Neural Delay Differential Equations

Qunxi Zhu, Yao Guo, Wei Lin

School of Mathematical Science, Research Institute of Intelligent Complex Systems and ISTBI State Key Laboratory of Medical Neurobiology and MOE Frontiers Center for Brain Science
Fudan University
Shanghai 200433, China
{qxzhu16, yguo, wlin}@fudan.edu.cn

9th International Conference on Learning Representations

ICLR 20201

Why Neural Delay Differential Equations (NDDEs)

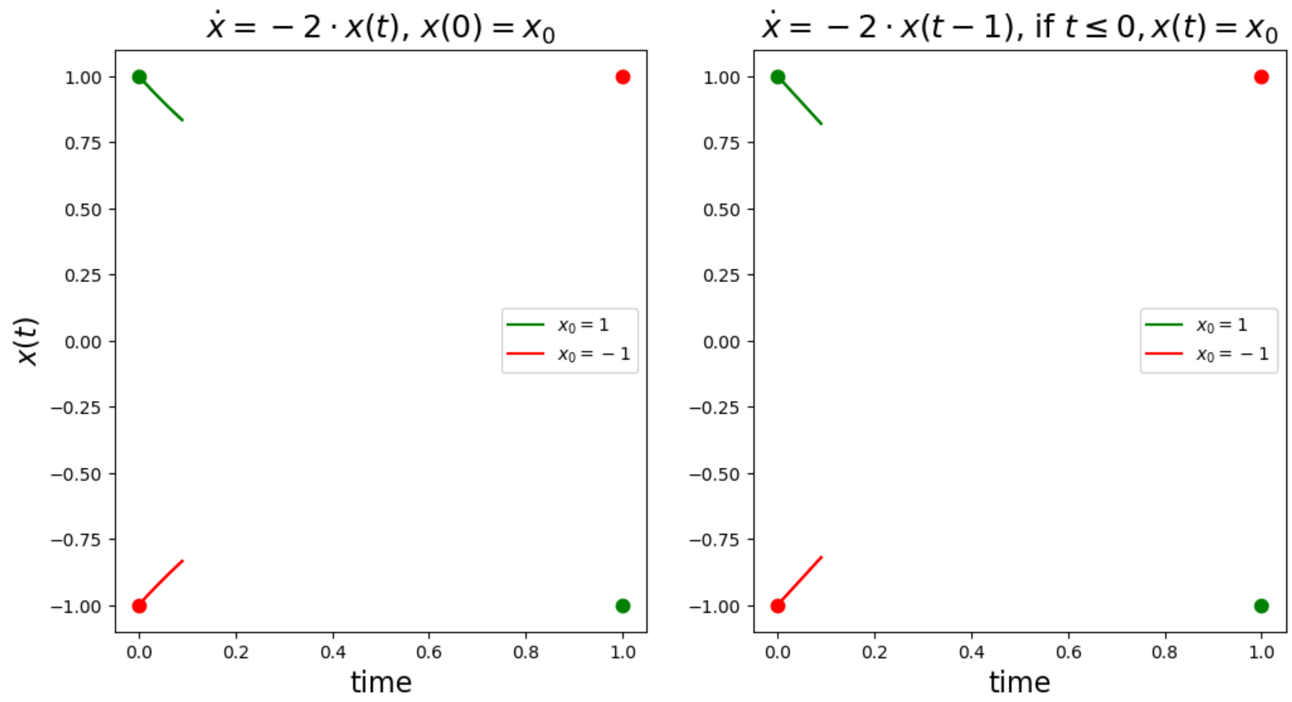
- Neural Ordinary Differential Equations (NODEs) are not universal, cannot represent some maps, such as the ***reflections*** or the ***concentric annuli***
- NODEs are not suitable to model the underlying system with the delay effect, such as **Mackey-Glass system**

Dupont et al., Augmented neural odes. **NeurIPS** 2019
Zhang et al., Approximation capabilities of neural odes and invertible residual networks. **ICML** 2020
Mackey, M. and Glass, L. Oscillation and chaos in physiological control systems. **Science**, 1977

Limitations of Neural ODEs

Dupont et al., Augmented Neural ODEs, NeurIPS, 2019: $g_{1d}(1) = -1,$ $g_{1d}(-1) = 1$ ***reflections***

Proposition 1. *The flow of an ODE cannot represent* $g_{1d}(x)$.



