DNN Modeling of Unknown PDEs with Incomplete Data

Zhongshu Xu

The Ohio State University

Joint work with:

Victor Churchill	Trinity College
Yuan Chen	The Ohio State University
Dongbin Xiu	The Ohio State University

Outline

- Flow Map Learning of unknown system
- Memory-based learning of reduced dynamical systems
- Nodal space DNN modeling of unknown PDEs
- Modeling unknown PDEs with incomplete data
- Conclusion

Flow Map Learning of unknown system[1]

From Euler forward to ResNet

1: T. Qin, K. Wu, and D. Xiu, Data driven governing equations approximation using deep neural networks, Journal of Computational Physics, 395 (2019), pp. 620 – 635.

Problem Setup:

autonomous ODE system

- ODE system: $\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}), \quad \mathbf{x}(0) = \mathbf{x}_0$
- Flow map: $\mathbf{\Phi} : \mathbb{R}^n \to \mathbb{R}^n, \quad \mathbf{x}(t + \Delta t) = \mathbf{\Phi}_{\Delta t}(\mathbf{x}(t))$
- Data: $\mathbf{X} = \left\{\mathbf{x}\left(t_j^{(k)}\right)\right\}, \quad j = 1, \ldots, \ell^{(k)}, \quad k = 1, \ldots, N_{\text{traj}}$
- Model to learn: $\mathcal{N} \approx \mathbf{\Phi}_{\Delta t}$
- Prediction: $\mathcal{N}^{(n)}(\mathbf{x}(0)) \approx \mathbf{x}(n \cdot \Delta t)$

Mathematical Motivation:

Euler forward = ResNet Curve

- Exact Euler forward: $\mathbf{x}(\Delta) = \mathbf{\Phi}_\Delta(\mathbf{x}(0))$

$$
\begin{aligned}
&= \mathbf{x}(0) + \int_0^\Delta \mathbf{f}(\mathbf{x}(t))dt \\
&= \mathbf{x}(0) + \Delta \cdot \mathbf{f}(\mathbf{x}(\tau)) \\
&= \mathbf{x}(0) + \Delta \cdot \mathbf{f}(\mathbf{\Phi}_\tau(\mathbf{x})), \quad 0 \leq \tau \leq \Delta \\
&\triangleq \mathbf{x}(0) + \phi_\Delta(\mathbf{x}; \mathbf{f}).
\end{aligned}
$$

- ResNet Curve: $\mathbf{N}(\mathbf{x}; \Theta) \approx \phi_\Delta(\mathbf{x}; \mathbf{f}) \quad \mathcal{N} = \mathbf{I} + \mathbf{N}$

- Training: $\Theta^* = \text{argmin} \frac{1}{N} \sum_{i=1}^{N} ||\mathbf{x}^i(\Delta) - \mathcal{N}(\mathbf{x}^i(0))||_2^2$

Memory-based learning for reduced systems[2]

Mori-Zwanzig formulation Inspired Neural Network

2: X. Fu, L.-B. Chang, and D. Xiu, Learning reduced systems via deep neural networks with memory, J. Machine Learning Model. Comput., 1 (2020), pp. 97–118.

Problem Setup:

Partially observed autonomous ODE systems

- ODE system: $\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}), \quad \mathbf{x}(0) = \mathbf{x}_0$
- Partially observed:

  Let $\mathbf{x} = (\mathbf{z}; \mathbf{w})$, where $\mathbf{z} \in \mathbb{R}^d$ is available and $\mathbf{w} \in \mathbb{R}^{n-d}$ is un-observable.
- Data we have: $\mathbf{Z} = \left\{\mathbf{z}\left(t_j^{(k)}\right)\right\}, \quad j = 1, \ldots, \ell^{(k)}, \quad k = 1, \ldots, N_{\text{traj}}$
- Flow-map to learn: $\mathcal{N} \approx \mathbf{\Phi}_{\Delta t} : \mathbf{z}(0) \longrightarrow \mathbf{z}(\Delta t)$

Mathematical Motivation:

Mori-Zwanzig formulation

- MZ formulation:

  The evolution of the reduced set of variables z follows generalized Langevin equation in the following form:

$$\frac{d}{dt}\mathbf{z}(t) = \mathbf{R}(\mathbf{z}(t)) + \int_0^t \mathbf{K}(\mathbf{z}(t-s), s)ds + \mathbf{F}(t, \mathbf{x}_0)$$

- Approximate MZ formulation:

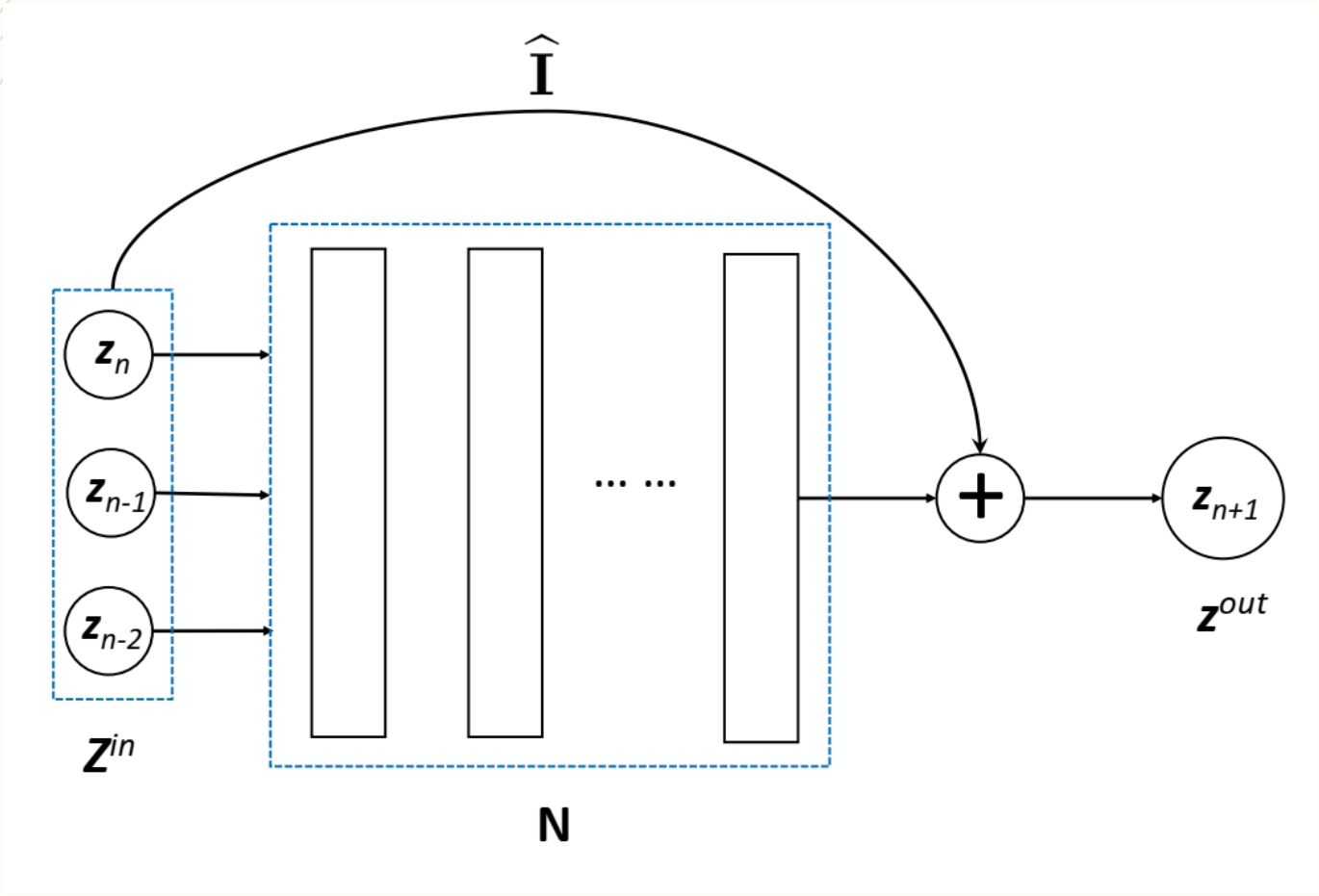
$$\frac{d}{dt}\hat{\mathbf{z}}(t) = \mathbf{R}(\hat{\mathbf{z}}(t)) + \int_0^{T_M} \mathbf{K}(\hat{\mathbf{z}}(t-s), s)ds$$

