

# 1 Neural ODE

## 1.1 Adjoint Method

Let  $M$  a manifold, and  $x(t) \in C^1(\mathbb{R}, M)$  a trajectory, obeying

$$\frac{dx}{dt}(t) = f(t, x(t); \theta),$$

and

$$x(t_0) = x_0,$$

where  $f \in C(\mathbb{R} \times M, T_M)$  parameterized by  $\theta$ . For  $\forall t_1 > t_0$ , let

$$x_1 := x_0 + \int_{t_0}^{t_1} f(t, x(t); \theta) dt.$$

Then

**Theorem 1.** Let  $\mathcal{C} \in C^1(M, \mathbb{R})$ , and  $\forall x(t) \in C^1(\mathbb{R}, M)$  obeying dynamics  $f(t, x; \theta) \in C^1(\mathbb{R} \times M, T_M)$  with initial value  $x(t_0) = x_0$ . Denote

$$L := \mathcal{C}\left(x_0 + \int_{t_0}^{t_1} f(\tau, x(\tau); \theta) d\tau\right).$$

Then we have, for  $\forall t \in [t_0, t_1]$  given,

$$\frac{\partial L}{\partial x^\alpha(t)} = \frac{\partial L}{\partial x_1^\alpha} - \int_t^{t_1} \frac{\partial L}{\partial x^\beta(\tau)} \frac{\partial f^\beta}{\partial x^\alpha}(\tau, x(\tau); \theta) d\tau,$$

and

$$\frac{\partial L}{\partial \theta} = - \int_{t_0}^{t_1} \frac{\partial L}{\partial x^\beta(\tau)} \frac{\partial f^\beta}{\partial \theta}(\tau, x(\tau); \theta) d\tau.$$

**Proof.** Suppose the  $x(t)$  is layerized, the  $L$  depends on the variables (inputs and model parameters) on the  $i$ th layer can be regarded as the loss of a new model by truncating the original at the  $i$ th layer, which we call  $L_i(z_i)$ .

$$\begin{aligned} \frac{\partial L_i}{\partial x_i^\alpha}(x_i) &= \frac{\partial L_{i+1}}{\partial x_{i+1}^\beta}(x_{i+1}) \frac{\partial x_{i+1}^\beta}{\partial x_i^\alpha}(x_i) \\ &= \frac{\partial L_{i+1}}{\partial x_1^\beta}(x_{i+1}) \frac{\partial}{\partial x_i^\alpha}(x_i^\beta + f^\beta(t_i, x_i; \theta) \Delta t) \\ &= \frac{\partial L_{i+1}}{\partial x_{i+1}^\alpha}(x_{i+1}) + \frac{\partial L_{i+1}}{\partial x_{i+1}^\beta}(x_{i+1}) \partial_\alpha f^\beta(t_i, x_i; \theta) \Delta t. \end{aligned}$$

This hints that

$$\frac{d}{dt} \frac{\partial L}{\partial x^\alpha(t)} = - \frac{\partial L}{\partial x^\beta(t)} \frac{\partial f^\beta}{\partial x^\alpha}(t, x(t); \theta).$$

The initial value is  $\partial L / \partial x_1$ . Thus

$$\frac{\partial L}{\partial x(t)} = \frac{\partial L}{\partial x_1} - \int_t^{t_1} \frac{\partial L}{\partial x^\beta(\tau)} \frac{\partial f^\beta}{\partial x^\alpha}(\tau, x(\tau); \theta) d\tau.$$

Varying  $\theta$  will vary the  $L_i(x_i)$  from two aspects, the effect from  $\partial L_{i+1}/\partial\theta$  and the  $\Delta x_{i+1}$  caused by  $\Delta\theta$ .

$$\begin{aligned}\frac{\partial L_i}{\partial\theta}(x_i) &= \frac{\partial L_{i+1}}{\partial\theta}(x_{i+1}) + \frac{\partial L_{i+1}}{\partial x_{i+1}} \frac{\partial x_{i+1}}{\partial\theta} \\ &= \frac{\partial L_{i+1}}{\partial\theta}(x_{i+1}) + \frac{\partial L_{i+1}}{\partial x_{i+1}} \frac{\partial}{\partial\theta}(x_i^\beta + f^\beta(t_i, x_i; \theta)\Delta t) \\ &= \frac{\partial L_{i+1}}{\partial\theta}(x_{i+1}) + \frac{\partial L_{i+1}}{\partial x_{i+1}} \frac{\partial f^\beta}{\partial\theta}(t_i, x_i; \theta)\Delta t.\end{aligned}$$

