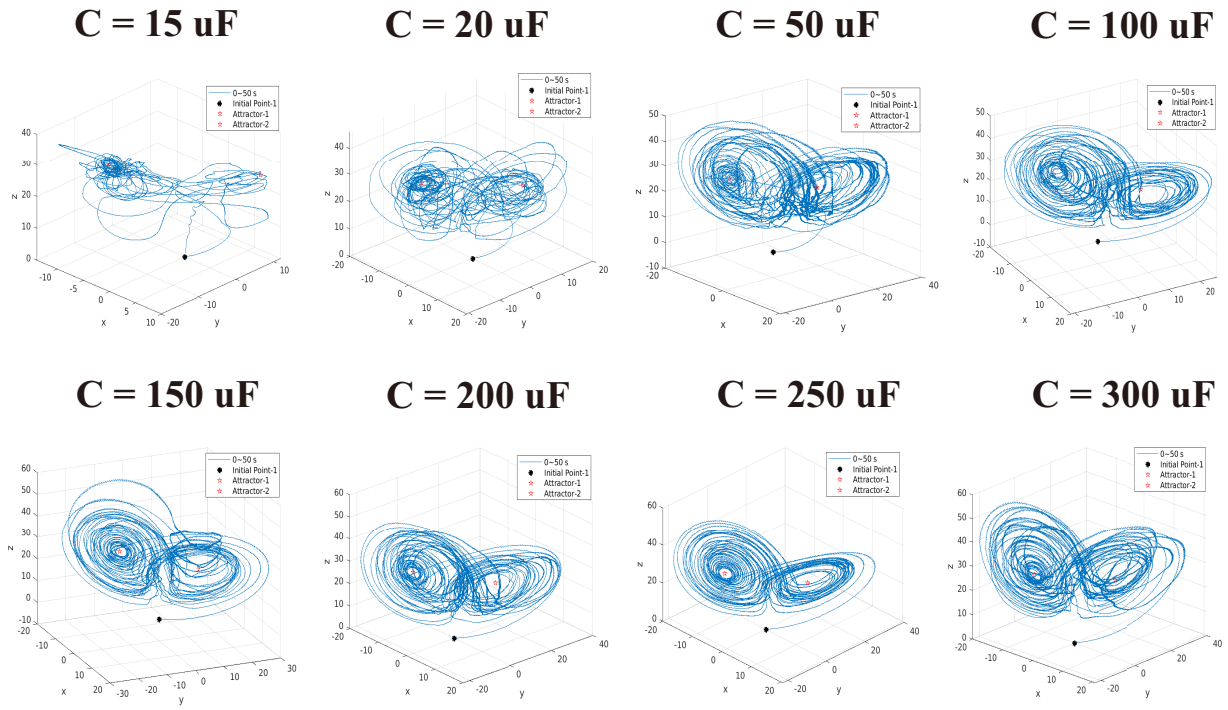

Fig. S2. The learning effects of the Lorenz curve with different values of C in the SNN model.

From Fig. S2, it can be observed that the SNN can accurately simulate the Lorenz curve when C is greater than or equal to 50uF, with the smoothest simulated curve achieved when C is set to 250uF. In contrast, when C is set to 15uF or 20uF, the SNN model fails to replicate the correct curve. It's worth noting that, in accordance with the parameter C mentioned in reference [1], we have consistently used a C value of 250uF in all of our experiments.

SUPPLEMENTARY NOTE 6: DETAILED EXPLANATION REGARDING THE REPRESENTATION OF FIG. 5(J)—(L) IN THE MAINTEXT.

The spike train of the BCI dataset underwent a time window binning with a window size of 25 ms, during which spikes within the window were accumulated. Correspondingly, in our SNN model, updates occur at 25-millisecond intervals, tracking the firing behavior of Izhikevich neurons.

Our objective is to visualize the firing spikes of different neurons over time. As a result, the Fig. 5(j)–(l) illustrate that every 25 milliseconds, Izhikevich neurons can fire a maximum of 1 spike. In contrast, the biological neurons recorded by the electrode can fire multiple spikes within the same time window, causing the values in the histogram to exceed the set boundaries.

Of course, an alternative approach would involve creating a three-dimensional graph featuring axes representing “time," “neurons," and “number of pulses," which might enhance clarity but could potentially lead to a cluttered visual representation.

Probably, the current method of visualization may appear somewhat confusing to readers. However, if the update frequency of our SNN model were to match the signal recording frequency of BCI electrodes (typically around 20 kHz or 30 kHz), then Fig. 5(k)-(l) would align with Fig. 5(j), resulting in rasterplots.

Nonetheless, it’s worth noting that BCI electrodes typically record signals at high frequencies, and running our SNN model at such frequencies would necessitate significantly more computational resources and time. To strike a balance between model efficiency and temporal alignment, we opt for an SNN model update frequency of 25 ms, consistent with the 25 ms time window of the spike train in the BCI dataset. This decision leads to the visual representation shown in our current Fig. 5(j)–(l).

In summary, the depiction in Fig. 5(j)–(l) neither compromises our conclusion that mixed neurons’ simulation results outperform those using only the other two neuron types nor does it affect the explanation for the superior simulation results when exclusively using biological neurons over Izhikevich neurons. The key reason is that within the same time interval, biological neurons fire more pulses than Izhikevich neurons, thus containing more information, creating a form of “burst spikes" effect, which ultimately results in improved simulation outcomes.

REFERENCES

1. W. Nicola and C. Clopath, “Supervised learning in spiking neural networks with force training,” *Nature Communications*, vol. 8, no. 1, pp. 1–15, 2017.