Equilibrium Point Learning

Dowoo Baik ARIZONA9506@GMAIL.COM
Naval Academy

Ji Won Yoon JIWONYOON@KOREA.AC.KR
Korea University

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

We present a novel approach, Equilibrium Point Learning (EPL), for training the deep equilibrium model (DEQ). In this method, the equilibrium point of the DEQ serves as the learnable parameters. Notably, the DEQ parameters encapsulate the learning algorithm itself and remain fixed. Consequently, by exploring the parameter space, we can discover a more efficient learning algorithm without relying on conventional techniques such as backpropagation or Q-learning. In this paper, we adopt an evolutionary approach inspired by biological neurons to evolve the DEQ model parameters. Initially, we examine the physical dynamics of neurons at the molecular level and translate them into a dynamical system representation. Subsequently, we formulate a deep implicit layer that is mathematically proven to possess an equilibrium point. The energy function of the implicit layer is defined using a quadratic form augmented with entropy and momentum terms. Given the resemblance between the dynamics of the deep implicit layer and the principles of physics and chemistry, it can effectively capture the biomodel of systems biology and the neural model of spiking neural networks (SNNs). This equivalence enables us to define the implicit layer of the DEQ, allowing for seamless integration with existing artificial neural networks (ANNs). Finally, we employ HyperNEAT to evolve the parameters of the dynamical system. Through our experiments, we observe a consistent improvement in learning efficiency, with each successive generation exhibiting a 0.2% increase in learning speed per generation.

Keywords: Deep equilibrium model, Learning Algorithm, Biomodel, HyperNEAT

1. Introduction

To describe biological neurons mathematically, there have been several academic attempts, including spiking neural networks, computational biology, and systems biology Pfeiffer and Pfeil (2018), Buesing et al. (2011), Bick et al. (2020). These endeavors aim to identify the function and role of neuronal components, such as gene-protein-reaction (GPR) rules Di Filippo et al. (2021). The ultimate goal of these efforts is to reach human intelligence and artificial general intelligence (AGI). In the field of systems biology, biological systems are mathematically analyzed and modeled as dynamical systems, often referred to as biomodels Chelliah et al. (2013). These biomodels can be simulated over time using ordinary differential equation (ODE) solvers, and well-crafted models exhibit similarities or comparable values to experimental observations Hernjak et al. (2005). However, due to the complexity and vastness of biological systems, many biological functions remain unexplained. In the context of spiking neural networks (SNNs), the problem of nonlinear learning is addressed using a backpropagation learning method, although it has not yet been fully biologically identified Tavanaei and Maida (2019).

From the perspective of artificial neural networks (ANNs), we hypothesized that biomodels share similarities with DEQ models. Additionally, simulating a biomodel is akin to the approach taken in neural ordinary differential equations (neural ODEs) Chen et al. (2018), which involves computing through an ODE solver in the context of weight-tied feedforward networks with infinitely many layers. However, learning in neurons through this method occurs without the need for separate learning algorithms such as backpropagation or Q-learning. In other words, the calculation of a deep implicit layer as an ODE solver constitutes the entire learning process. This is a capability that has not yet been achieved in ANN architectures. In this paper, we introduce a novel approach to learning the DEQ model, which we refer to as Equilibrium Point Learning (EPL). In EPL, the parameters of the model serve as the learning algorithms themselves, analogous to backpropagation in traditional approaches. We draw parallels to the learning process in biological neurons, where the search for optimal parameters can be seen as a favorable case. EPL encompasses elements of both supervised and unsupervised learning, and it distinguishes between the time for behavior determination and the time for feedback acceptance. This learning method aligns with the principles outlined in studies on neural learning and decision-making Glimcher (2011). By leveraging EPL, our objective is to explore the parameter space of the DEQ model and identify the most effective learning algorithm.

Biological neurons have previously explored the parameter space using genetic algorithms, aiming to identify dynamical systems that are conducive to achieving survival goals within their physical environment. It is this sequence of evolutionary processes that serves as our primary focus. In light of this, we introduce a dynamical system that can capture certain aspects of neuronal behavior, such as the Hodgkin-Huxley (HH) model in spiking neural networks (SNNs). Our approach centers around biological and chemical phenomena, seeking to depict the physical system and energy dynamics from a molecular perspective within the context of a dynamical system. By adopting this perspective, we can effectively represent the neuron models found in systems biology. Additionally, we reinterpret the dynamical system as a means of solving a convex optimization problem. Through this reinterpretation, we devise a deep implicit layer that is mathematically proven to possess an equilibrium point.

We further developed the deep implicit layer using a Neat algorithm, which is a type of artificial neural network evolution method. Specifically, we employed the HyperNeat algorithm, which is well-suited for dynamic systems that necessitate the creation of large-scale networks and the repetition of network patterns.

2. Related Work

Deep Equilibrium model The DEQ model Bai et al. (2019) focuses on a weight-tied deep implicit layer, which involves finding the equilibrium point through an infinite layer via root finding. In contrast to conventional deep implicit layers, the DEQ model enables analytical backpropagation through the equilibrium point using implicit differentiation. This approach requires only a single layer of information, resulting in lower memory usage. In this paper, we adopt the same architecture as the DEQ model and compare our learning methods with it.

Training of Spiking Neural Network The Spiking Neural Network (SNN) is a neural network model that draws inspiration from the spiking behavior observed in biological neurons during information processing. SNN encompasses various neuron models, including the Hodgkin-Huxley (HH) model Cronin and Boutelle (1987), which closely resembles computational biology by simulating sodium and potassium channels, and the Izhikevich model Izhikevich (2003), which exhibits similar behavior with lower computational complexity. In this paper, our focus lies in designing an implicit layer that can accommodate these dynamical system models. SNN incorporates parallel and biologically plausible learning rules such as Spike-Timing-Dependent Plasticity (STDP) and the Bienenstock-Cooper-Munro (BCM) theory, along with extensive training methods like backpropagation. In a previous study Huang and Huang (2016), researchers evolved the best-fit model constants by controlling the neuron model constants. Neurons are known to learn through Hebbian learning, an unsupervised learning paradigm exemplified by STDP Caporale and Dan (2008). More recently, a method was proposed in Xiao et al. (2021) that applies implicit differentiation of DEQ to backpropagation in the equilibrium state. This suggests a way to incorporate DEQ's implicit differentiation into the training process.

Training of Biological Neural Network In the study by Glimcher (2011), it is demonstrated that nerves are activated by both an activation signal and a feedback signal mediated by dopamine. As the learning progresses, the feedback signal gradually diminishes due to the ability to predict the reward signal.

Taylor and Ivry (2014) present research on reinforcement learning algorithms in the cerebellum. If there exists a dynamical system that is intricately represented by biomodels such as purkinje cells, granule cells, and star cells in the cerebellum, this suggests that the dynamical system may implicitly incorporate a Q-learning algorithm.

HyperNeat NEAT Stanley and Miikkulainen (2002) is a genetic algorithm that evolves artificial neural networks (ANNs). It aims to strike a balance between fitness and diversity, allowing for the evolution of species within the network. HyperNEAT (Stanley et al., 2009) utilizes a method of network creation known as a Connection Pattern Producing Network (CPPN), which is evolved by NEAT. HyperNEAT employs an indirect encoding approach that leverages the geometric properties of the network. This enables the design of regular patterns within the network structure.

3. Proposed EPL Method

In this section, we first provide an overview of the Equilibrium Point Learning (EPL) approach and introduce its core concepts. We then provide a detailed explanation of the EPL method in comparison with the learning methodology of DEQ. Additionally, we briefly discuss the training of model parameters in EPL learning and explain how to align it with the back-propagation algorithm, which is well known as a learning method in ANN. Next, we introduce the properties of the implicit layer required in EPL learning. Finally, we introduce how to design an implicit layer that convergence (a prerequisite for EPL learning) of the model is mathematically proven. It can also accommodate the biomodel in system biology.