This hints that

$$\frac{d}{dt} \frac{\partial L}{\partial\theta} = - \frac{\partial L}{\partial x^\alpha(t)} \frac{\partial f^\beta}{\partial\theta}(t, x(t), \theta).$$

The initial value is 0 since  $\mathcal{C}(\cdot)$  is explicitly independent on  $\theta$ . Thus

$$\frac{\partial L}{\partial\theta} = - \int_{t_0}^t \frac{\partial L}{\partial x^\beta(\tau)} \frac{\partial f^\beta}{\partial\theta}(\tau, x(\tau); \theta) d\tau. \quad \square$$

## 2 Hopfield Network

### 2.1 Discrete-time Hopfield Network

**Definition 2.** [Discrete-time Hopfield Network]

Let  $t \in \mathbb{N}$  and  $x \in \{-1, +1\}^d$ ,  $W \in \mathbb{R}^d \times \mathbb{R}^d$  with  $W_{\alpha\beta} = W_{\beta\alpha}$  and  $W_{\alpha\alpha} = 0$ , and  $b \in \mathbb{R}^d$ . Define discrete-time dynamics

$$x^\alpha(t+1) = \text{sign}(W_{\alpha\beta} x^\beta(t) + b^\alpha).$$

The  $(x, W, b)$  is called a discrete-time Hopfield network.

**Lemma 3.** Let  $(x, W, b)$  a discrete-time Hopfield network. Define  $\mathcal{E}(x) := -(1/2)W_{\alpha\beta} x^\alpha x^\beta - b_\alpha x^\alpha$ . Then  $\mathcal{E}(x(t+1)) - \mathcal{E}(x(t)) \leq 0$ .

**Proof.** Consider async-updation of Hopfield network, that is, change the component at dimension  $\hat{\alpha}$ , i.e.  $x'_{\hat{\alpha}} = \text{sign}[W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}}]$ , then

$$\begin{aligned}\mathcal{E}(x') - \mathcal{E}(x) &= -\frac{1}{2}W_{\alpha\beta} x'^\alpha x'^\beta - b_\alpha x'^\alpha + \frac{1}{2}W_{\alpha\beta} x^\alpha x^\beta + b_\alpha x^\alpha \\ &= -2(x'^{\hat{\alpha}} - x^{\hat{\alpha}})(W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}}),\end{aligned}$$

which employs conditions  $W_{\alpha\beta} = W_{\beta\alpha}$  and  $W_{\alpha\alpha} = 0$ . Next, we prove that, combining with  $x'_{\hat{\alpha}} = \text{sign}[W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}}]$ , this implies  $\mathcal{E}(x') - \mathcal{E}(x) \leq 0$ .

If  $(x'^{\hat{\alpha}} - x^{\hat{\alpha}}) > 0$ , then  $x'^{\hat{\alpha}} = 1$  and  $x^{\hat{\alpha}} = -1$ . Since  $x'_{\hat{\alpha}} = \text{sign}[W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}}]$ ,  $W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}} > 0$ . Then  $\mathcal{E}(x') - \mathcal{E}(x) < 0$ . Contrarily, if  $(x'^{\hat{\alpha}} - x^{\hat{\alpha}}) < 0$ , then  $x'^{\hat{\alpha}} = -1$  and  $x^{\hat{\alpha}} = 1$ , implying  $W_{\hat{\alpha}\beta} x^\beta + b_{\hat{\alpha}} < 0$ . Also  $\mathcal{E}(x') - \mathcal{E}(x) < 0$ . Otherwise,  $\mathcal{E}(x') - \mathcal{E}(x) = 0$ . So, we conclude  $\mathcal{E}(x') - \mathcal{E}(x) \leq 0$ .  $\square$

**Theorem 4.** *[Convergence of Discrete-time Hopfield Network] Let  $(x, W, b)$  a discrete-time Hopfield network. Then any trajectory obeying the update rule will converge either to a fixed point or a limit circle.*

**Proof.** Since the states of the network are finite, the  $\mathcal{E}$  is lower bounded. □

## 2.2 Continuous-time Hopfield Network

**Definition 5.** *[Continuous-time Hopfield Network]*

Let  $t \in [0, +\infty)$  and  $x \in [-1, +1]^d$ ,  $W \in \mathbb{R}^d \times \mathbb{R}^d$  with  $W_{\alpha\beta} = W_{\beta\alpha}$ , and  $b \in \mathbb{R}^d$ . Define dynamics

$$\tau \frac{dx^\alpha}{dt}(t) = -x^\alpha(t) + f(W^\alpha_\beta x^\beta(t) + b^\alpha),$$

where  $\tau \in (0, +\infty)$  a constant and  $f: \mathbb{R} \rightarrow [-1, 1]$  being increasing. The  $(x, W, b; \tau, f)$  is called a continuous-time Hopfield network.

