Parametric reduced order modeling for nonlocal PDEs

Yumeng Wang Yanzhi Zhang Shiping Zhou

Missouri University of Science and Technology, Rolla, USA

Problem Setting: Parameterized Partial Differential Equations (PDEs)

General parametric nonlocal PDEs :

$$\begin{aligned} \mathcal{L}_t^{\boldsymbol{\mu}} u(\mathbf{x},t;\boldsymbol{\mu}) &= \mathcal{L}_{\mathbf{x}}^{\boldsymbol{\mu}} u(\mathbf{x},t;\boldsymbol{\mu}) + \mathcal{F}(\mathbf{x},t,u;\boldsymbol{\mu}), && \text{for } \mathbf{x}\in\Omega, && t\in(0,T],\\ \mathcal{B}u(\mathbf{x},t;\boldsymbol{\mu}) &= g(\mathbf{x},t;\boldsymbol{\mu}), && \text{for } \mathbf{x}\in\Omega_\Gamma, && t\in(0,T], \end{aligned}$$

Parameters $\boldsymbol{\mu}$ can be from the equation, initial condition or boundary conditions.
Notation: $\mathcal{L}_t^{\boldsymbol{\mu}}$ and $\mathcal{L}_{\mathbf{x}}^{\boldsymbol{\mu}}$ are the linear differential operators for t and x; $\mathcal{F}, \mathcal{B}$ represents the nonlinear terms of u and a linear boundary operator, respectively.

Challenge and Goal

Challenges:

- Repeatedly solving nonlocal PDEs with varying parameters
- Numerical methods for nonlocal PDEs need intensive storage and computation cost
- It is computationally infeasible for high dimensions

Method: Reduced order modeling (ROM).
Challenges of ROM for developing nonlocal problems:

- Lack of affine dependence
- Singularity
- Nonlocal boundary conditions

Goal: A neural network based ROM, nonlinear and non-intrusive method for nonlocal parameterized PDEs.

1. Dimensional reduction with Convolutional Autoencoder (CAE)
2. Solution approximation in latent space

Proposed method

Offline training:
Reduced order model: CAE learns a low-dimensional representation of high-dimensions, via reconstructing the input data accurately with the help of convolutional layer.
Mathematics: Denote the encoder as $\Psi:\mathbb{R}^N\to\mathbb{R}^n$, decoder as $\Phi:\mathbb{R}^n\to\mathbb{R}^N$

$$\begin{aligned}\mathbf{v} &= \Psi(\mathbf{u};\boldsymbol{\vartheta}_{\mathrm{e}}),\\ \hat{\mathbf{u}} &= \Phi(\mathbf{v};\boldsymbol{\vartheta}_{\mathrm{d}}),\end{aligned}$$

Here, $\mathbf{u}\in\mathbb{R}^N, \hat{\mathbf{u}}\in\mathbb{R}^N, \mathbf{v}\in\mathbb{R}^n$, and dimension $n \ll N$.
Here convolutional encoder and decoder are trained together.