3.1. Overview of the EPL Learning

Equilibrium Point Learning (EPL) works as schematically shown in Fig. 1. The key is to expect all learning to take place while overwriting $\boldsymbol{z}$ to $\boldsymbol{z}^*$. In the context of traditional artificial neural networks (ANN), The role of the learning parameter shifts from θ to z, and the model's output inherently includes the parameter learning amount, with this role attributed to the fixed parameter θ. Therefore, exploring the parameter space of θ implies searching for a more efficient learning algorithm.

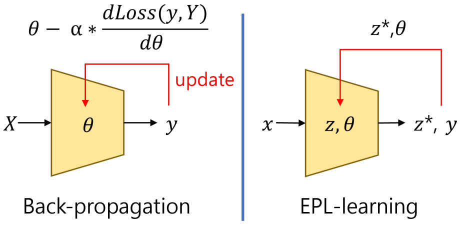
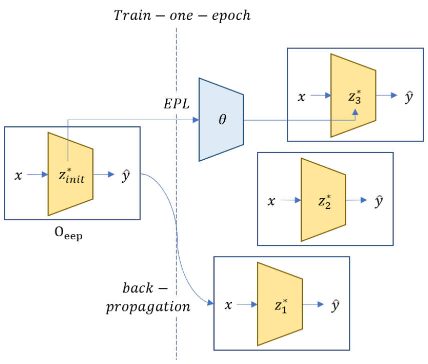


Figure 1: Concept of EPL Learning

3.2. EPL Learning Process

Algorithm 1 Compare Learning Algorithm from DEQ

```
procedure Train DEQ $f_\theta(\boldsymbol{z}, \boldsymbol{x})$
    while epoch do
        $\boldsymbol{z}^{post} \leftarrow \boldsymbol{x}$
        while $|\boldsymbol{z}^{post} - \boldsymbol{z}^{pre}| > \varepsilon$ do
            $\boldsymbol{z}^{pre} \leftarrow \boldsymbol{z}^{post}$
            $\boldsymbol{z}^{post} = f_\theta(\boldsymbol{z}^{pre}, \boldsymbol{x})$
        end while
        $\boldsymbol{z}^* \leftarrow \boldsymbol{z}^{post}$
        compute $\ell(\boldsymbol{z}^*, \boldsymbol{y})$
        $\theta \leftarrow \theta - \alpha\nabla\ell$
    end while
end procedure
```



```
procedure Train EPL $f_\theta(\boldsymbol{x})$
    $\boldsymbol{z}^{post} \leftarrow \theta[\boldsymbol{x}_0]$
    while $|\boldsymbol{z}^{post} - \boldsymbol{z}^{pre}| > \varepsilon$ do
        $\boldsymbol{z}^{pre} \leftarrow \boldsymbol{z}^{post}$
        $\boldsymbol{z}^{post} = f_\theta(\boldsymbol{z}^{pre})$
    end while
    $\boldsymbol{z}^* \leftarrow \boldsymbol{z}^{post}$                                ▷ $\boldsymbol{z}^*_{init}$
    while epoch do
        $\boldsymbol{z}^{post} \leftarrow \boldsymbol{z}^* + \boldsymbol{I}_z \cdot \boldsymbol{x}$
        $\widehat{\boldsymbol{y}} = emptylist$
        while $|\boldsymbol{z}^{post} - \boldsymbol{z}^{pre}| < \varepsilon$ do
            $\widehat{\boldsymbol{y}}.append(O_z \cdot \boldsymbol{z}^{post})$
            $\boldsymbol{z}^{pre} \leftarrow \boldsymbol{z}^{post}$
            $\boldsymbol{z}^{post} = f_\theta(\boldsymbol{z}^{pre})$
        end while
        $\boldsymbol{z}^* \leftarrow \boldsymbol{z}^{post}$                        ▷ $\boldsymbol{z}^*_{self-learn}$
        $\ell \leftarrow \ell(Active(\widehat{\boldsymbol{y}}), \boldsymbol{y})$          ▷ $\ell(O_{eep}, \boldsymbol{y})$
        $\boldsymbol{z}^{post} \leftarrow \boldsymbol{z}^* + \boldsymbol{F}_z \cdot \ell$
        while $|\boldsymbol{z}^{post} - \boldsymbol{z}^{pre}| > \varepsilon$ do
            $\boldsymbol{z}^{pre} \leftarrow \boldsymbol{z}^{post}$
            $\boldsymbol{z}^{post} = f_\theta(\boldsymbol{z}^{pre})$
        end while
        $\boldsymbol{z}^* \leftarrow \boldsymbol{z}^{post}$                ▷ $\boldsymbol{z}^*_{supervised-learn}$
    end while
end procedure
```

In the learning process of the DEQ, equilibrium point $\boldsymbol{z}^* \in \mathbb{R}^{t\times 1}$ is the output of input value $\boldsymbol{x} \in \mathbb{R}^{t\times 1}$. Then, calculate the loss of $\boldsymbol{z}^*$ and $\boldsymbol{y} \in \mathbb{R}^{t\times 1}$, and update the parameter of the model that in the direction of the loss decrease. In comparison, EPL training presented in this paper, at first, find the first equilibrium point $\boldsymbol{z}^*_{init} \in \mathbb{R}^{n\times 1}$ from the given initial value $\boldsymbol{x}_0 \in \mathbb{R}^{n\times 1}$ in model parameter, like Eq. (9). This becomes the first pre-learned parameter. Next, using the input constant matrix $\boldsymbol{I}_z \in \mathbb{R}^{n\times t}, n > t$, the vector $\boldsymbol{z}^* + \boldsymbol{I}_z \cdot \boldsymbol{x}$ is the next input for the layer. This is a key difference between DEQ and EPL methods. When $\boldsymbol{z}$ reaches the new equilibrium point $\boldsymbol{z}^*_{self-learn}$, it is the self-learning process of the equilibrium point. Until equilibrium point $\boldsymbol{z}^*_{self-learn}$ is reached, we collect $\boldsymbol{O}_z \cdot \boldsymbol{z}^{post}$, that the inner value of output constant matrix $\boldsymbol{O}_z \in \mathbb{R}^{r\times n}$ and $\boldsymbol{z}$. Then, output becomes the value taken by its activation function (like the average or the number of times the threshold has been exceeded). We name the output for these inputs as *Output of Exploration Equilibrium Point*, that is, O_{eep}. In addition, the function that represents the newly reached equilibrium point is named *Function of New Equilibrium Point*, f_{nep} (Eq. (1)).

$$\begin{aligned} \forall \boldsymbol{z}, \quad \boldsymbol{z}_0 \leftarrow \boldsymbol{z}, \quad \boldsymbol{z}_{i+1} = f_\theta(\boldsymbol{z}_i) \\ find \quad * \in \mathbb{N} \quad s.t \; |f_\theta(\boldsymbol{z}_*) - \boldsymbol{z}_*| \le \epsilon, \\ Let \quad O_{eep}(\boldsymbol{z}) = Active((\boldsymbol{O}_z \cdot \boldsymbol{z}_i)_{[0:*]}), \quad f_{nep}(\boldsymbol{z}) = \boldsymbol{z}_* \end{aligned} \tag{1}$$

These output values can be an action in the environment and obtain a corresponding fitness score, or a feedback (loss) value $\ell = \ell(O_{eep}, \boldsymbol{y}) \in \mathbb{R}^{1\times 1}$ can be obtained by calculating the distance from the expected output value $\boldsymbol{y}$. Then, use the feedback constant $F_z \in \mathbb{R}^{n\times 1}$, so that vector $\boldsymbol{z}^* + F_z \cdot \ell$, which is a linear combination of the equilibrium point $\boldsymbol{z}^*_{self-learn}$ and the feedback, is the next input of the layer. $\boldsymbol{z}$ reaches another new equilibrium point $\boldsymbol{z}^*_{supervised-learn}$, which becomes the supervision-learning process of the equilibrium point. One learning ends when a new equilibrium point is reached.