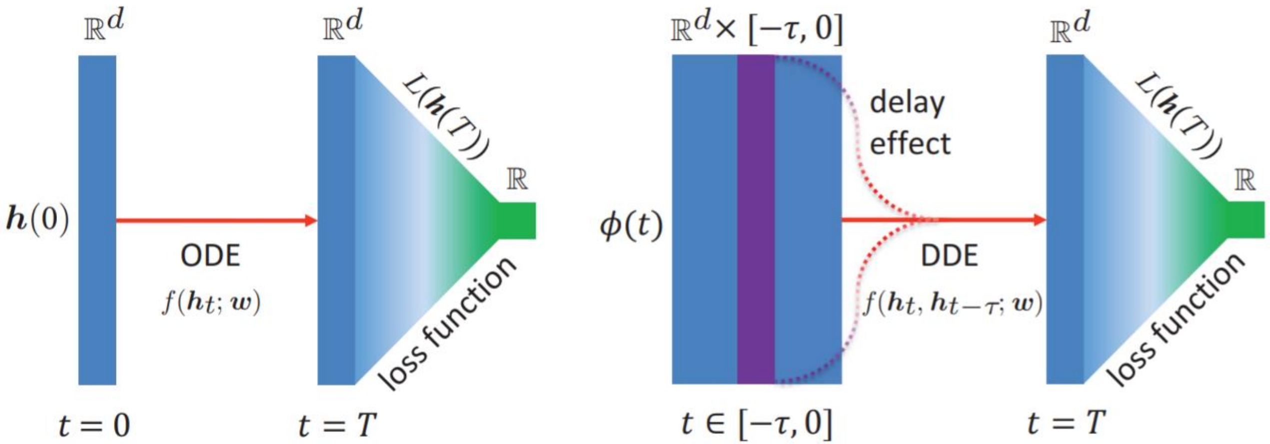
Neural Delay Differential Equations (NDDEs)

- Neural Ordinary Differential Equations (ODEs):

$$\frac{\mathrm{d}h(t)}{\mathrm{d}t} = f(h(t), w),\ h(0) = h_0$$

- Neural Delay Differential Equations (DDEs):

$$\frac{\mathrm{d}h(t)}{\mathrm{d}t} = f(h(t), h(t-\tau), w),\ if\ t \leq 0, h(t) = h_0$$



Universal approximation and adjoint dynamics of NDDEs

Theorem 2 *(Universal approximating capability of the NDDEs). For any given continuous function* $F : \mathbb{R}^n \to \mathbb{R}^n$*, if one can construct a neural network for approximating the map* $G(\boldsymbol{x}) = \frac{1}{T}[F(\boldsymbol{x}) - \boldsymbol{x}]$*, then there exists an NDDE of* n*-dimension that can model the map* $\boldsymbol{x} \mapsto F(\boldsymbol{x})$*, that is,* $h(T) = F(\boldsymbol{x})$ *with the initial function* $\phi(t) = \boldsymbol{x}$ *for* $t \leq 0$.