- Discrete approximate MZ formulation:

$$\left.\frac{d}{dt}\tilde{\mathbf{z}}(t)\right|_{t=t_n} = \mathbf{R}(\tilde{\mathbf{z}}(t))|_{t=t_n} + \mathbf{M}(\tilde{\mathbf{z}}_{n-n_M}, \ldots, \tilde{\mathbf{z}}_{n-1}, \tilde{\mathbf{z}}_n)$$

Neural Network structure

$$\mathbf{z}_{n+1} = \mathbf{z}_n + \mathbf{N}(\mathbf{z}_n, \mathbf{z}_{n-1}, \ldots, \mathbf{z}_{n-n_M}; \Theta)$$



DNN modeling of unknown PDEs[3]

flow-map based deep learning in nodal space

3: Z. Chen, V. Churchill, K. Wu, and D. Xiu, Deep neural network modeling of unknown partial differential equations in nodal space, Journal of Computational Physics, 449 (2022), p. 110782.

Problem Setup:

autonomous time-dependent PDE

- PDE: $\begin{cases} u_t = \mathcal{L}(u), & (x,t) \in \Omega \times \mathbb{R}^+ \\ \mathcal{B}(u) = 0, & (x,t) \in \partial\Omega \times \mathbb{R}^+ \\ u(x,0) = u_0(x), & x \in \bar{\Omega} \end{cases}$
- Set of nodal points: $X_N = \{x_1, \ldots, x_N\} \subset \Omega$
- Data: $\mathbf{u}(t) = (u(x_1,t), \ldots, u(x_N,t))^T$

$$\mathbf{U} = \left\{ \mathbf{u}\left(t_j^{(k)}\right) \right\}, \quad j = 1, \ldots, \ell^{(k)}, \quad k = 1, \ldots, N_{\text{traj}}$$

- Flow-map to learn: $\mathbf{\Phi} : \mathbf{u}(0) \longrightarrow \mathbf{u}(\Delta t)$

Mathematical Motivation:

- p-th order autonomous PDE: $u_t = \mathcal{L}\left(u, \partial^{(1)}u, \ldots \partial^{(p)}u\right)$
- Exact Euler forward: $\mathbf{u}(t+\Delta t) = \mathbf{u}(t) + \Delta t \cdot (\mathbf{\Psi}_\tau(\mathbf{u}(t)))$
- Proposition:

There exists a set of functions $\{F_i : \mathbb{R}^N \to \mathbb{R}^N, i = 1, \ldots, J\}, J \geq 1$, and an iterative scheme

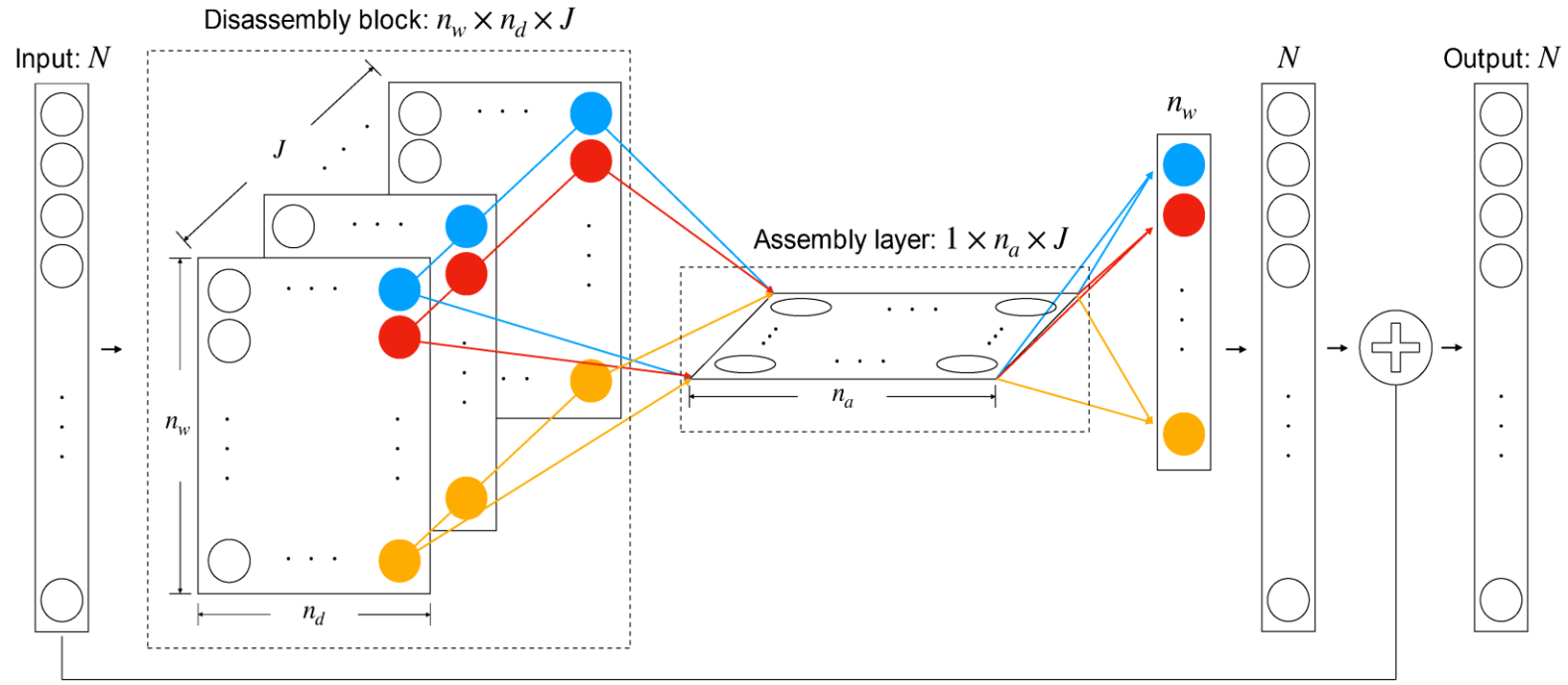
$$\mathbf{v}(t+\Delta t) = \mathbf{v}(t) + \Delta t \cdot \mathcal{M}[F_1(\mathbf{v}(t)), \cdots, F_J(\mathbf{v}(t))]$$

where $\mathcal{M}$ is a (nonlinear) function operated component-by-component, such that for sufficiently large J, the exact solution satisfies

$$\mathbf{u}(t+\Delta t) = \mathbf{u}(t) + \Delta t \cdot (\mathcal{M}[F_1(\mathbf{u}(t)), \cdots, F_J(\mathbf{u}(t))] + \boldsymbol{\tau})$$

Disassembly-Assembly Neural Network structure

$$\mathbf{v}(t+\Delta t) = \mathbf{v}(t) + \Delta t \cdot \mathcal{M}[F_1(\mathbf{v}(t)), \cdots, F_J(\mathbf{v}(t))]$$



Numerical Example:

1D uniform grid

- 4th-order PDE: $\frac{\partial u}{\partial t} + c\frac{\partial^4 u}{\partial x^4} = 0$

  with $c = 1 \times 10^{-2}$ and 2π-periodic boundary condition.