3.3. Find Back-Propagation Algorithm Parameter of EPL Learning

If EPL learning occurs (if the loss value is reduced), this means that Eq. (2). must be satisfied.

$$\ell(O_{eep}(\boldsymbol{z}^*_{pre-train} + \boldsymbol{I}_z \cdot \boldsymbol{x}), \boldsymbol{y}) > \ell(O_{eep}(\boldsymbol{z}^*_{post-train} + \boldsymbol{I}_z \cdot \boldsymbol{x}), \boldsymbol{y}) \tag{2}$$

If EPL learning expects to learn, such as back-propagation algorithm, in which all steps of epoch flow exactly to a gradient that reduces loss, the conditions to be satisfied can be written as Eq. (3).

$$-\Delta \boldsymbol{z}^* = \boldsymbol{z}^* - f_{nep}(f_{nep}(\boldsymbol{z}^* + \boldsymbol{I}_z \cdot \boldsymbol{x}) + \boldsymbol{F}_z \cdot \ell) = \alpha(\frac{\partial \ell(O_{eep}(\boldsymbol{z}^* + \boldsymbol{I}_z \cdot \boldsymbol{x}), \boldsymbol{y})}{\partial \boldsymbol{z}^*})^T \tag{3}$$

This means that if the purpose of the model parameter is to find the back-propagation algorithm, if $\boldsymbol{I}_z, \boldsymbol{O}_z$ is small, it is to reduce the cosine-similarity like this Eq. (4).

$$\begin{aligned} Let \quad \boldsymbol{A} = -\Delta \boldsymbol{z}^* \approx -\frac{\partial f_{nep}(\boldsymbol{z}^* + \boldsymbol{z})}{\partial \boldsymbol{z}}(\boldsymbol{I}_z \cdot \boldsymbol{x} + \boldsymbol{F}_z \cdot \ell), \; \boldsymbol{B} = (\frac{\partial \ell(O_{eep}(\boldsymbol{z}^* + \boldsymbol{I}_z \cdot \boldsymbol{x}), \boldsymbol{y})}{\partial \boldsymbol{z}^*})^T \\ \mathcal{L}(\theta, back - propagation - algorithm) = \frac{\boldsymbol{A} \cdot \boldsymbol{B}}{|\boldsymbol{A}|\,|\boldsymbol{B}|} \end{aligned} \tag{4}$$

In this paper, we explore θ spaces in which numerous training methods exist, including back-propagation algorithm.

3.4. Property of EPL Learning

EPL learning occurs by repeatedly passing through implicit layers infinitely. During this process, if even a single value of the vector starts to diverge, it becomes impossible to compute the output of the implicit layer. Therefore, the implicit layer requires a *Non-diverging* property. In this paper, we can achieve this with the following strategy:

1. We divide the implicit layer into two stages: first passing through the local implicit layer and then the external implicit layer.
2. For every input space, there exists a function that maps it to a single scalar energy.
3. The possible input space of the implicit layer cannot have infinite negative energy.
4. The output of the local implicit layer always has lower energy than the input.
5. At this time, the amount of reduction is proportional to the magnitude of the vector's components. In other words, if even one value is infinite, the amount of energy reduction becomes infinite.
6. The external implicit layer always flows finite energy with an upper limit.

3.5. Design Implict Layer

We propose a dynamical system $\frac{d\boldsymbol{z}}{dt} = F_\theta(\boldsymbol{z}),\ \boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{p}),\ \boldsymbol{x} \in \mathbb{R}^{n\times 1}_{>0}, \boldsymbol{p} \in \mathbb{R}^{n\times 1}$, compatible with biomodel of system biology. This can be treated as implicit layer f_θ follows: $\frac{d\boldsymbol{z}}{dt} = F_\theta(\boldsymbol{z}) \ \Leftrightarrow\ \boldsymbol{z}^{i+1} = \boldsymbol{z}^i + dt \cdot F_\theta(\boldsymbol{z}^i) = f_\theta(\boldsymbol{z}^i) \ \Rightarrow\ \boldsymbol{z}^* = \boldsymbol{z}^* + dt \cdot F_\theta(\boldsymbol{z}^*) \ \Leftrightarrow\ F_\theta(\boldsymbol{z}^*) = 0$ We prove mathematically convergence of the dynamical system. In addition, we propose a reduced dynamical system that can significantly reduce the computation of the implicit layer.

3.5.1. Energy

The energy function of the dynamical system (DS) is defined based on the actual physical quantity. Consider a node by grouping the same states (component, composition state, momentum, direction of motion, position) of molecules. Each node in DS has a mass and momentum, expressed as $\boldsymbol{x}$ and $\boldsymbol{p}$. DS represents the sum of four energies: $H(\boldsymbol{x}, \boldsymbol{p}) = -S_{entrophy} + H_{enthalpy} + U_{electric} + E_{kinetic}$. From a physical and chemical point of view, physical energy in a system with a *constant temperature* can be expressed. The presented form of the energy function Eq. (5). is consequently an extension of entropy and physical quantity in quadratic function Nocedal and Wright (2006). Low et al. (1973) This allows us to explain the laws and models of biology and chemistry. ($\boldsymbol{x}^T$: Transpose of vector, $\boldsymbol{x} \circ \boldsymbol{y}$: Hadamard product)

$$H(\boldsymbol{x}, \boldsymbol{p}) = \boldsymbol{x}^T \ln(\frac{\boldsymbol{x}}{\boldsymbol{x}^*}) + \frac{1}{2}\boldsymbol{x}^T \boldsymbol{V} \boldsymbol{x} + \frac{1}{2\boldsymbol{m}^T}(\boldsymbol{p} \circ \boldsymbol{p} \circ \boldsymbol{x}) \tag{5}$$

Entrophy & Enthalpy $-S_{entropy}(\boldsymbol{x}, \boldsymbol{p}) + H_{enthalpy}(\boldsymbol{x}, \boldsymbol{p}) = \boldsymbol{x}^T \ln(\frac{\boldsymbol{x}}{\boldsymbol{x}^*}) = \boldsymbol{x}^T \ln(\boldsymbol{x}) + \boldsymbol{a}^T \boldsymbol{x}$ The probability that a single molecule exists in that state, by Arrhenius equation, is inversely proportional to its energy state, the enthalpy $\boldsymbol{x}^* \propto e^{-\boldsymbol{E}_x}$. In this paper, we replace the energy constant with $\boldsymbol{a} \in \mathbb{R}^{n\times 1}$. The molecular chemical reaction network (CRN) can be described as a flow that reduces the defined Gibbs free energy $H - S$. However, when electric Potential Energy and kinetic Energy (the energy of the other two energy terms U, E) change, It means that the energy state of the molecule ($\frac{\partial H}{\partial \boldsymbol{x}}$) changes, which also changes the probability of a molecule's presence. This explains the change in the activation rate according to the voltage of the neuron's voltage-gated sodium channels de Lera Ruiz and Kraus (2015).

Electric Potential Energy $U_{electric}(\boldsymbol{x}, \boldsymbol{p}) = \frac{1}{4\pi\epsilon_0}\boldsymbol{q}_x^T \frac{1}{\boldsymbol{R}}\boldsymbol{q}_x = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{V}\boldsymbol{x}, \boldsymbol{V} = \frac{diag(\boldsymbol{q}_c)^2}{2\pi\epsilon_0\boldsymbol{R}} \in \mathbb{R}^{n \times n}$ Electrical potential energy is related to the movement that creates an electric current. The movement of electrons perpendicular to the gradient of the energy function, the movement of the hamiltonian flow, and the movement of the longitudinal velocity at which electrons collide with protons as they move, are current and resistance Vilasi (2001).

Kinetic Energy $E_{kinetic}(\boldsymbol{x}, \boldsymbol{p}) = \frac{1}{2\boldsymbol{m}^T}(\boldsymbol{p} \circ \boldsymbol{p} \circ \boldsymbol{x})$ This is the momentum energy of the molecules. $\boldsymbol{m} \in \mathbb{R}^{n \times 1}_{>0}$ corresponds to the mass of the molecule. Electrical potential energy is converted to kinetic energy due to hamiltonian flow.

3.5.2. Flow

We define symbol $\boldsymbol{M} \in \mathbb{R}^{n \times c}_{\geq 0}$ that is the elemental matrix of $\boldsymbol{x}$, and $\boldsymbol{M}^{\perp} \in \mathbb{R}^{n \times e}$ that is stoichiometric metrics. They are orthogonal ($\boldsymbol{M}^T\boldsymbol{M}^{\perp} = 0_{c \times e}$) to each other. Using this, we define a total of four types of flows.