Adjoint: $\lambda(t) = \frac{\partial L(x(T))}{\partial x(t)}$

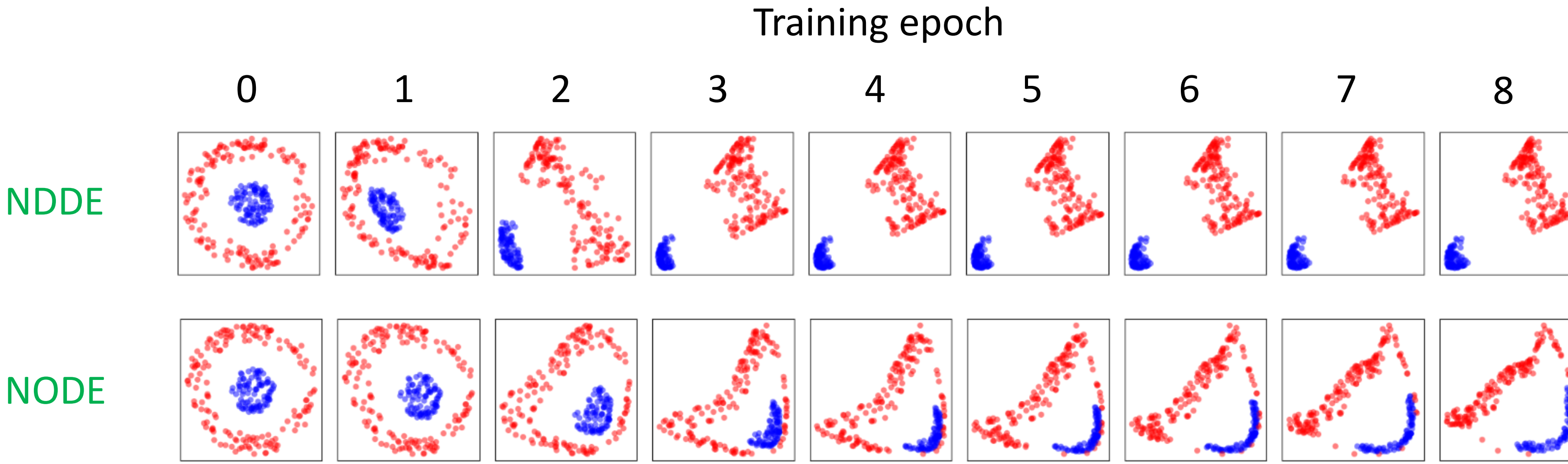
Theorem 1 *(Adjoint method for NDDEs). Consider the loss function* $L(\cdot)$*. Then, the dynamics of adjoint can be written as*

$$\begin{cases} \frac{d\boldsymbol{\lambda}(t)}{dt} = -\boldsymbol{\lambda}(t)^\top \frac{\partial f(\boldsymbol{h}_t, \boldsymbol{h}_{t-\tau}, t; \boldsymbol{w})}{\partial \boldsymbol{h}_t} - \boldsymbol{\lambda}(t+\tau)^\top \frac{\partial f(\boldsymbol{h}_{t+\tau}, \boldsymbol{h}_t, t; \boldsymbol{w})}{\partial \boldsymbol{h}_t} \chi_{[0,T-\tau]}(t),\ t <= T \\ \boldsymbol{\lambda}(T) = \frac{\partial L(\boldsymbol{h}(T))}{\partial \boldsymbol{h}(T)}, \end{cases} \tag{2}$$

where $\chi_{[0,T-\tau]}(\cdot)$ *is a typical characteristic function.*

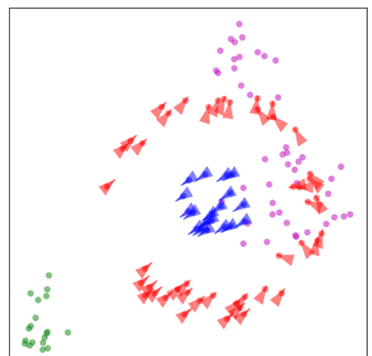
$$\frac{dL}{d\boldsymbol{w}} = \int_T^0 -\boldsymbol{\lambda}(t)^\top \frac{\partial f(\boldsymbol{h}_t, \boldsymbol{h}_{t-\tau}, t; \boldsymbol{w})}{\partial \boldsymbol{w}} dt.$$