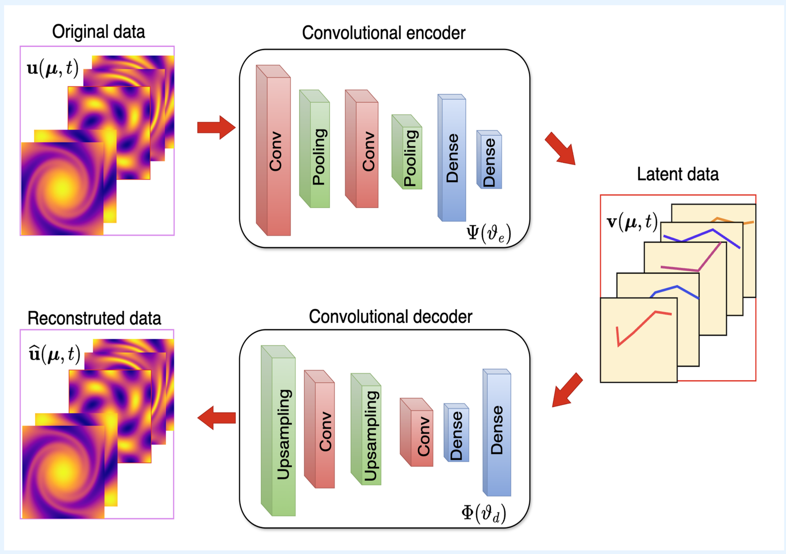

Figure 1. CAE

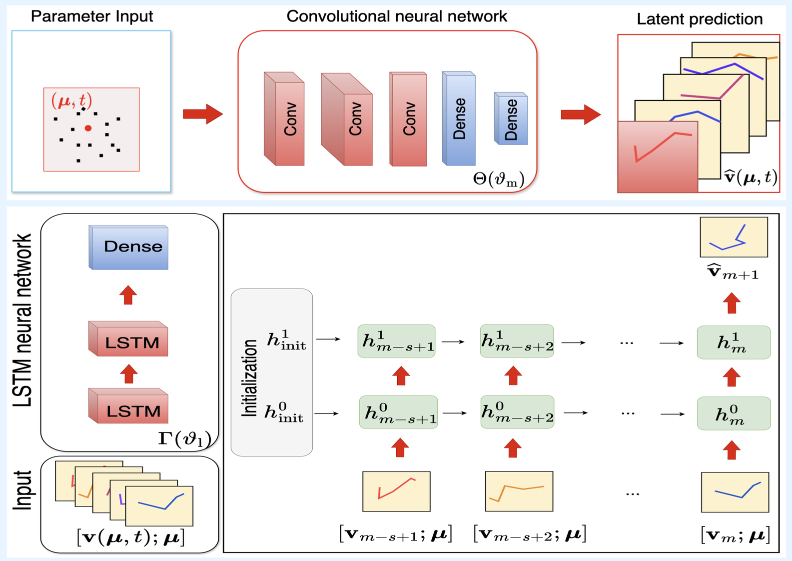

Figure 2. Latent space modeling

Reconstruction loss:

$$\mathcal{L}_{\text{CAE}}(\boldsymbol{\vartheta}_{\mathrm{e}},\boldsymbol{\vartheta}_{\mathrm{d}}) = \frac{1}{N_{\mathrm{tr}}^t N_{\mathrm{tr}}^{\boldsymbol{\mu}}}\sum_{i=1}^{N_{\mathrm{tr}}^{\boldsymbol{\mu}}}\sum_{m=1}^{N_{\mathrm{tr}}^t}\left\|\mathbf{u}(\boldsymbol{\mu}_i,t_m) - \Phi(\Psi(\mathbf{u}(\boldsymbol{\mu}_i,t_m);\boldsymbol{\vartheta}_{\mathrm{e}});\boldsymbol{\vartheta}_{\mathrm{d}})\right\|_{\mathrm{mse}}$$

Latent space modeling

- Latent mapping with convolutional neural network (CNN) learns the mapping
Mathematics:Denote CNN as Θ: $\mathbb{R}^p\times\mathbb{R}\to\mathbb{R}^n$,
$$\hat{\mathbf{v}} = \Theta(\boldsymbol{\mu},t;\boldsymbol{\vartheta}_{\mathrm{m}})$$
Note: time is processed as an additional parameter.
Mapping loss:
$$\mathcal{L}_{\text{Map}}(\boldsymbol{\vartheta}_{\mathrm{m}}) = \frac{1}{N_{\mathrm{tr}}^t N_{\mathrm{tr}}^{\boldsymbol{\mu}}}\sum_{i=1}^{N_{\mathrm{tr}}^{\boldsymbol{\mu}}}\sum_{m=1}^{N_{\mathrm{tr}}^t}\|\mathbf{v}(\boldsymbol{\mu}_i,t_m) - \Theta(\boldsymbol{\mu}_i,t_m;\boldsymbol{\vartheta}_{\mathrm{m}})\|_{\mathrm{mse}}$$
- Latent time marching with long short-term memory network (LSTM) models the temporal dynamics
Mathematics: Denote the LSTM as $\Gamma:\mathbb{R}^{n+p}\to\mathbb{R}^n$, i.e.,
$$\hat{\mathbf{v}}(\boldsymbol{\mu},t_{m+1}) = \Gamma(\{[\mathbf{v}(\boldsymbol{\mu},t_{m-k+1});\boldsymbol{\mu}]\}_{k=1}^s;\boldsymbol{\vartheta}_{\mathrm{l}})$$
Time marching loss:
$$\mathcal{L}_{\text{LSTM}}(\boldsymbol{\vartheta}_{\mathrm{l}}) = \frac{1}{(N_{\mathrm{tr}}^t - s)N_{\mathrm{tr}}^{\boldsymbol{\mu}}}\sum_{i=1}^{N_{\mathrm{tr}}^{\boldsymbol{\mu}}}\sum_{m=s}^{N_{\mathrm{tr}}^t}\|\mathbf{v}(\boldsymbol{\mu}_i,t_{m+1}) - \Gamma(\{[\mathbf{v}(\boldsymbol{\mu},t_{m-k+1});\boldsymbol{\mu}]\}_{k=1}^s;\boldsymbol{\vartheta}_{\mathrm{l}})\|_{\mathrm{mse}}$$

