1 Neural ODE

1.1 Adjoint Method

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

$$\frac{\mathrm{d}x}{\mathrm{d}t}(t)=f(t,x(t);\theta),$$

and

$$x(t_0)=x_0,$$

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

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

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*

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

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

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

and

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

Proof. Suppose the $x(t)$ is layerized, the L depends on the variables (inputs and model parameters) on the ith layer can be regarded as the loss of a new model by truncating the original at the ith 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

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

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

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

Varing θ will vary the $L_i(x_i)$ from two aspects, the effect from $\partial L_{i+1}/\partial\theta$ and the Δx_{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

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

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

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

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

$$x^\alpha(t+1) = \mathrm{sign}(W^\alpha{}_\beta x^\beta(t) + b^\alpha).$$

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 $\mathcal{E}(x(t+1)) - \mathcal{E}(x(t)) \leqslant 0$.*

Proof. Consider async-updation of Hopfield network, that is, change the component at dimension $\hat\alpha$, i.e. $x'_{\hat\alpha} = \mathrm{sign}[W_{\hat\alpha\beta}x^\beta + b_{\hat\alpha}]$, 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 $x'_{\hat\alpha} = \mathrm{sign}[W_{\hat\alpha\beta}x^\beta + b_{\hat\alpha}]$, this implies $\mathcal{E}(x') - \mathcal{E}(x) \leqslant 0$.

If $(x'^{\hat\alpha} - x^{\hat\alpha}) > 0$, then $x'^{\hat\alpha} = 1$ and $x^{\hat\alpha} = -1$. Since $x'_{\hat\alpha} = \mathrm{sign}[W_{\hat\alpha\beta}x^\beta + b_{\hat\alpha}]$, $W_{\hat\alpha\beta}x^\beta + b_{\hat\alpha} > 0$. 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 $\mathcal{E}(x') - \mathcal{E}(x) \leqslant 0$. $\square$

Theorem 4. *Let (x, W, b) a discrete-time Hopfield network. Then $\exists t_\star < +\infty$, s.t. $x(t+1) = x(t)$.*

Proof. Since the states of the network are finite, the $\mathcal{E}$ is lower bounded. Thus $\exists t_\star < +\infty$, s.t. $x(t+1) = x(t)$. [limit circle?] $\square$

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

$$\tau \frac{\mathrm{d}x^\alpha}{\mathrm{d}t}(t) = -x^\alpha(t) + f(W^\alpha{}_\beta x^\beta(t) + b^\alpha),$$

where $\tau \in (0, +\infty)$ a constant and $f: \mathbb{R} \to [-1, 1]$ 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 $f(.) = \mathrm{sign}(.)$ gives

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

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

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

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

Then $\mathcal{E}(y(x(t + \mathrm{d}t))) - \mathcal{E}(y(x(t))) \leqslant 0$.

Proof. The dynamics of a^α is

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

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

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

Notice that, the second term of rhs is exactly the dynamics of a_α, then

$$
\begin{aligned}
\frac{\mathrm{d}\mathcal{E}}{\mathrm{d}t} &= -\tau \frac{\mathrm{d}y^\alpha}{\mathrm{d}t}\frac{\mathrm{d}a_\alpha}{\mathrm{d}t} \\
&= -\tau \frac{\mathrm{d}y^\alpha}{\mathrm{d}a^\alpha}\left(\frac{\mathrm{d}a^\alpha}{\mathrm{d}t}\frac{\mathrm{d}a_\alpha}{\mathrm{d}t}\right) \\
&= -\tau f'(a^\alpha)\left(\frac{\mathrm{d}a^\alpha}{\mathrm{d}t}\frac{\mathrm{d}a_\alpha}{\mathrm{d}t}\right).
\end{aligned}
$$

Since f is increasing and $\tau > 0$, $\mathrm{d}\mathcal{E}/\mathrm{d}t \leqslant 0$. $\square$

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. *Let $(x, W, b; \tau, f)$ a continous-time Hopfield network. Then for $\forall \epsilon > 0$, $\exists t_\star < +\infty$, s.t. $\|\mathrm{d}x/\mathrm{d}t\| < \epsilon$. [limit circle, again?]*

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

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

***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. $\mathrm{d}x/\mathrm{d}t = 0$ at x_n, i.e.

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

When $W_{\alpha\alpha} = 0$ for $\forall \alpha$, $f(W^\alpha{}_\beta x^\beta + b^\alpha)$ thus has no information of x^α, it has to predict the x^α by the interaction between x^α 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
        y = ode_solve(f=lambda t, x: -x + f(W @ x + b), t0=0, t1=dt, x0=x)
```

1. We use Greek alphabet for component in $\mathbb{R}^d$ and Lattin alphabet for element in dataset.
2. This algorithm generalizes the algorithm 42.9 of Mackay.

```
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.