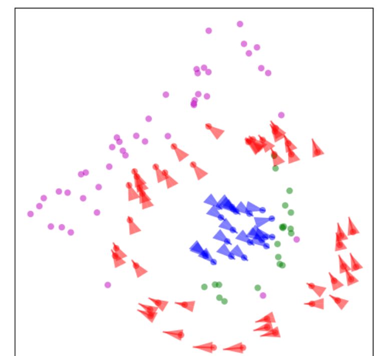
Example: ***concentric annuli***



$$\dot{x} = f(x(t-1), \theta)$$
$$t \in [0, 1]$$

$$if\ t \leq 0, x(t) = x_0$$



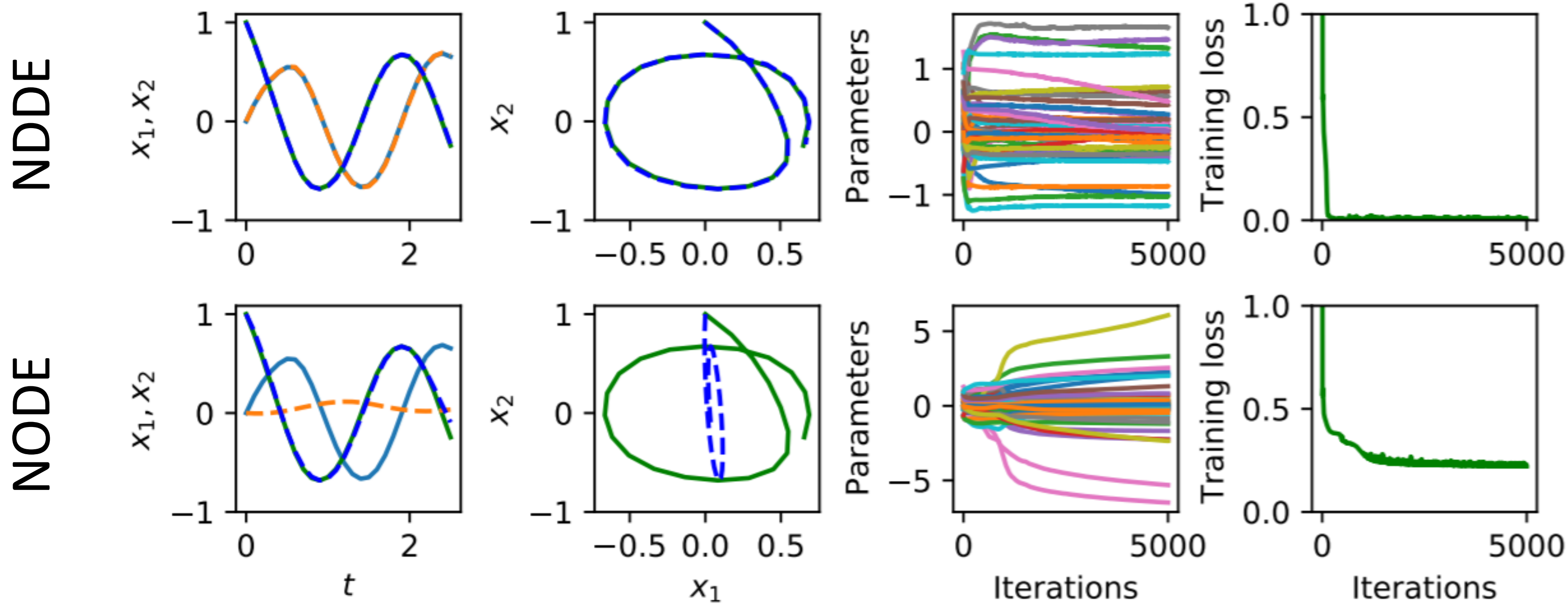


$$\dot{x} = f(x(t), \theta)$$
$$t \in [0, 1]$$

$$x(0) = x_0$$

Example: ***2-dimentional DDEs***

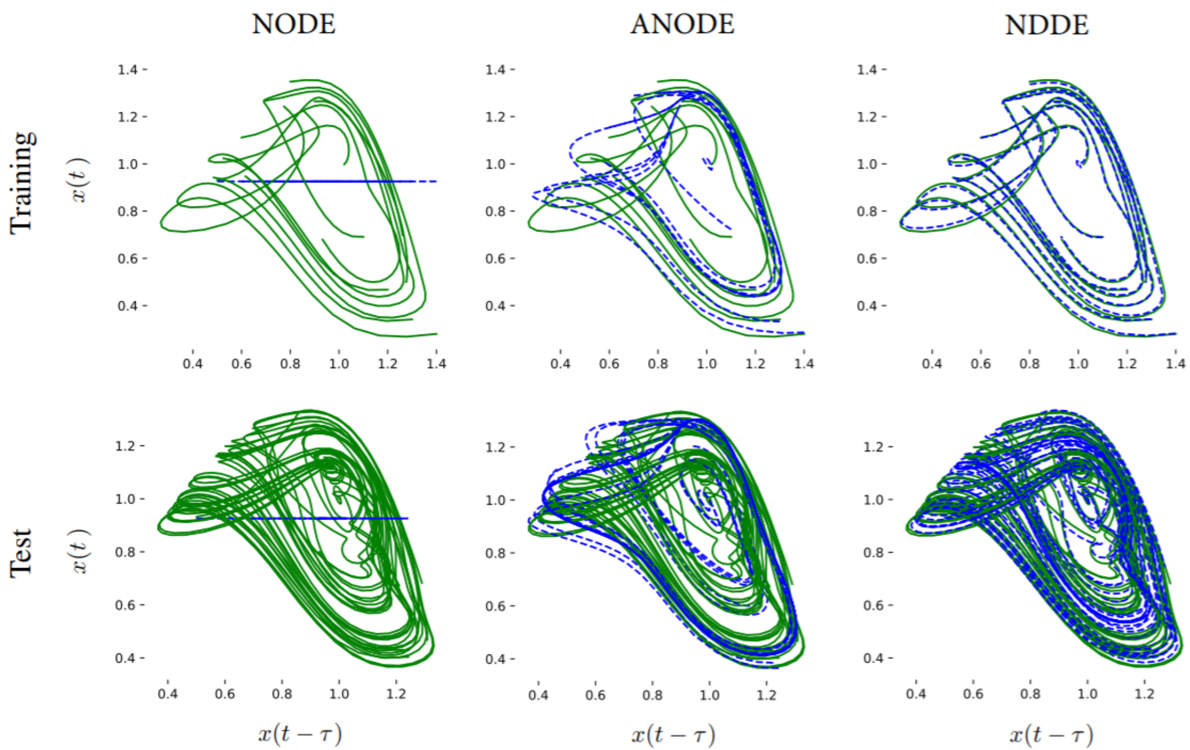
$$\dot{\boldsymbol{x}} = \boldsymbol{A}\tanh(\boldsymbol{x}(t) + \boldsymbol{x}(t-\tau)) \text{ with } \boldsymbol{x}(t) = \boldsymbol{x}_0 \text{ for } t < 0$$



Example: Mackey-Glass system

$$\dot{x} = \beta \frac{x(t-\tau)}{1+x^n(t-\tau)} - \gamma x(t)$$

- $x(t)$ is the number of the blood cells,
- β, n, τ, γ are the parameters of biological significance



Mackey, M. and Glass, L. Oscillation and chaos in physiological control systems. **Science**, 1977