Total Flow The total flow is as per the equation below ($r(\boldsymbol{x}) := relu(\boldsymbol{x})$), This means dynamical system.

$$\begin{aligned} \frac{d\boldsymbol{x}}{dt} &= \boldsymbol{M}^{\perp}\left[-\boldsymbol{k} \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} r(\boldsymbol{M}^{\perp})} - e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})})^T + \boldsymbol{v} \circ (\tfrac{\partial H}{\partial p}\boldsymbol{M}^{\perp})^T \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T\right] + \boldsymbol{h} \circ \boldsymbol{x} \circ (e^{-\frac{\partial H}{\partial \boldsymbol{x}}} - 1)^T \\ \frac{d\boldsymbol{p}}{dt} &= -\boldsymbol{M}^{\perp}\left[\boldsymbol{v} \circ (\tfrac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp})^T \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T\right] - \boldsymbol{c} \odot (\tfrac{\partial H}{\partial \boldsymbol{p}})^T \end{aligned} \tag{6}$$

The gradient ($\frac{\partial H}{\partial \boldsymbol{x}} \in \mathbb{R}^{1 \times n}$, $\frac{\partial H}{\partial \boldsymbol{p}} \in \mathbb{R}^{1 \times n}$) at point $(\boldsymbol{x}, \boldsymbol{p})$ is calculated as Eq. (7).

$$\frac{\partial H}{\partial \boldsymbol{x}} = (\ln(\boldsymbol{x}) + 1)^T + \boldsymbol{a}^T + \boldsymbol{x}^T\boldsymbol{V} + \left(\frac{\boldsymbol{p} \circ \boldsymbol{p}}{2\boldsymbol{m}}\right)^T \quad \frac{\partial H}{\partial \boldsymbol{p}} = (\frac{\boldsymbol{p} \circ \boldsymbol{x}}{\boldsymbol{m}})^T \tag{7}$$

Chemical Flow $\frac{d\boldsymbol{x}}{dt} = -\boldsymbol{M}^{\perp}\left[-\boldsymbol{k} \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} r(\boldsymbol{M}^{\perp})} - e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})})^T\right], \quad \boldsymbol{k} \in \mathbb{R}^{e \times 1}_{\geq 0}$
This flow can explain chemical reaction and diffusio Hochberg and Ribó (2018).

Hamiltonian Flow
$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{M}^{\perp}\left[\boldsymbol{v} \circ (\frac{\partial H}{\partial p}\boldsymbol{M}^{\perp})^T \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T\right], \frac{d\boldsymbol{p}}{dt} = -\boldsymbol{M}^{\perp}\left[\boldsymbol{v} \circ (\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp})^T \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T\right]$
$\boldsymbol{v} \in \mathbb{R}^{e \times 1}_{>0}$ This flow can account for the current. With hamiltonian flow alone, it's like a current without resistance. This preserves the energy of the entire system. It's a term that's never considered in the problem of finding the solution of the convex optimization flow, but in real biology, this flow causes the flow of electrons. It's important to play an algorithmic role in going to an increasingly learned fixed point, not just looking for fixed points.

Collision Flow $\frac{d\boldsymbol{p}}{dt} = -\boldsymbol{c} \odot (\frac{\partial H}{\partial \boldsymbol{p}})^T,\ \boldsymbol{c} \in \mathbb{R}^{e\times 1}_{>0}$ It means that the momentum decreases due to the collision of molecules. Unlike hamiltonian flow, these flows reduce the energy of the system. In the actual physical phenomenon, just as heat energy is emitted from resistance, it is transformed into the translational or rotational motion of molecules, the thermal energy. But these temperatures quickly spread around. Therefore, it can reduce the amount of computation by ignoring this term.

External Homeostasis Flow $\boldsymbol{h} \circ (e^{\ln(\boldsymbol{x})-(\frac{\partial H}{\partial \boldsymbol{x}})^T} - \boldsymbol{x}), \quad \boldsymbol{h} \in \mathbb{R}^{n\times 1}_{\geq 0}$ Living things need endless sources of energy from the outside to match the homeostasis of ATP-ADP energy sources. To accommodate this, we treat that $\boldsymbol{x}_{[i:n]}$, part of $\boldsymbol{x}$, is connected to the external system. Their flow is expressed by the constant h. In other words, it satisfies $\boldsymbol{h}_{[0:i-1]} = 0$, and $\boldsymbol{h}_{[i:n]}$ indicates the speed of the external flow. Unlike the three flows above, only substances that are not connected to the external system satisfy the law of conservation of mass. $\frac{d}{dt}(\boldsymbol{M}_h^T\boldsymbol{x}) = 0$

3.5.3. Convex Optimization Problem

All defined flows can be explained by the direction of decreasing the proposed system's energy function $H(\boldsymbol{x}, \boldsymbol{p})$ by proving to Appendix A. Also, $\boldsymbol{x}$ has a convex domain that satisfies both the law of conservation of mass : $= \frac{d}{dt}(\boldsymbol{M}_h^T\boldsymbol{x}) = 0$ and positive of mass. So, their flow can be reinterpreted as the solving convex optimization problem Eq. (8).

$$\begin{aligned} &\underset{\boldsymbol{x},\boldsymbol{p}}{min} && H(\boldsymbol{x}, \boldsymbol{p}) \\ &subject\ to && \boldsymbol{x} > 0, \quad \boldsymbol{M}_h^T\boldsymbol{x} = \boldsymbol{C}_h \end{aligned} \tag{8}$$

By solving the finite convex set, the energy satisfies the bounded condition. Through energy reduction and bounded conditions, mathematically satisfied that it always converges to the equilibrium point.

3.5.4. Reduced Dynamical System

We will solve the system defined above with ODE solver to find the equilibrium point. However, the computational difficulty is high due to frequent log and exp functions. Therefore, we present a reduced dynamical system (RDS), which is a scale-down model of the defined dynamical system.

$$RDS := F_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{p}), \quad \boldsymbol{\theta} = [\boldsymbol{x}_0, \boldsymbol{M}, \boldsymbol{M}^{\perp}, \boldsymbol{S}, \boldsymbol{D}, \boldsymbol{m}_c, \boldsymbol{q}_c, \boldsymbol{a}, \boldsymbol{k}, \boldsymbol{v}, \boldsymbol{c}, \boldsymbol{h}] \tag{9}$$

Give each node its genetic and spatial information. This will become a substrate in HyperNeat as the system evolves in the future. Most log and exp calculations can be converted to multiplication functions by grouping the spaces of nodes.

Gene of Entity For all $\boldsymbol{x}_i \in \mathbb{X}$, the function $f_g : \mathbb{X} \mapsto \mathbb{G} = \{\boldsymbol{g}_1, \boldsymbol{g}_2, ..., \boldsymbol{g}_m\}, \boldsymbol{g}_i \in \mathbb{R}^{g\times 1}$ maps to the genetic space $\mathbb{G}$. For all $\boldsymbol{g}_i$, the function $f_c : \mathbb{G} \mapsto \mathbb{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_k\}, \boldsymbol{c}_i \in \mathbb{R}^{c\times 1}$ maps to the component space $\mathbb{C}$. Each component has a mass and a charge, which is expressed as $\boldsymbol{m}_c \in \mathbb{R}^{c\times 1}_{\geq 0}$ and $\boldsymbol{q}_c \in \mathbb{R}^{c\times 1}$. That is, the mass and charge of $\boldsymbol{x}_i$ are $f_c(f_g(\boldsymbol{x}_i))^T \cdot \boldsymbol{m}_c$ and $f_c(f_g(\boldsymbol{x}_i))^T \cdot \boldsymbol{q}_c$. For any gene $\boldsymbol{g}_i, \boldsymbol{g}_j, \boldsymbol{g}_k$, Whether a chemical reaction $\boldsymbol{g}_i + \boldsymbol{g}_j \Rightarrow \boldsymbol{g}_k$ is satisfied can be determined as a reaction function f_r, that $f_r(\boldsymbol{g}_i, \boldsymbol{g}_j) = \boldsymbol{g}_k$ and $f_c(\boldsymbol{g}_i) + f_c(\boldsymbol{g}_j) = f_c(\boldsymbol{g}_k)$

. In RDS, only two reactants (two may be the same) are allowed to create one product. The Na+ channel of the HH model can operate as a function of the active element only when three active factors are combined. Such a multi-molecular product and the like can be expressed by the markov model chemical reaction network, such as Fink and Noble (2009).