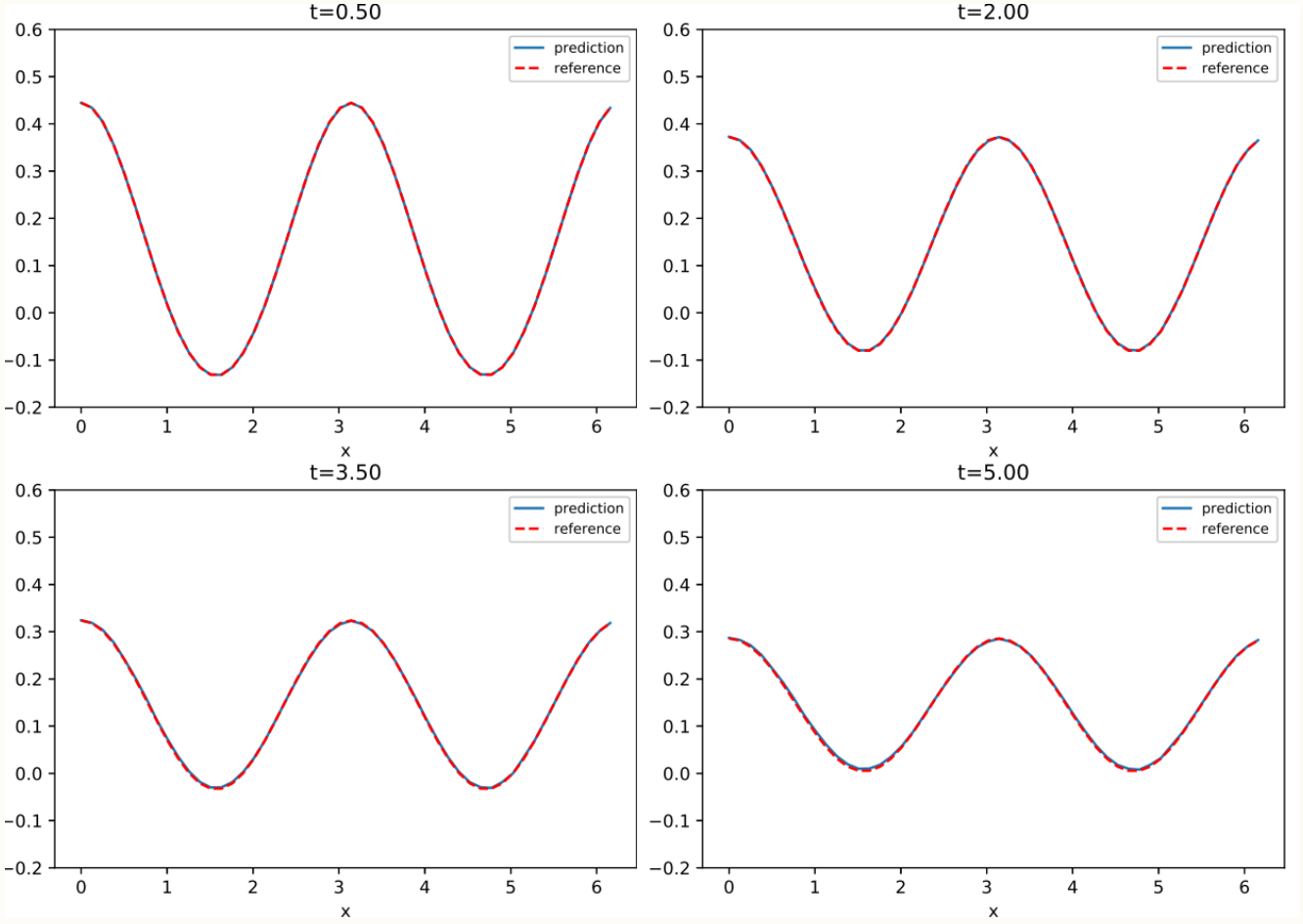
- Uniform grid: $x_i = \frac{2\pi}{N} i, i = 0, \ldots, N-1$ with $N = 50, \Delta t = 0.01$

- Network structure: $n_d = 1, n_a = 1, n_w = N$ and $J = 5$.

- Network training: 10, 000 trajectories training set, trained for 2, 000 epochs with batch size 50.

Test result:

initial condition $u(x,0) = \exp\left(-\sin^2(x)\right) - 1/2$



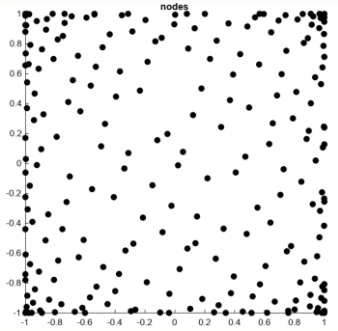
Numerical Example:

2D unstructured grid

- Advection-diffusion equation: $\frac{\partial u}{\partial t} + \nabla \cdot (\boldsymbol{\alpha} u) = \nabla \cdot (\kappa \nabla u)$

  on an unstructured grid over domain $(x, y) \in [-1, 1] \times [-1, 1]$ with zero Dirichlet boundary condition. The transport velocity field is set as $\boldsymbol{\alpha}(x, y) = (y, -x)^T$, and the viscosity is set as $\kappa = 5 \times 10^{-3}$.