Example: Image datasets

	CIFAR10	MNIST	SVHN
NODE	$53.92\% \pm 0.67$	$96.21\% \pm 0.66$	$80.66\% \pm 0.56$
NDDE	$55.69\% \pm 0.39$	$96.22\% \pm 0.55$	$81.49\% \pm 0.09$
NODE+NDDE	$\mathbf{55.89\% \pm 0.71}$	$\mathbf{97.26\% \pm 0.22}$	$\mathbf{82.60\% \pm 0.22}$
A1+NIDE	$56.14\% \pm 0.48$	$97.89\% \pm 0.14$	$81.17\% \pm 0.29$
A1+NDDE	$56.83\% \pm 0.60$	$97.83\% \pm 0.07$	$82.46\% \pm 0.28$
A1+NODE+NDDE	$\mathbf{57.31\% \pm 0.61}$	$\mathbf{98.16\% \pm 0.07}$	$\mathbf{83.02\% \pm 0.37}$
A2+NODE	$57.27\% \pm 0.46$	$98.25\% \pm 0.08$	$81.73\% \pm 0.92$
A2+NDDE	$58.13\% \pm 0.32$	$98.22\% \pm 0.04$	$82.43\% \pm 0.26$
A2+NODE+NDDE	$\mathbf{58.40\% \pm 0.31}$	$\mathbf{98.26\% \pm 0.06}$	$\mathbf{83.73\% \pm 0.72}$
A4+NODE	$58.93\% \pm 0.33$	$98.33\% \pm 0.12$	$82.72\% \pm 0.60$
A4+NDDE	$59.35\% \pm 0.48$	$98.31\% \pm 0.03$	$82.87\% \pm 0.55$
A4+NODE+NDDE	$\mathbf{59.94\% \pm 0.66}$	$\mathbf{98.52\% \pm 0.11}$	$\mathbf{83.62\% \pm 0.51}$

Table 1: The test accuracies with their standard deviations over 5 realizations on the three image datasets. In the first column, p (=1, 2, or 4) in Ap means the number of the channels of zeros into the input image during the augmentation of the image space $\mathbb{R}^{c\times h\times w} \to \mathbb{R}^{(c+p)\times h\times w}$ (Dupont et al., 2019). For each model, the initial (resp. final) time is set as 0 (resp. 1), and the delays of the NDDEs and its extensions are all set as 1, simply equal to the final time.

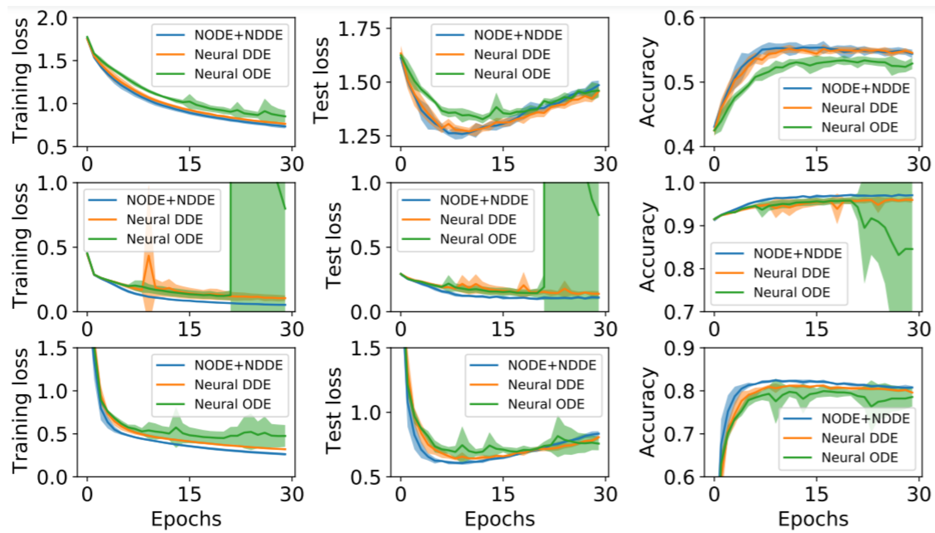


Figure 8: The training loss (left column), the test loss (middle column), and the accuracy (right column) over 5 realizations for the three image sets, i.e., CIFAR10 (top row), MNIST (middle row), and SVHN (bottom row).

Conclusion and future directions

Conclusion

NDDEs with dependency on a time delay allow to model a larger class of physical systems, in particular adding the possibility of crossing paths in phase space.

Future directions

- Applications
  - Continuous-time series modelling (Irregular-sampled, physics models)
  - Generative modelling (continuous normalizing flows)
  - Applications to traditional mathematical modelling (SIR, . . .), and traditional machine learning problems
- Differential Equations
  - Higher-Order Differential Equations
  - Stochastic Differential Equations
  - Partial differential equations
- Numerical optimization of Neural ODEs
  - Regularizing learned dynamics to be faster to solve
- And …

Thank you ！