Space of Entity Every $\boldsymbol{x}_i \in \mathbb{X}$ has their own space $\boldsymbol{s_i}$, a function that maps it is defined as follows: $f_s : \mathbb{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\} \mapsto \mathbb{S} = \{\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_s\}$. Let a space matrix $\boldsymbol{S} \in \mathbb{R}^{n\times s}_{\geq 0}$ as follows: $\boldsymbol{S}_{ij} := \begin{cases} 0 \quad if \quad f_s(\boldsymbol{x}_i) \neq \boldsymbol{s}_j \\ 1 \quad if \quad f_s(\boldsymbol{x}_i) = \boldsymbol{s}_j \end{cases}$. Let a distance matrix $\boldsymbol{D} \in \mathbb{R}^{s\times s}_{\geq 0}$, This is the matrix that determines the distance between spaces. For computational gain, we make the inverse of these distance matrix $\frac{1}{\boldsymbol{D}}$ sparse matrix. In other words, it makes the distance of the neighboring spaces infinite, leaving them out of the calculate electrical potential energy. With the newly defined gen-space information, the electric potential matrix $\boldsymbol{V}$ can be calculated as follows: $\boldsymbol{q} = diag((\boldsymbol{M}\boldsymbol{q}_c)^T)\boldsymbol{S}, \quad \boldsymbol{V} = \boldsymbol{q}(\frac{1}{\boldsymbol{D}})\boldsymbol{q}^T$

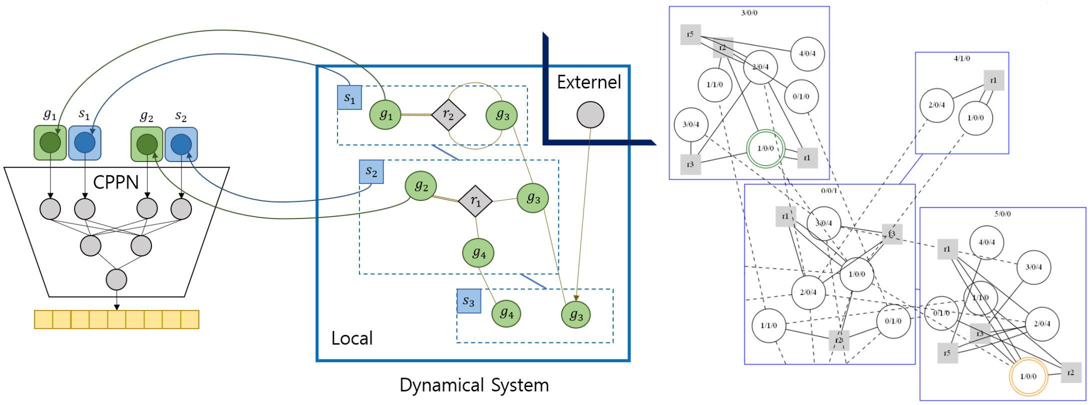
Reduce Flow and Calculate Depending on whether the space of the components is the same or different, an allowable flow is determined. For two elements of the same space, only the chemical flow of nodes capable of chemical reaction is acceptable In other spaces, diffusion flow and hamiltonian flow of nodes with the same genetic information are possible. Through these constraints, the calculation volume can be significantly reduced for three reasons.

1. $\boldsymbol{D}$ becomes sparse matrix, so $\boldsymbol{V}$ also becomes sparse matrix.

2. In the chemical reaction, the flow in the same space occurs, so electric potential energy is not changed. $\because \frac{dU_{electric}}{d(chemical\ react)} \propto \frac{d\boldsymbol{x}^T\boldsymbol{q}}{d(chemical\ react)} \propto \boldsymbol{M}^{\perp}_{ch} diag((\boldsymbol{M}\boldsymbol{q}_c)^T)\boldsymbol{S} \propto \boldsymbol{M}^{\perp}_{ch}\boldsymbol{M} = 0_{ch\times s}$

3. Since the chemical flow has two reactants, one product, log and expterm can be replaced by one multiplication. (ex) In the reaction of $a + b \Leftrightarrow c$, the existing one is $e^{\ln(a)+\ln(b)} - e^{\ln(c)}$ If so, it can be calculated by replacing it with $a * b - c$)

4. Experiment



(a) Using space and gene information to HyperNeat substrate

(b) Example of DS

Figure 2: Generation Dynamical System by HyperNeat and Example

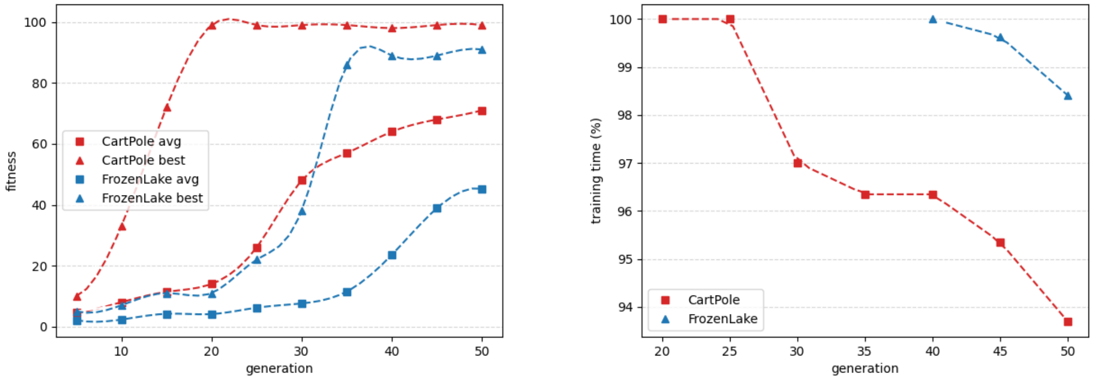
When biological cells undergo expression, the configuration of the dynamical system is influenced by the concentration of surrounding substances, and it exhibits symmetry within the same cell type. To leverage this characteristic, we employ HyperNeat (Fig. 2(a)subfigure) for evolving the dynamical system, which enables the design of regular patterns such as symmetry, repetition, and variation. By selectively disabling specific components of the CPPN input, we define the necessary functions.

Algorithm 2 Expression Algorithm

```
procedure EXPRESSION BY $CPPN(\boldsymbol{g}_i, \boldsymbol{s}_i, \boldsymbol{g}_i, \boldsymbol{s}_i)$
    $\boldsymbol{g}_i$ exist $\leftarrow \tanh^{-1}(CPPN(\boldsymbol{g}_i, 0, \boldsymbol{g}_i, 0))[0] > 0$        ▷ Generation gene map
    $\boldsymbol{s}_i$ exist $\leftarrow \tanh^{-1}(CPPN(0, \boldsymbol{s}_i, 0, \boldsymbol{s}_i))[0] > 0$        ▷ Generation Space map
    node $(\boldsymbol{g}_i, \boldsymbol{s}_i)$ exist $\leftarrow \tanh^{-1}(CPPN(\boldsymbol{g}_i, \boldsymbol{s}_i, \boldsymbol{g}_i, \boldsymbol{s}_i))[0] > 0$        ▷ Generation node
    reaction function $f_r \leftarrow abs(\tanh^{-1}(CPPN(\boldsymbol{g}_i, 0, \boldsymbol{g}_j, 0)[0])$
    node $(\boldsymbol{g}_i, \boldsymbol{s}_i)$ connected externel nodes $\leftarrow \tanh^{-1}(CPPN(\boldsymbol{g}_i, \boldsymbol{s}_i, \boldsymbol{g}_i, \boldsymbol{s}_i))[1] > 0$
    space $(\boldsymbol{s}_i), (\boldsymbol{s}_j)$ is neighborhood $\leftarrow \tanh^{-1}(CPPN(0, \boldsymbol{s}_i, 0, \boldsymbol{s}_j))[1] > 0$
    charge of $(\boldsymbol{c}_i) \leftarrow \tanh^{-1}(CPPN(\boldsymbol{c}_i, 0, 0, 0))[2]$
    mass of $(\boldsymbol{c}_i) \leftarrow abs(\tanh^{-1}(CPPN(0, 0, \boldsymbol{c}_i, 0))[2])$
    k or non-exist$(< 0)$ of chemical reaction edge $\leftarrow abs(\tanh^{-1}(CPPN(\boldsymbol{g}_i, \boldsymbol{s}_i, \boldsymbol{g}_j, \boldsymbol{s}_i))[2])$
    v or non-exist$(< 0)$ of hamiltonian edge $\leftarrow abs(\tanh^{-1}(CPPN(\boldsymbol{g}_i, \boldsymbol{s}_i, 0, \boldsymbol{s}_j))[2])$
    k or non-exist$(< 0)$ of diffusion edge $\leftarrow abs(\tanh^{-1}(CPPN(0, \boldsymbol{s}_i, \boldsymbol{g}_i, \boldsymbol{s}_j))[2])$
end procedure
```