- Unstructured grid: 236 random points $\Delta t = 0.002$

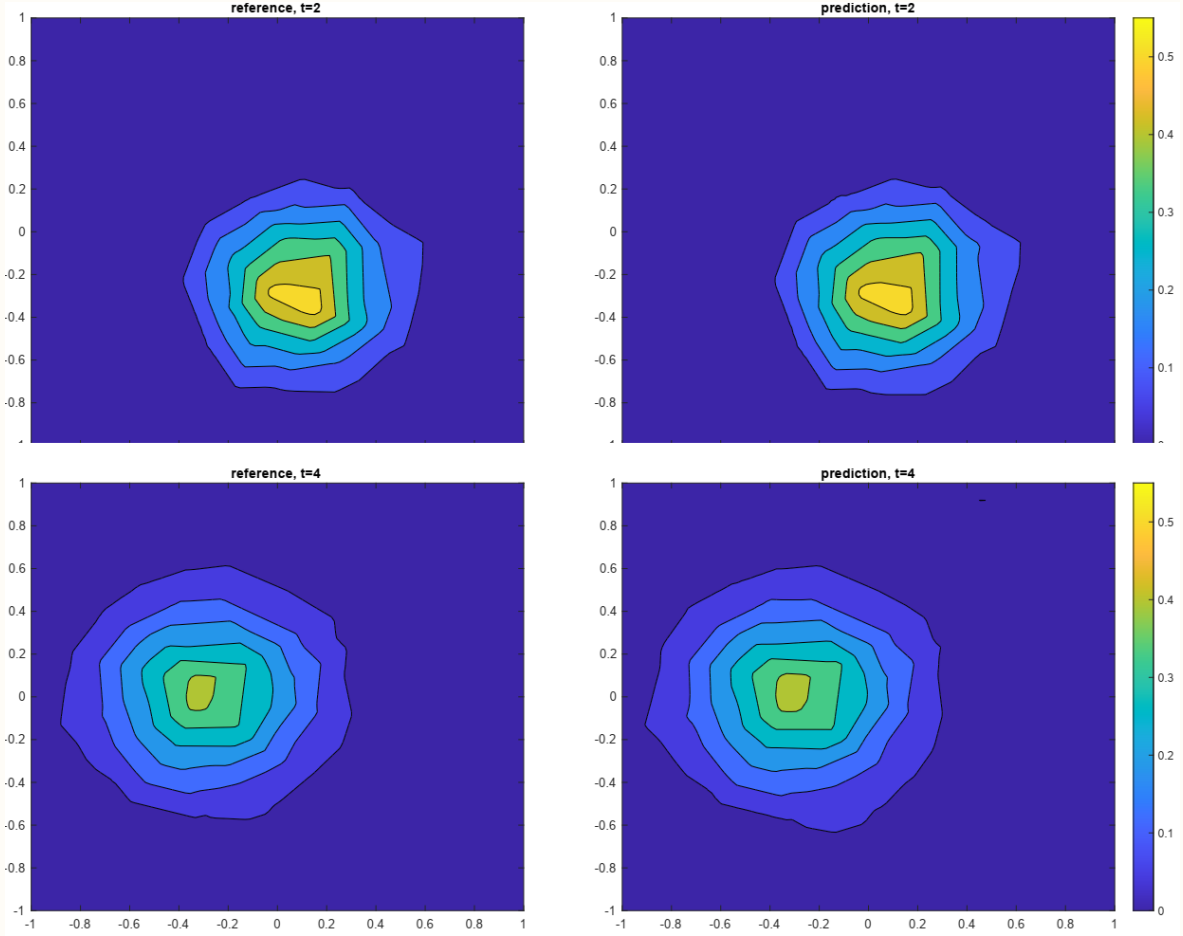


- Network structure: $n_d = 1, n_a = 1, n_w = N = 236,$ and $J = 3$

- Network training: 10, 000 trajectories training set, trained for 6, 000 epochs with batch size 100.

Test result:

initial condition $u_0(x,y) = \frac{C}{\sqrt{4\pi^2\sigma_x^2\sigma_y^2}} \exp\left[-\frac{1}{2}\frac{(x-\mu_x)^2}{\sigma_x^2} - \frac{1}{2}\frac{(y-\mu_y)^2}{\sigma_y^2}\right]$



DNN modeling of unknown PDEs with incomplete data[4]

Memory-based modeling for PDEs

4: V. Churchill, Y. Chen, Z. Xu, and D. Xiu, DNN modeling of partial differential equations with incomplete data, Journal of Computational Physics 493 (2023), p. 112502.

Problem Setup:

Partially observed autonomous time-dependent PDE systems

- PDE: $\begin{cases} u_t = \mathcal{L}(u), & (x,t) \in \Omega \times \mathbb{R}^+ \\ \mathcal{B}(u) = 0, & (x,t) \in \partial\Omega \times \mathbb{R}^+ \\ u(x,0) = u_0(x), & x \in \bar{\Omega} \end{cases}$
- Partial variables: Let $\mathbf{u} = (\mathbf{v}; \mathbf{w})$, where $\mathbf{v}$ contains the m available state variables and w contains the $n - m$ un-observable state variables.
- Partial domain: $X_N = \{x_1, \ldots, x_N\} \subset \Omega, |\Omega \setminus \mathrm{Conv}(X_N)| = \mathcal{O}(1)$
- Data we have: $\mathbf{V}(t) = (V_1(t), \ldots, V_m(t))^T \in \mathbb{R}^{m \cdot N}$, where $V_j(t) = (v_j(x_1, t), \ldots, v_j(x_N, t)) \in \mathbb{R}^N, j = 1, \ldots, m$
- Flow-map to learn: $\mathbf{\Phi} : \mathbf{V}(0) \longrightarrow \mathbf{V}(\Delta t)$

Mathematical Motivation:

Combination of memory-based modeling and Disa-Asse structure

- p-th order autonomous PDE: $u_t = \mathcal{L}\left(u, \partial^{(1)}u, \ldots \partial^{(p)}u\right)$
- Euler forward : $\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t \cdot \mathcal{L}\left(\mathbf{u}_n, \mathbf{D}^{(1)}\mathbf{u}_n, \ldots, \mathbf{D}^{(p)}\mathbf{u}_n\right)$
- Expand: $$\begin{bmatrix}\mathbf{v}\\ \mathbf{w}\end{bmatrix}_{n+1} = \begin{bmatrix}\mathbf{v}\\ \mathbf{w}\end{bmatrix}_{n} + \Delta t \cdot \begin{bmatrix}\mathcal{L}|_{\mathbf{v}}\left(\mathbf{v}_n, \mathbf{D}_{\mathbf{v}}^{(1)}\mathbf{v}_n, \ldots, \mathbf{D}_{\mathbf{v}}^{(p)}\mathbf{v}_n, \mathbf{w}_n, \mathbf{D}_{\mathbf{w}}^{(1)}\mathbf{w}_n, \ldots, \mathbf{D}_{\mathbf{w}}^{(p)}\mathbf{w}_n\right)\\ \mathcal{L}|_{\mathbf{w}}\left(\mathbf{v}_n, \mathbf{D}_{\mathbf{v}}^{(1)}\mathbf{v}_n, \ldots, \mathbf{D}_{\mathbf{v}}^{(p)}\mathbf{v}_n, \mathbf{w}_n, \mathbf{D}_{\mathbf{w}}^{(1)}\mathbf{w}_n, \ldots, \mathbf{D}_{\mathbf{w}}^{(p)}\mathbf{w}_n\right)\end{bmatrix}$$
- Using Discrete approximate MZ formulation :