Note:A decoupled training strategy
Online test:
Given the unseen parameters $\boldsymbol{\mu}_{\text{new}}$, CNN mapping:

$$(\boldsymbol{\mu}_{\text{new}},t_{\text{new}}) \xrightarrow{\Theta_m(\boldsymbol{\vartheta}_{\mathrm{m}}^*)} \hat{\mathbf{v}}(\boldsymbol{\mu}_{\text{new}},t_{\text{new}}) \xrightarrow{\Phi(\boldsymbol{\vartheta}_{\mathrm{d}}^*)} \hat{\mathbf{u}}(\boldsymbol{\mu}_{\text{new}},t_{\text{new}})$$

Examples

- Benchmark Test: Nonlocal parameterized 2D Poisson Equation on the domain $\Omega=(-1,1)^2$:
$$\begin{aligned}(-\Delta)^{\frac{\alpha}{2}}u(\mathbf{x}) &= f(\mathbf{x},\mu_1,\mu_2), && \text{for } \mathbf{x}\in\Omega,\\ u(\mathbf{x}) &= 0, && \text{for } \mathbf{x}\in\mathbb{R}^2\backslash\Omega,\end{aligned}$$
where
$$f(\mathbf{x},\mu_1,\mu_2) = \exp[-2((x-\mu_1)^2 + (y-\mu_2)^2)],$$
for $\alpha\in(0,2]$, and $\mu_1,\mu_2\in[-1,1]$.
Data: High-fidelity data are prepared by numerically solving the equation, detail are removed.
- Nonlocal parameterized 2D Allen–Cahn equation defined on $\Omega=(-1,1)^2$, $t\in(0,10]$
$$\partial_t u(\mathbf{x},t) = 10\int_\Omega J_\varepsilon(\mathbf{x}-\mathbf{x}')\big(u(\mathbf{x}',t) - u(\mathbf{x},t)\big)d\mathbf{x}' + \beta(u-u^3),$$
where $J_\varepsilon(\mathbf{x}-\mathbf{x}') = \frac{1}{\varepsilon}e^{-|\mathbf{x}-\mathbf{x}'|^2/\varepsilon^2}$ for $\varepsilon>0$. and the initial condition is
$$u(\mathbf{x},0) = 2\cos(2\pi x)\cos(2\pi y), \quad \text{for } \mathbf{x}\in\bar{\Omega},$$
Here the parameters are $\varepsilon\in[0.05,0.1]$ and $\beta\in[0.1,0.5]$.

Numerical Results

Benchmark results: Latent dimension = 4, compared with original dimension = 256^2 for 931 parameters.