**Remark 6.** With

$$\tau \frac{x^\alpha(t + \Delta t) - x^\alpha(t)}{\Delta t} = -x^\alpha(t) + f(W^\alpha_\beta x^\beta(t) + b^\alpha).$$

Setting  $\Delta t = \tau$  gives and  $f(\cdot) = \text{sign}(\cdot)$  gives

$$x^\alpha(t + \tau) = \text{sign}(W^\alpha_\beta x^\beta(t) + b^\alpha),$$

which is the same as the discrete-time Hopfield network.

**Lemma 7.** Let  $(x, W, b; \tau, f)$  a continuous-time Hopfield network. Define  $a^\alpha := W^\alpha_\beta x^\beta + b^\alpha$  and  $y^\alpha := f(a^\alpha)$ , then

$$\mathcal{E}(y) := -\frac{1}{2} W_{\alpha\beta} y^\alpha y^\beta - b_\alpha y^\alpha + \sum_\alpha \int^{y^\alpha} f^{-1}(y^\alpha) dy^\alpha.$$

Then  $\mathcal{E}(y(x(t + dt))) - \mathcal{E}(y(x(t))) \leq 0$ .

**Proof.** The dynamics of  $a^\alpha$  is

$$\begin{aligned} \tau \frac{da^\alpha}{dt} &= \tau W^\alpha_\beta \frac{dx^\beta}{dt} \\ &= W^\alpha_\beta [-x^\beta(t) + f(a^\beta)] \\ &= -(W^\alpha_\beta x^\beta(t) + b^\alpha) + b^\alpha + W^\alpha_\beta y^\beta \\ &= W^\alpha_\beta y^\beta + b^\alpha - a^\alpha. \end{aligned}$$

Since  $W$  is symmetric, we have  $\partial \mathcal{E} / \partial y^\alpha = -W_{\alpha\beta} y^\beta - b_\alpha + f^{-1}(y_\alpha)$ . Then

$$\begin{aligned} \frac{d\mathcal{E}}{dt} &= \frac{dy^\alpha}{dt} (-W_{\alpha\beta} y^\beta - b_\alpha + f^{-1}(y_\alpha)) \\ &= \frac{dy^\alpha}{dt} (-W_{\alpha\beta} y^\beta - b_\alpha + a_\alpha) \\ &= -\frac{dy^\alpha}{dt} (W_{\alpha\beta} y^\beta + b_\alpha - a_\alpha) \end{aligned}$$

Notice that, the second term of rhs is exactly the dynamics of  $a_\alpha$ , then

$$\begin{aligned}\frac{d\mathcal{E}}{dt} &= -\tau \frac{dy^\alpha}{dt} \frac{da_\alpha}{dt} \\ &= -\tau \frac{dy^\alpha}{da^\alpha} \left( \frac{da^\alpha}{dt} \frac{da_\alpha}{dt} \right) \\ &= -\tau f'(a^\alpha) \left( \frac{da^\alpha}{dt} \frac{da_\alpha}{dt} \right).\end{aligned}$$

Since  $f$  is increasing and  $\tau > 0$ ,  $d\mathcal{E}/dt \leq 0$ .  $\square$

**Remark 8.** The condition  $W_{\alpha\alpha} = 0$  for  $\forall \alpha$  is not essential for this lemma. Indeed, this condition is absent in the proof. This differs from the case of discrete-time.