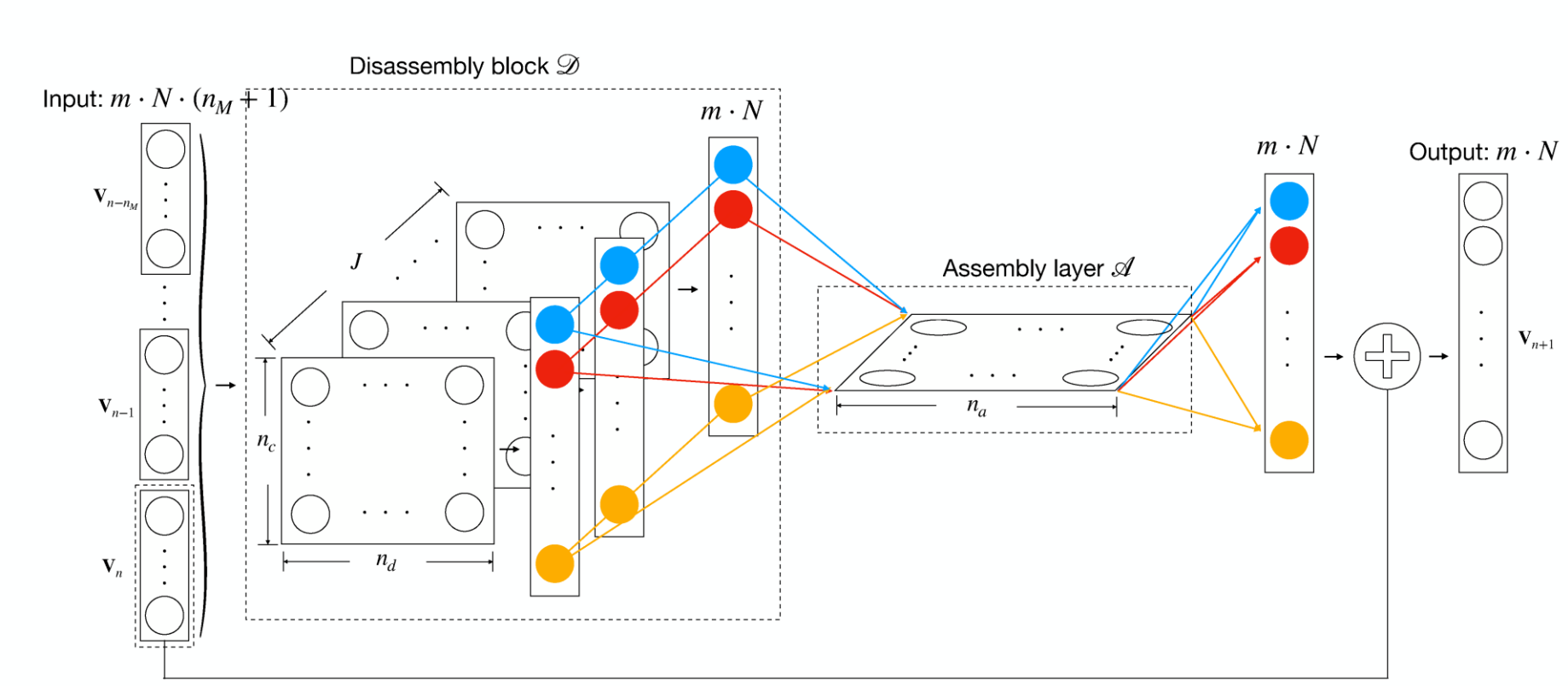
$$\mathbf{v}_{n+1} = \mathbf{v}_n + \Delta t \cdot \mathcal{L}|_{\mathbf{v}}\left(\mathbf{F}^{(1)}(\mathbf{v}_n, \ldots, \mathbf{v}_{n-n_M}), \ldots, \mathbf{F}^{(q)}(\mathbf{v}_n, \ldots, \mathbf{v}_{n-n_M})\right)$$

- Using proposition:

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \mathcal{M}[\mathcal{N}_1(\mathbf{v}_n, \ldots, \mathbf{v}_{n-n_M}), \ldots, \mathcal{N}_J(\mathbf{v}_n, \ldots, \mathbf{v}_{n-n_M})]$$

Disassembly-Assembly-Memory Neural Network structure

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \mathcal{M}[\mathcal{N}_1(\mathbf{v}_n, \ldots, \mathbf{v}_{n-n_M}), \ldots, \mathcal{N}_J(\mathbf{v}_n, \ldots, \mathbf{v}_{n-n_M})]$$



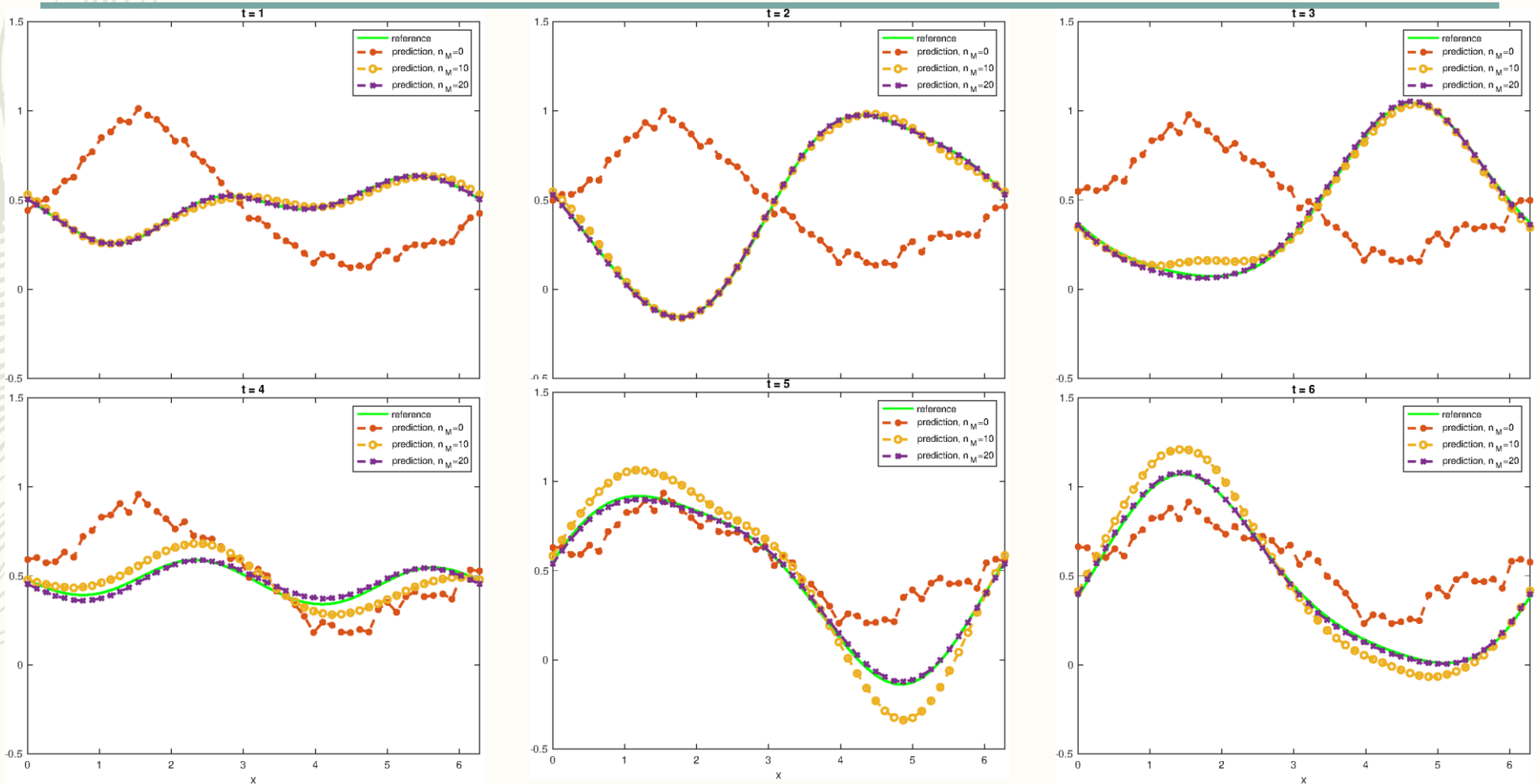
Numerical Example:

Missing Variable: Wave Equations

- PDE: $\begin{bmatrix} v \\ w \end{bmatrix}_t = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix}_x, \quad (x,t) \in [0, 2\pi] \times \mathbb{R}^+$
- full grid: $x_i = \frac{2\pi}{N} i, i = 0, \ldots, N-1$ with $N = 50, \Delta t = 0.01$
- Various memory length: $n_M = 0, 10, 20$
- Network structure: $n_d = 1, n_a = 1, n_c = 1000$ and $J = 5$
- Network training: 10, 000 trajectories training set, trained for 10, 000 epochs