The dynamical system (DS) is shown in Fig. 2(b)subfigure. It consists of a blue square region with nodes as circles. Input nodes are green, the output node is orange, and nodes with chemical reactions are gray squares labeled with the reaction type. Connections exist between nearby spaces. Diffusion flow uses solid lines and Hamiltonian flow uses dotted lines, but the external flow isn't visually shown; it's stored as a model parameter.

5. Result



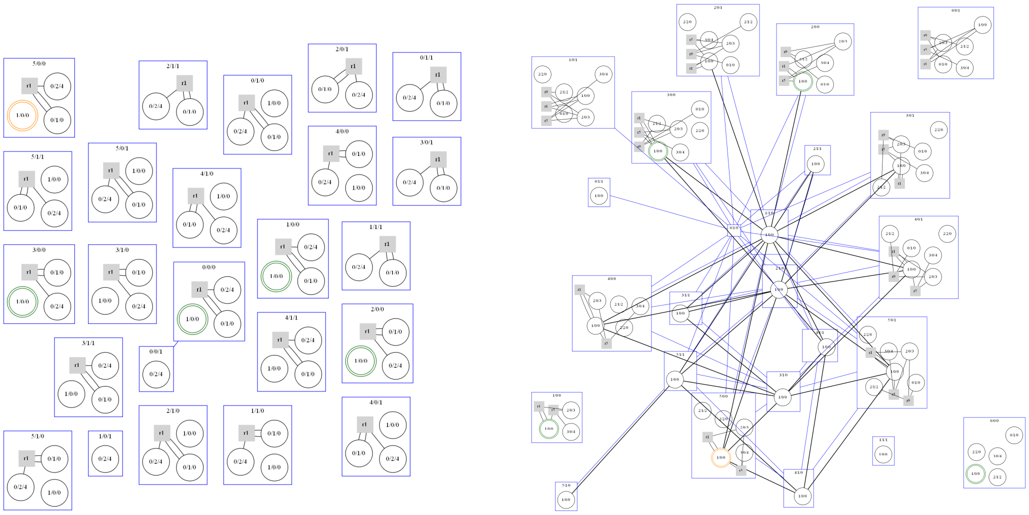
(a) Average(avg) and best result per generation

(b) Reduced training time (%)

Figure 3: Training results for two environments

We conducted experiments using the CartPole-v1 and FrozenLake-v0 environments from OpenAI. A total of 100 populations were evolved over 50 generations, utilizing an RTX 2070 GPU. The entire process took approximately 87 hours. We evaluated the fitness of the generated dynamical systems (DS) after 10 epochs of learning.

The results showed that the CartPole environment began to learn effectively around generations 10-20, while FrozenLake showed significant progress around generations 30-40. Furthermore, once the goal (including environment reward, learning speed, and memory usage) was achieved, we observed an average reduction of 0.2 in learning time per generation.

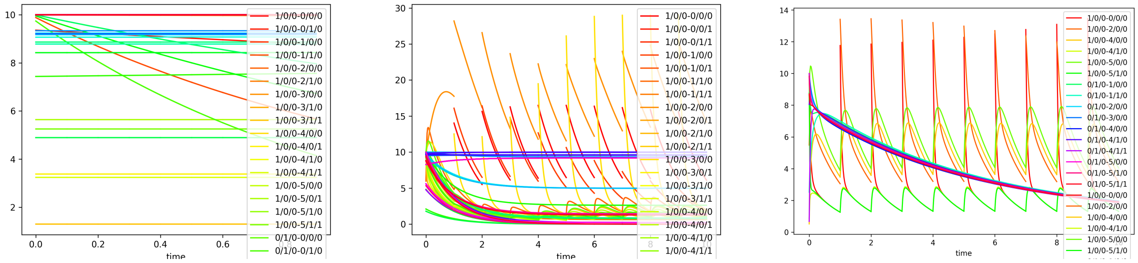


(a) Dynamical system example in gen 1

(b) Dynamical system example in gen 10

Figure 4: Examples of Dynamical systems from the 1st and 10th generations

The initial generations 1-4 consisted of simple repetitive structures where input and output nodes were not even connected (Fig. 4(*a*)subfigure). From generations 5-9, individuals that exhibited a positive impact between their input and output nodes were favored for survival. In generations 10-20, more diverse chemical reaction patterns emerged, and N:N multiple connections were identified (Fig. 4(*b*)subfigure).



(a) Gen 1

(b) Gen 10

(c) Gen 50

Figure 5: Best Fitness Examples from the 1st, 10th, and 50th Generations

Surprisingly, the output responses began to resemble the spiking neuron model (Fig. 5(b)subfigure). By generations 40-50, spaces started clustering, forming units resembling neurons. Additionally, the output responses became more sensitive to the input, resembling an adaptive leaky integrate-and-fire (LIF) model similar to that of spiking neural networks (SNN) (see Fig. 5(c)subfigure for an example at generation 50).

6. Conclusion

We have examined the biological properties and learning mechanisms of neurons and proposed a novel learning approach for artificial neural networks (ANNs). In this approach, the parameters of the model inherently represent the essence of the learning algorithm itself. Consequently, we conducted an evolutionary search for model parameters to discover alternative training methods that do not rely on traditional algorithms like back-propagation. Through this evolutionary process, we identified dynamical systems that are evolutionarily selected, indicating the presence of learnable implicit layers. By improving the efficacy of the evolutionary algorithm, we can accelerate the ODE solver's ability to discover more effective learning strategies. Ultimately, our aim is to uncover solutions to previously unsolvable problems using conventional learning methods.

Acknowledgments

Ji Won Yoon was supported by the MSIT(Ministey of science and ICT), Korea under the ITRC (Information Technology Research Center) support program (IITP-2023-RS-2022-00164800) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation)

References

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in Neural Information Processing Systems*, 32, 2019.

Christian Bick, Marc Goodfellow, Carlo R Laing, and Erik A Martens. Understanding the dynamics of biological and neural oscillator networks through exact mean-field reductions: a review. *The Journal of Mathematical Neuroscience*, 10(1):1–43, 2020.

Lars Buesing, Johannes Bill, Bernhard Nessler, and Wolfgang Maass. Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons. *PLoS computational biology*, 7(11):e1002211, 2011.

Natalia Caporale and Yang Dan. Spike timing–dependent plasticity: a hebbian learning rule. *Annu. Rev. Neurosci.*, 31:25–46, 2008.

Vijayalakshmi Chelliah, Camille Laibe, and Nicolas Le Novère. Biomodels database: a repository of mathematical models of biological processes. In *In Silico Systems Biology*, pages 189–199. Springer, 2013.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Jane Cronin and Jane Cronin Boutelle. *Mathematical aspects of Hodgkin-Huxley neural theory*. Number 7. Cambridge University Press, 1987.

Manuel de Lera Ruiz and Richard L Kraus. Voltage-gated sodium channels: structure, function, pharmacology, and clinical indications. *Journal of medicinal chemistry*, 58(18):7093–7118, 2015.