**Theorem 9.** *[Convergence of Continuous-time Hopfield Network] Let  $(x, W, b; \tau, f)$  a continuous-time Hopfield network. Then any trajectory along the dynamics will converge either to a fixed point or a limit circle.*

**Proof.** The function  $E := \mathcal{E} \circ y$  is lower bounded since  $y$ , i.e. function  $f: \mathbb{R} \rightarrow [-1, 1]$ , is bounded. This  $E$  is a Lyapunov function for the continuous-time Hopfield network.  $\square$

**Corollary 10.** *Let  $(x, W, b; \tau, f)$  a continuous-time Hopfield network. And  $D := \{x_n | x_n \in \mathbb{R}^d, n=1, \dots, N\}$  a dataset<sup>1</sup>. If add constraint  $W_{\alpha\alpha} = 0$  for  $\forall \alpha$ , then we can train the Hopfield network by seeking a proper parameters  $(W, b)$ , s.t. its stable points cover the dataset as much as possible, by<sup>2</sup>*

**Algorithm 1**

```
W, b = init_W, init_b # e.g. by Glorot initializer
for step in range(max_step):
    for x in dataset:
        y = f(W @ x + b)
        loss = norm(x - y)
        optimizer.minimize(objective=loss, variables=(W, b))
    W = set_zero_diag(symmetrize(W))
```

**Proof.** For  $\forall x_n \in D$ , we try to find  $(W, b)$ , s.t.  $dx/dt = 0$  at  $x_n$ , i.e.

$$x_n^\alpha = f(W_{\alpha\beta}^\alpha x_n^\beta + b^\alpha).$$

When  $W_{\alpha\alpha} = 0$  for  $\forall \alpha$ ,  $f(W_{\alpha\beta}^\alpha x_n^\beta + b^\alpha)$  thus has no information of  $x^\alpha$ , it has to predict the  $x^\alpha$  by the interaction between  $x^\alpha$  and the other  $x$ 's components.  $\square$

**Remark 11.** This algorithm is equivalent to

**Algorithm 2**

```
dt = ... # e.g. 0.1
W, b = init_W, init_b
for step in range(max_step):
    for x in dataset:
        # that is, compute x(dt), with x(0) = x
```

1. We use Greek alphabet for component in  $\mathbb{R}^d$  and Latin alphabet for element in dataset.

2. This algorithm generalizes the algorithm 42.9 of Mackay.

```

y = ode_solve(f=lambda t, x: -x + f(W @ x + b), t0=0, t1=dt, x0=x)
loss = norm(x - y)
optimizer.minimize(objective=loss, variables=(W, b))
W = set_zero_diag(symmetrize(W))

```

Indeed, trying to reach  $y = x$  within a small interval will force  $x$  to be a fixed point.

## 3 Variations

### 3.1 Variation 1

**Theorem 12.** Let  $v \in \mathbb{R}^d$ ,  $F \in C^1(\mathbb{R}^n, \mathbb{R})$ ,  $W \in \mathbb{R}^n \times \mathbb{R}^d$ ,  $b \in \mathbb{R}^n$ , and  $\tau > 0$ . Define the dynamics

$$\tau \frac{dx}{dt} = -\nabla E(x) = -x + W^T \cdot \nabla F(W \cdot x + b) + v.$$

If  $\nabla F(\cdot)$  is bounded, i.e.  $\exists K > 0$  s.t.  $\max_{x \in \mathbb{R}^n} \{\|\nabla F(x)\|\} < K$ , then any trajectory along the dynamics will converge either to a fixed point or a limit circle.

**Proof.** Let  $E(x) := \frac{1}{2}x_\alpha x^\alpha - v_\alpha x^\alpha - F(W^\alpha_\beta x^\beta + b^\alpha)$ , then  $\tau dx/dt = -\nabla E(x)$ . The  $-x$  term will dominate the  $W^T \cdot \nabla F(W \cdot x + b)$  term for  $\|x\| > K\|W\|$ , thus converges. So  $E$  is a Lyapunov function of the dynamics.  $\square$

**Example 13.** Let  $F(x) := \sum_\alpha \int^{x^\alpha} \sigma(s) ds$ , where  $\sigma$  is sigmoid function. Then

$$\tau \frac{dx}{dt} = -x + W^T \cdot \sigma(W \cdot x + b) + v.$$

This coincides with the form in ref 1.

**Example 14.** Let  $F(x) := \beta^{-1} \ln(\beta \sum_\alpha e^{x^\alpha})$ ,  $b = 0$ , and  $v = 0$ , then

$$\tau \frac{dx}{dt} = -x + W^T \cdot \text{softmax}(\beta W \cdot x).$$

This coincides with the form in ref 2.

## 4 References

1. [On autoencoder scoring.](#)
2. [Hopfield networks is All You Need.](#)