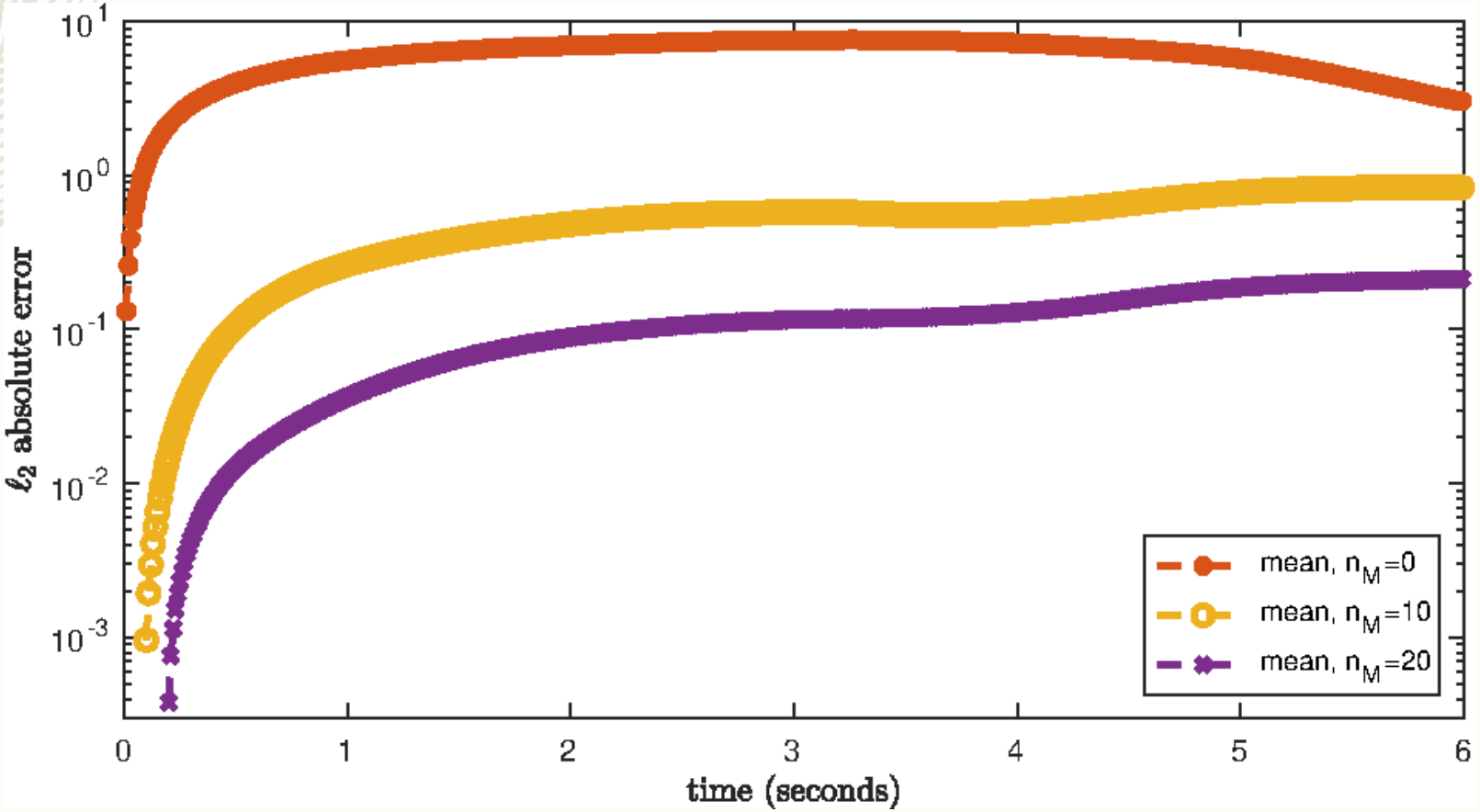
Test result:

initial condition same distribution as training set



Test result:

mean ℓ_2 absolute error of 100 test trajectories over time.



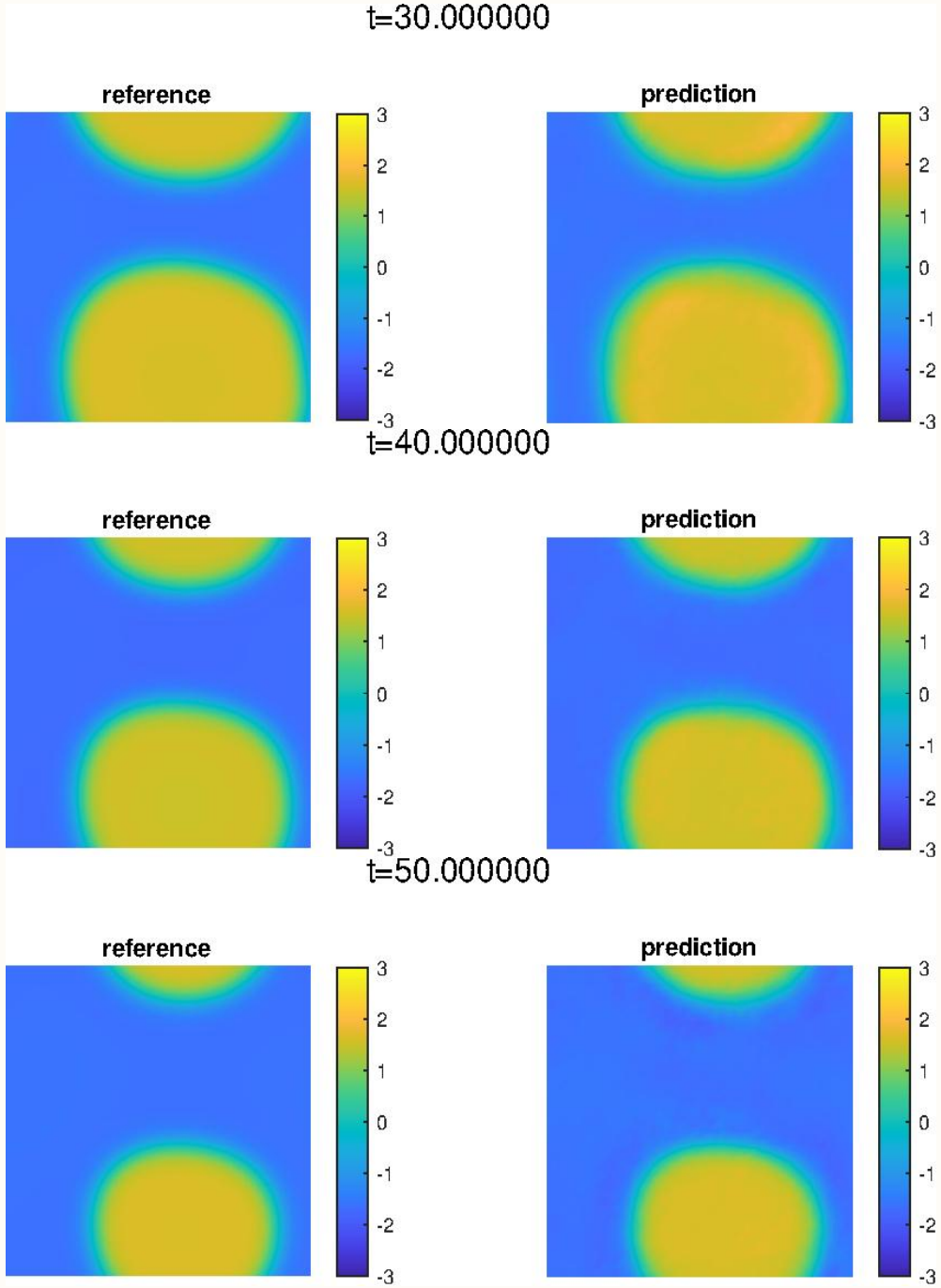
Numerical Example:

Missing Variable: 2D FitzHugh-Nagumo System

- PDE: $$\begin{aligned} v_t &= v - \frac{1}{3}v^3 - w + I + D_v \Delta v \\ w_t &= \epsilon(v + b - cw) + D_w \Delta w \end{aligned}$$ $D_v = 0.1, D_w = 1.6, \epsilon = 0.08, b = 0.7, c = 0.8, I = 0.35$
- full grid: uniform grid of $N = 50^2$ points over domain $[0, 10] \times [0, 10]$, $\Delta t = 1$
- Memory length: $n_M = 20$, input length $m \cdot N \cdot (n_M + 1) = 52500$
- Network structure: $n_d = 1, n_a = 1, n_c = 100$ and $J = 5$
- Network training: $1,000$ trajectories training set, trained for $10,000$ epochs

Test result:

initial condition same distribution as training set



Numerical Example:

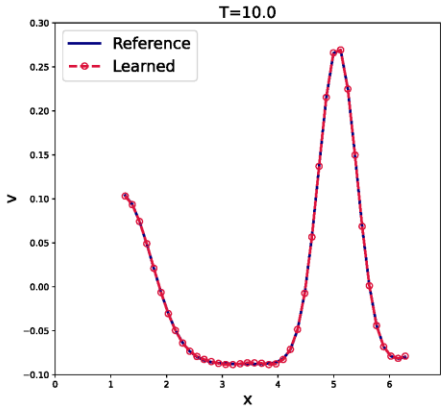
Missing Domain: Advection-Diffusion equation.

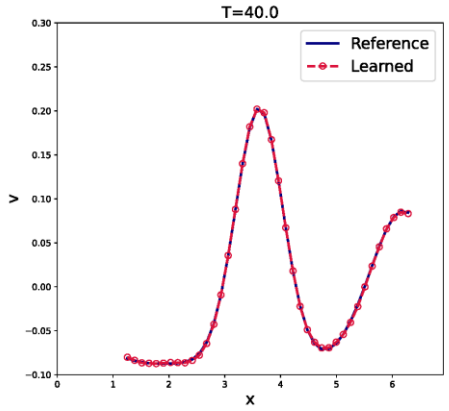
- PDE: $\frac{\partial u}{\partial x} = -\frac{\partial u}{\partial x} + \kappa \frac{\partial^2 u}{\partial x^2}, \quad x \in [0, 2\pi], \quad \kappa = 10^{-3}$
- Available domain: $\Delta x = \frac{\pi}{25}$ in (i) subdomain $\left[\frac{2}{5}\pi, 2\pi\right](80\%)$, (ii) subdomain $\left[\frac{6}{5}\pi, 2\pi\right](40\%), \Delta t = 0.1$
- Memory length: (i): $n_M = 15$, (ii): $n_M = 60$
- Network structure: $n_d = 1, n_a = 1, n_c = 40$ and $J = 5$.
- Network training: $10,000$ trajectories training set, trained for $10,000$ epochs

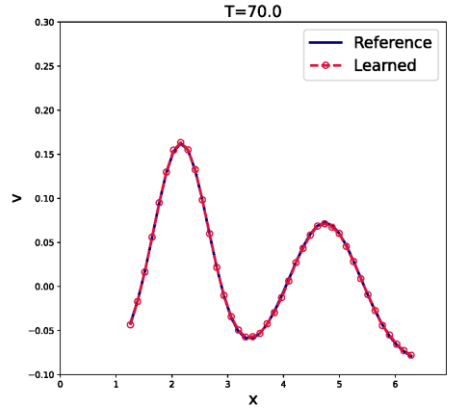
Test result:

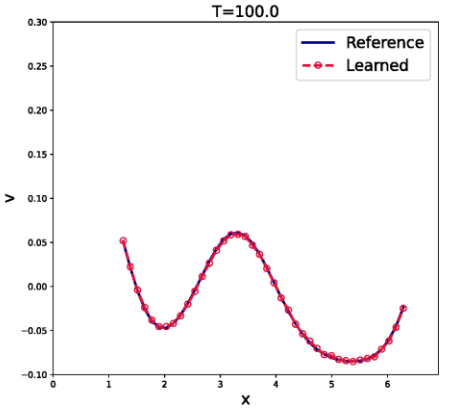
initial condition: $u_0(x) = \frac{2}{5}e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} + \frac{1}{5}e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}} - \frac{1}{5}$

80% domain, $n_M = 15$

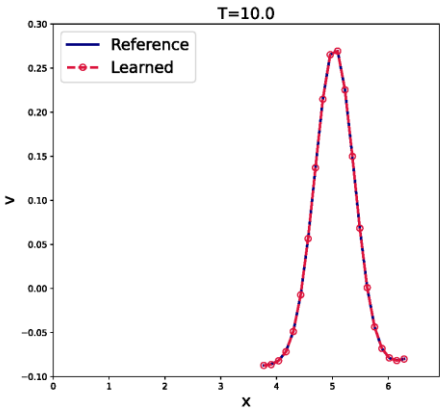


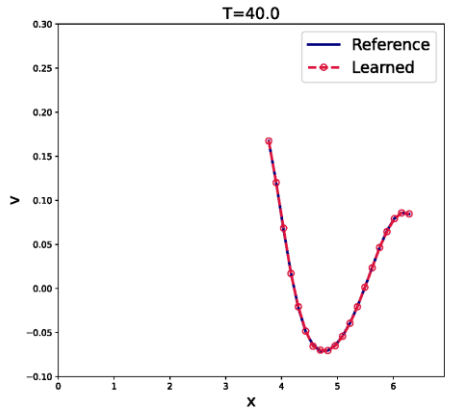


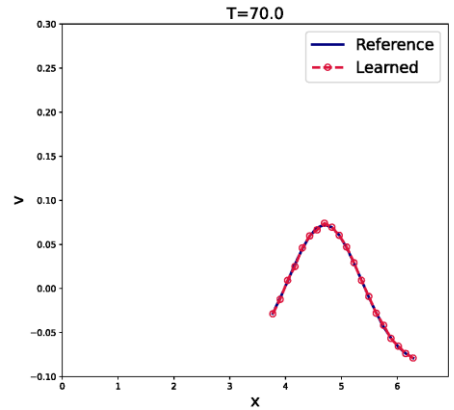


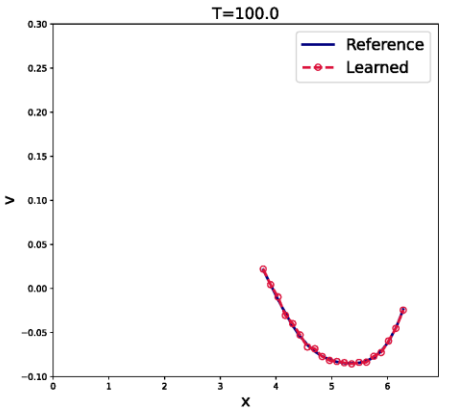


40% domain, $n_M = 60$









Numerical Example:

Missing Domain: Inviscid Burger's equation.