Marzia Di Filippo, Chiara Damiani, and Dario Pescini. Gpruler: Metabolic gene-protein-reaction rules automatic reconstruction. *PLoS computational biology*, 17(11):e1009550, 2021.

Martin Fink and Denis Noble. Markov models for ion channels: versatility versus identifiability and speed. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1896):2161–2179, 2009.

Paul W Glimcher. Understanding dopamine and reinforcement learning: the dopamine reward prediction error hypothesis. *Proceedings of the National Academy of Sciences*, 108(Supplement 3): 15647–15654, 2011.

Nicholas Hernjak, Boris M Slepchenko, Kathleen Fernald, Charles C Fink, Dale Fortin, Ion I Moraru, James Watras, and Leslie M Loew. Modeling and analysis of calcium signaling events leading to long-term depression in cerebellar purkinje cells. *Biophysical journal*, 89(6):3790–3806, 2005.

David Hochberg and Josep M Ribó. Stoichiometric network analysis of entropy production in chemical reactions. *Physical Chemistry Chemical Physics*, 20(36):23726–23739, 2018.

Jianxin Huang and Zhehuang Huang. A parameters optimization of synergetic neural network based on differential evolution algorithm. *International Journal of Hybrid Information Technology*, 9(4): 359–366, 2016.

Eugene M Izhikevich. Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14 (6):1569–1572, 2003.

Philip S Low, Jeffrey L Bada, and George N Somero. Temperature adaptation of enzymes: roles of the free energy, the enthalpy, and the entropy of activation. *Proceedings of the National Academy of Sciences*, 70(2):430–432, 1973.

Jorge Nocedal and Stephen J Wright. Quadratic programming. *Numerical optimization*, pages 448–492, 2006.

Michael Pfeiffer and Thomas Pfeil. Deep learning with spiking neurons: opportunities and challenges. *Frontiers in neuroscience*, page 774, 2018.

Kenneth O Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.

Amirhossein Tavanaei and Anthony Maida. Bp-stdp: Approximating backpropagation using spike timing dependent plasticity. *Neurocomputing*, 330:39–47, 2019.

Jordan A Taylor and Richard B Ivry. Cerebellar and prefrontal cortex contributions to adaptation, strategies, and reinforcement learning. *Progress in brain research*, 210:217–253, 2014.

Gaetano Vilasi. *Hamiltonian dynamics*. World Scientific, 2001.

Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Yisen Wang, and Zhouchen Lin. Training feedback spiking neural networks by implicit differentiation on the equilibrium state. *Advances in Neural Information Processing Systems*, 34, 2021.

Appendix A. Proof : Flow satisfies the condition

One of the core ideas is whether these flows satisfy the condition. First, the system flows in the direction of decreasing energy Second, about conservation of Mass, where x moves in convex satisfying $\boldsymbol{M}^T x = C$ Finally, this flow is a proof of Positive Of Mass that x is not negative.

A.1. Chemical Flow

Flow

$$\begin{aligned}\frac{d\boldsymbol{x}}{dt} &= -\boldsymbol{M}^{\perp}\left[\boldsymbol{k}^T \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} r(\boldsymbol{M}^{\perp})} - e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})})\right]^T \\ &= -\boldsymbol{M}^{\perp}\left[\boldsymbol{k}^T \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} \boldsymbol{M}^{\perp}} - 1) \circ e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})}\right]^T\end{aligned}$$

Energy

$$\begin{aligned}\frac{dH}{dt} &= \frac{\partial H}{\partial \boldsymbol{x}}\frac{d\boldsymbol{x}}{dt} + \frac{\partial H}{\partial \boldsymbol{p}}\frac{d\boldsymbol{p}}{dt} \\ &= -\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp}\left[\boldsymbol{k}^T \circ \frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp} \circ (\frac{e^{\frac{\partial H}{\partial \boldsymbol{x}} \boldsymbol{M}^{\perp}} - 1}{\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp}}) \circ e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})}\right]^T \\ &= -(\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp}) diag\left[\boldsymbol{k} \circ (\frac{e^{\frac{\partial H}{\partial \boldsymbol{x}} \boldsymbol{M}^{\perp}} - 1}{\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp}})^T \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})})^T\right](\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp})^T < 0\end{aligned}$$

Conservation

$$\begin{aligned}\frac{d}{dt}(\boldsymbol{M}^T\boldsymbol{x}) = \boldsymbol{M}^T\frac{d\boldsymbol{x}}{dt} &= -\boldsymbol{M}^T\boldsymbol{M}^{\perp}\left[\boldsymbol{k}^T \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} \boldsymbol{M}^{\perp}} - 1) \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})})\right]^T \\ &= 0_{c\times e}\left[\boldsymbol{k}^T \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} \boldsymbol{M}^{\perp}} - 1) \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})})\right]^T = 0\end{aligned}$$

Positive

$$\begin{aligned}\frac{d\boldsymbol{x}}{dt} &= -\boldsymbol{M}^{\perp}\left[\boldsymbol{k}^T \circ (e^{\frac{\partial H}{\partial \boldsymbol{x}} \boldsymbol{M}^{\perp}} - 1) \circ e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})}\right]^T \\ &= \boldsymbol{M}_{+}(r(-\boldsymbol{M}^{\perp}) - r(\boldsymbol{M}^{\perp}))(e^{\frac{\partial H}{\partial \boldsymbol{x}} r(\boldsymbol{M}^{\perp})} - e^{\frac{\partial H}{\partial \boldsymbol{x}} r(-\boldsymbol{M}^{\perp})})^T \\ &\geq -\boldsymbol{M}_{+}r(\boldsymbol{M}^{\perp})\left[e^{(\frac{\partial H}{\partial \boldsymbol{x}} r(\boldsymbol{M}^{\perp}))^T} \circ \boldsymbol{k}\right] - \boldsymbol{M}_{+}r(-\boldsymbol{M}^{\perp})\left[e^{(\frac{\partial H}{\partial \boldsymbol{x}} R(-\boldsymbol{M}^{\perp}))^T} \circ \boldsymbol{k}\right] \\ &- r(\boldsymbol{M}^{\perp})\left[e^{(\frac{\partial H}{\partial \boldsymbol{x}} r(\boldsymbol{M}^{\perp}))^T} \circ \boldsymbol{k}\right] = -R(\boldsymbol{M}^{\perp})\left[e^{((\ln(\boldsymbol{x})^T + h(\boldsymbol{x},p))r(\boldsymbol{M}^{\perp}))^T} \circ \boldsymbol{k}\right] \\ &= -r(\boldsymbol{M}^{\perp}) diag(e^{(\ln(\boldsymbol{x})^T R(\boldsymbol{M}^{\perp}))^T})\left[e^{(h(\boldsymbol{x},p)R(\boldsymbol{M}^{\perp}))^T} \circ \boldsymbol{k}\right] \\ &\left[-r(\boldsymbol{M}^{\perp}) diag(e^{(\ln(\boldsymbol{x})^T r(\boldsymbol{M}^{\perp}))^T})\right]_i = -r(\boldsymbol{M}^{\perp})_i \prod \boldsymbol{x}_i^{r(\boldsymbol{M}^{\perp})_i^T} \\ &- \sum r(\boldsymbol{M}^{\perp})_{ij} \prod \boldsymbol{x}_i^{r(\boldsymbol{M}^{\perp})_{ij}} = \sum \begin{cases} 0 & if r(\boldsymbol{M}^{\perp})_{ij} = 0 \\ f(\boldsymbol{x}, p) \circ \boldsymbol{x}^{R(\boldsymbol{M}^{\perp})_{ij}} & if r(\boldsymbol{M}^{\perp})_{ij} \neq 0 \end{cases}\end{aligned}$$

A.2. Hamiltonian Flow

Flow

$$\begin{aligned}\frac{d\boldsymbol{x}}{dt} &= \boldsymbol{M}^{\perp}\left[\boldsymbol{v}^T \circ (\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp}) \circ (e^{\ln(\boldsymbol{x})^T |\boldsymbol{M}^{\perp}|})\right]^T \\ \frac{d\boldsymbol{p}}{dt} &= \boldsymbol{M}^{\perp}\left[\boldsymbol{v}^T \circ (\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp}) \circ (e^{\ln(\boldsymbol{x})^T |\boldsymbol{M}^{\perp}|})\right]^T\end{aligned}$$