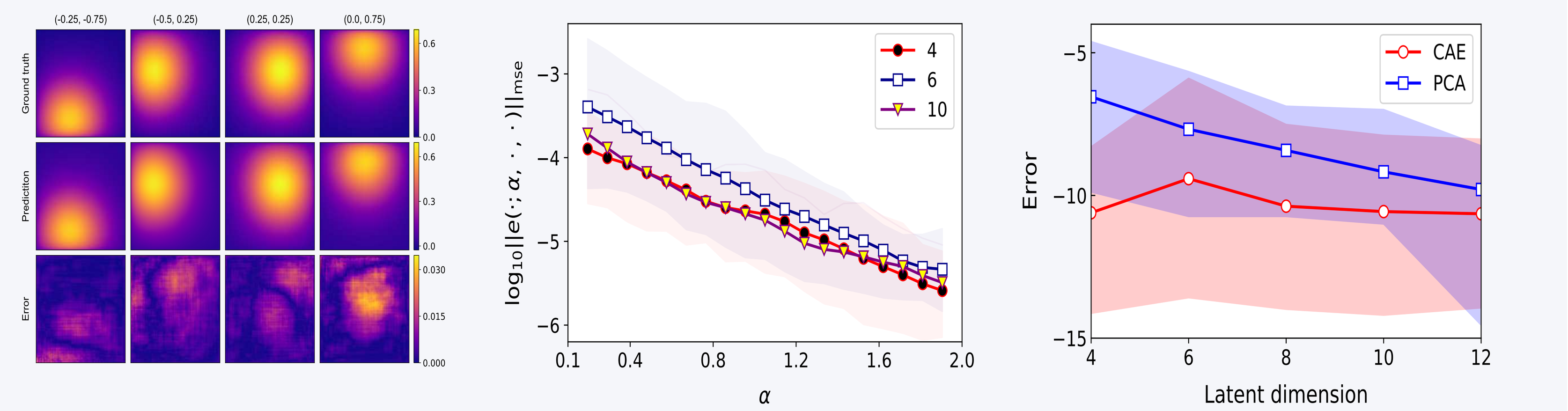

Figure 3. From left to right: Performance for $\alpha=0.48$; α log loss for different latent dimension; CAE vs PCA

2D Allen Cahn Equation:

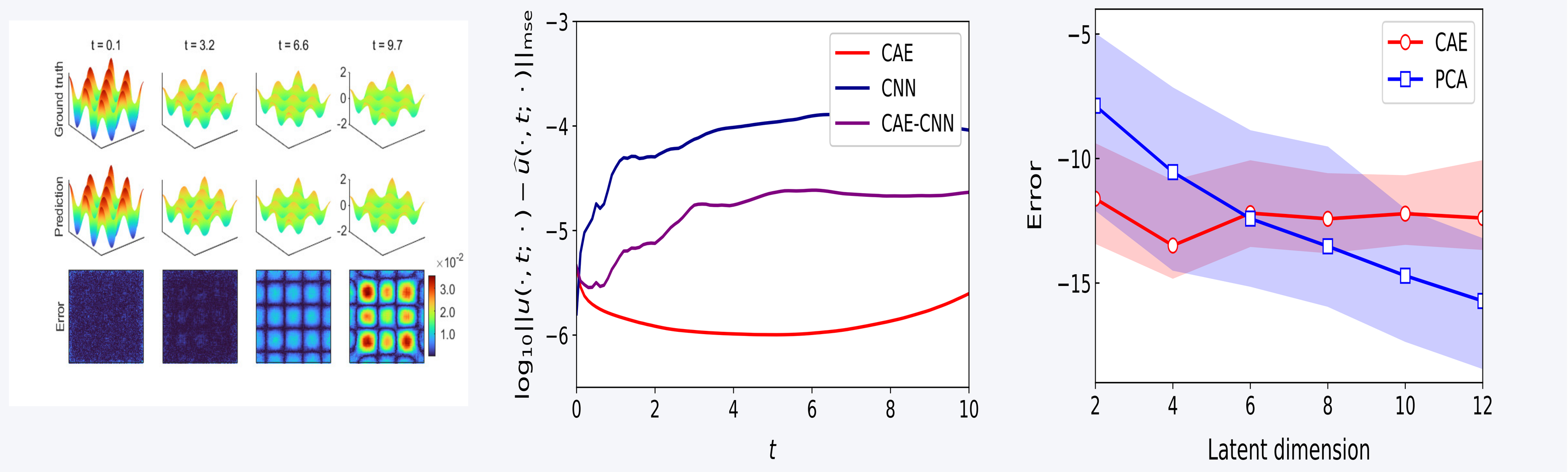

Figure 4. From left to right: $\epsilon=0.0955$ and $\beta=0.468$; model performance; CAE vs PCA

Computational efficiency for 5000 parameters

	Train/Test	$N=2^{14}$	$N=2^{16}$
Offline Training	CAE	3.552e+4	9.107e+4
	CNN	230	443
Online Prediction	$t=1$	2.999	9.074
	$t=10$	3.05	9.095
Numerical Methods	$t=1$	3.264e+4	5.335e+5
	$t=10$	3.334e+5	5.240e+6

Table 1. Computational time of our model & traditional method (seconds)

Summary

Our proposed method is the first neural network based ROMs to solve parameterized nonlocal PDEs

- The framework separates high-dimensional prediction of nonlocal parameterized PDEs into two sub-tasks: reduce order model and latent space modeling
- CAE reduce computational cost significantly for nonlocal nonlinear and nonintrusive problem
- CNN-mapping learns continuous mapping both in time and parameters space
- LSTM-time marching for time dependent PDEs, learning temporal dynamics alternatively.

References

[1] Y. Wang, S. Zhou, and Y. Zhang. Convolutional neural network-based reduced-order modeling for parametric nonlocal pdes. 2024(submit).