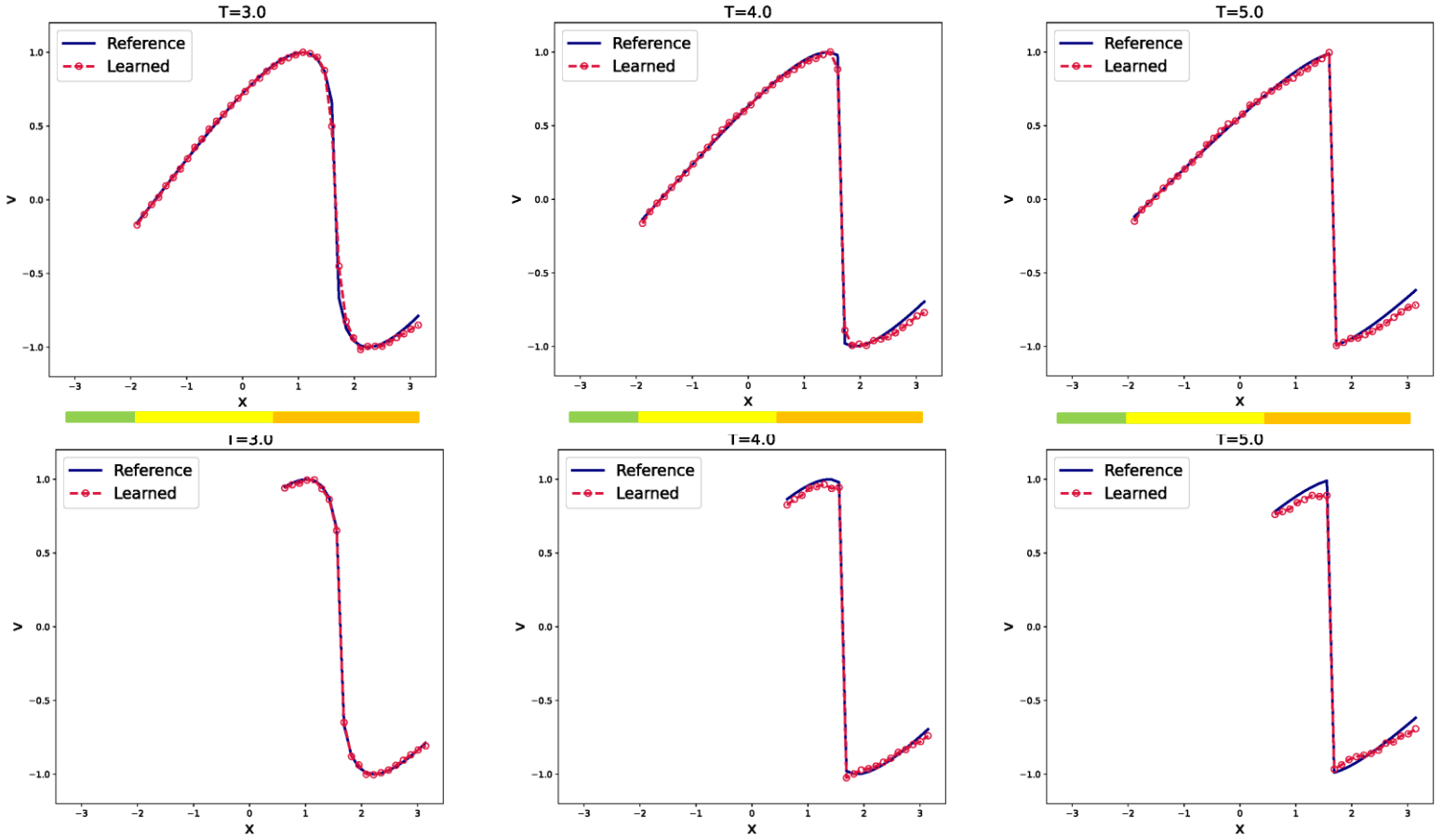
- PDE: $\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0, \quad x \in [-\pi, \pi]$
- Available domain: $\Delta x = \frac{\pi}{25}$ in (i) subdomain $\left[-\frac{3}{5}\pi, \pi\right](80\%)$, (ii) subdomain $\left[\frac{1}{5}\pi, \pi\right](40\%), \Delta t = 0.05$
- Memory length: (i): $n_M = 25$, (ii): $n_M = 50$
- Network structure: $n_d = 1, n_a = 1, n_c = 40$ and $J = 5$.
- Network training: $10,000$ trajectories training set, trained for $10,000$ epochs

Test result:

initial condition： $u_0(x) = -\sin x$

80% domain, $n_M = 25$

40% domain, $n_M = 50$



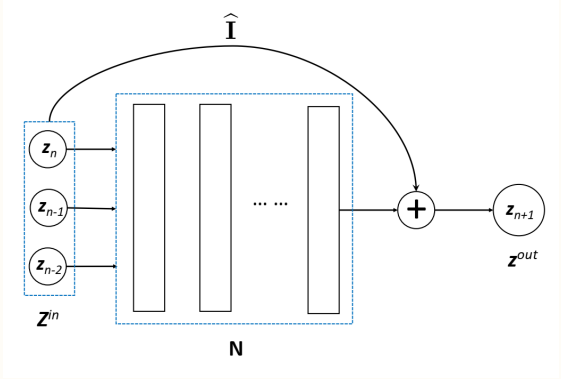
Conclusion

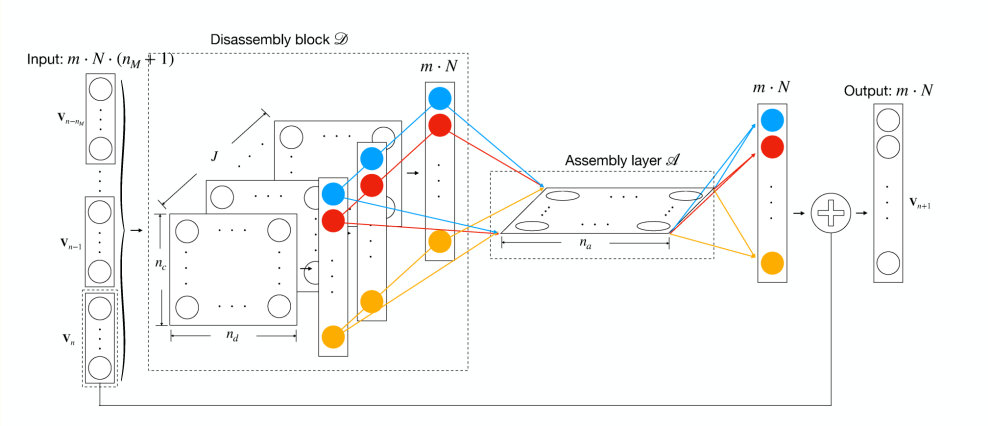
Evolution of Flow Map Learning

Flow Map Learning
ResNet

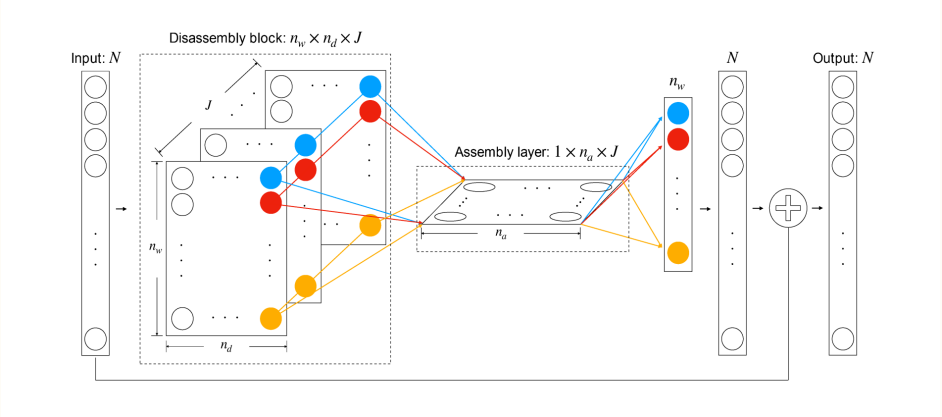
FML of reduced system
ResNet+Memory

FML of PDEs
Disassembly-Assembly





FML of PDEs with incomplete Data
Disassembly-Assembly +Memory




Thank
you