Energy

$$\begin{aligned}\frac{dH}{dt} &= \frac{\partial H}{\partial \boldsymbol{x}}\frac{d\boldsymbol{x}}{dt} + \frac{\partial H}{\partial \boldsymbol{p}}\frac{d\boldsymbol{p}}{dt} \\ &= \frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp}\left[\boldsymbol{v}^T \circ (\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp}) \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})\right]^T - \frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp}\left[\boldsymbol{v}^T \circ (\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp}) \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})\right]^T \\ &= (\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp}\left[(\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp})^T\right] - \frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp}\left[(\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp})^T\right]) \circ \boldsymbol{v} \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T = 0 \\ &\because let \quad f = \left[\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp}\right](\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp})^T \in \mathbb{R}^{1\times 1} \\ f = f^T &= (\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp})\left[(\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp})^T\right] = \left[\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp}\right](\frac{\partial H}{\partial \boldsymbol{x}}\boldsymbol{M}^{\perp})^T\end{aligned}$$

Conservation

$$\begin{aligned}\frac{d}{dt}(\boldsymbol{M}^T\boldsymbol{x}) = \boldsymbol{M}^T\frac{d\boldsymbol{x}}{dt} &= \boldsymbol{M}^T\boldsymbol{M}^{\perp}\left[(\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp})^T \circ \boldsymbol{v} \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T\right] \\ &= 0_{c\times e}\left[(\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp})^T \circ \boldsymbol{v} \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T\right] = 0\end{aligned}$$

Positive

$$\begin{aligned}\frac{d\boldsymbol{x}}{dt} &= \boldsymbol{M}^{\perp}\left[(\frac{\partial H}{\partial \boldsymbol{p}}\boldsymbol{M}^{\perp})^T \circ \boldsymbol{v} \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T\right] \\ &= \boldsymbol{M}^{\perp}(\boldsymbol{M}^{\perp})^T\left[\frac{1}{m} \circ p \circ \boldsymbol{v} \circ (e^{\ln(\boldsymbol{x})^T|\boldsymbol{M}^{\perp}|})^T\right] \circ \boldsymbol{x}\end{aligned}$$

A.3. Collision Flow

Flow

$$\frac{d\boldsymbol{p}}{dt} = -c \circ (\frac{\partial H}{\partial \boldsymbol{p}})^T$$

Energy

$$\frac{dH}{dt} = \frac{\partial H}{\partial \boldsymbol{x}}\frac{d\boldsymbol{x}}{dt} + \frac{\partial H}{\partial \boldsymbol{p}}\frac{d\boldsymbol{p}}{dt} = -\frac{\partial H}{\partial \boldsymbol{p}}\left[(\frac{\partial H}{\partial \boldsymbol{p}})^T \circ c\right] = -\left[\frac{\partial H}{\partial \boldsymbol{p}} \circ \frac{\partial H}{\partial \boldsymbol{p}}\right]c < 0$$

A.4. External Homeostasis Flow

Flow

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{h} \circ \boldsymbol{x} \circ (e^{-\frac{\partial H}{\partial \boldsymbol{x}}} - 1)^T$$

Energy

$$\begin{aligned}\frac{dH}{dt} = \frac{\partial H}{\partial \boldsymbol{x}}\frac{d\boldsymbol{x}}{dt} + \frac{\partial H}{\partial \boldsymbol{p}}\frac{d\boldsymbol{p}}{dt} &= \frac{\partial H}{\partial \boldsymbol{x}}[\boldsymbol{h} \circ \boldsymbol{x} \circ (e^{-\frac{\partial H}{\partial \boldsymbol{x}}} - 1)^T] \\ &= -(\frac{\partial H}{\partial \boldsymbol{x}})diag[\boldsymbol{h} \circ \boldsymbol{x} \circ (\frac{(-e^{-\frac{\partial H}{\partial \boldsymbol{x}}} + 1)}{\frac{\partial H}{\partial \boldsymbol{x}}})^T](\frac{\partial H}{\partial \boldsymbol{x}})^T < 0\end{aligned}$$

Positive

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{h} \circ \boldsymbol{x} \circ (e^{-\frac{\partial H}{\partial \boldsymbol{x}}} - 1)^T = [\boldsymbol{h} \circ (e^{-\frac{\partial H}{\partial \boldsymbol{x}}} - 1)^T] \circ \boldsymbol{x}$$

Appendix B. Explanation of Bio Model

To represent the HH model with an implicit layer, the chemical reactions, diffusion, current, voltage-gated channels, and membrane potential must each be explainable as flows in Eq. (6), This can be explained by modeling the three spaces of neurons (Extracellar Space, Cell Membrane, Interacellar Space). Fig. 6 shows one of the four channels (Voltage-gated Na+ channel, Voltage-gated K+ channel, K+ Leak channel, and Na+/K+ pump) of the HH model to dynamic system. In this appendix, we discuss how we transform the foundational Fick's Law for diffusion and the Nernst equation, which underlies the membrane potential, into our designed implicit layer, excluding the chemical reactions, voltage-gated channels, and current described in the main paper.

Fick's second Law Fick's diffusion law $= D\nabla^2 n(x)$ This can be described by the diffusion flow of $\boldsymbol{g}_1$, an element of successive spaces $\boldsymbol{s}_{[0:n+1]}$. For each dt, each emits an amount proportional to its mass to both sides of the space Their formula can be described as a quadratic differential $\frac{d\boldsymbol{x}_i}{dt} = \alpha(\boldsymbol{x}_{i-1} + \boldsymbol{x}_{i+1} - 2\boldsymbol{x}_i)$. If n go to this limit, it can explain the Fick's second law formula.

Nernst equation Nernst equation determining the membrane potential $E = E_0 + \frac{RT}{nF} \ln \frac{x_{in}}{x_{out}}$, It can be derived from Nernst-Planck equation $\frac{\partial c}{\partial t} + \nabla \cdot \boldsymbol{J} = 0$. At this point, the flow of molecules, $\boldsymbol{J}$, is $-D + \frac{Dze}{k_B T} c\boldsymbol{E}$ Can be described as diffusion flow + hamiltonian flow of $\boldsymbol{g}_1$. The sum of the diffusion and hamiltonian flows must be zero to meet each concentration equilibrium point.

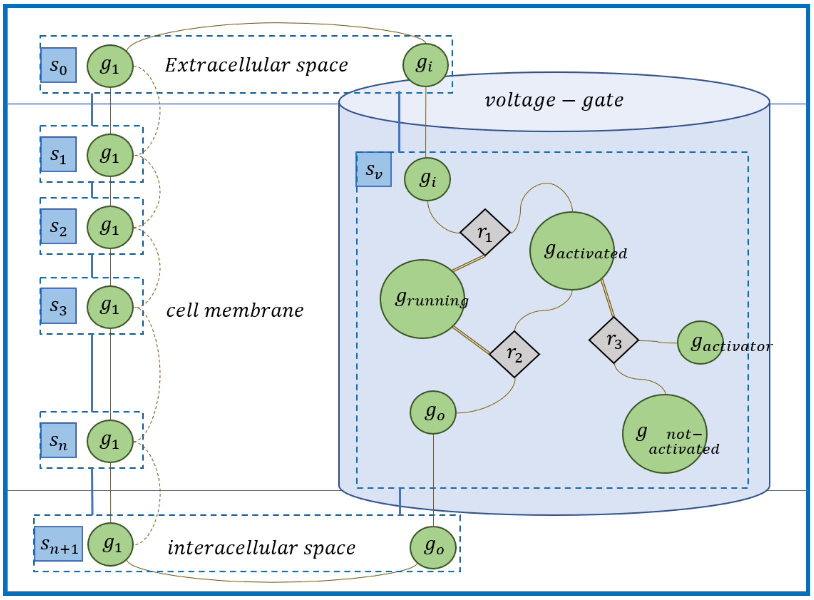


Figure 6: Explanation Hodgkin Huxley model by dynamical system