Parametric Model Reduction with Convolutional Neural Networks

Yumeng Wang Yanzhi Zhang Shiping Zhou

Missouri University of Science and Technology, Rolla, USA

Problem Setting: Parameterized Partial Differential Equations (PDEs)

Parameterized PDEs :

$$\partial_t u(\mathbf{x},t;\xi) + \mathcal{N}([u(\mathbf{x},t;\xi);\xi]) + \mathcal{L}([u(\mathbf{x},t;\xi);\xi]) = f(\mathbf{x},t;\xi), \quad (\mathbf{x},t,\xi) \in \Omega \times \mathcal{T} \times \mathcal{P},$$

with following properly defined initial and boundary conditions:

$$\begin{aligned} u(\mathbf{x},t;\xi) &= u_0(\mathbf{x};\xi), & t &= t_0, \\ \mathcal{B}[u(\mathbf{x},t;\xi);\xi] &= h(\mathbf{x},t;\xi), & \mathbf{x} &\in \partial\Omega. \end{aligned}$$

Parameters: ξ can be from equation, initial condition or boundary conditions.
Notation: $u(\mathbf{x},t;\xi)$ denotes the solution of PDE. $\mathcal{N}[\cdot]$, $\mathcal{L}[\cdot]$ and $\mathcal{B}[\cdot]$ are nonlinear,linear and boundary operator, respectively.

Motivation and Goal

Motivation: To find a parameterized PDE solver $\Gamma : \mathbb{R}^P \to \mathbb{R}^N$

$$u = \Gamma(\xi), \qquad \xi \in \mathbb{R}^P, u \in \mathbb{R}^N$$

- **Challenges:** Repeatedly solving PDE with varying parameters
- **High dimensions:** Computationally infeasible for high dimensions

Goal: Solve parameterized PDE efficiently.

- Construct parameterized PDEs solver by neural networks.
- Reduce high-dimensions to low-dimensions.

Convolutional Neural Network

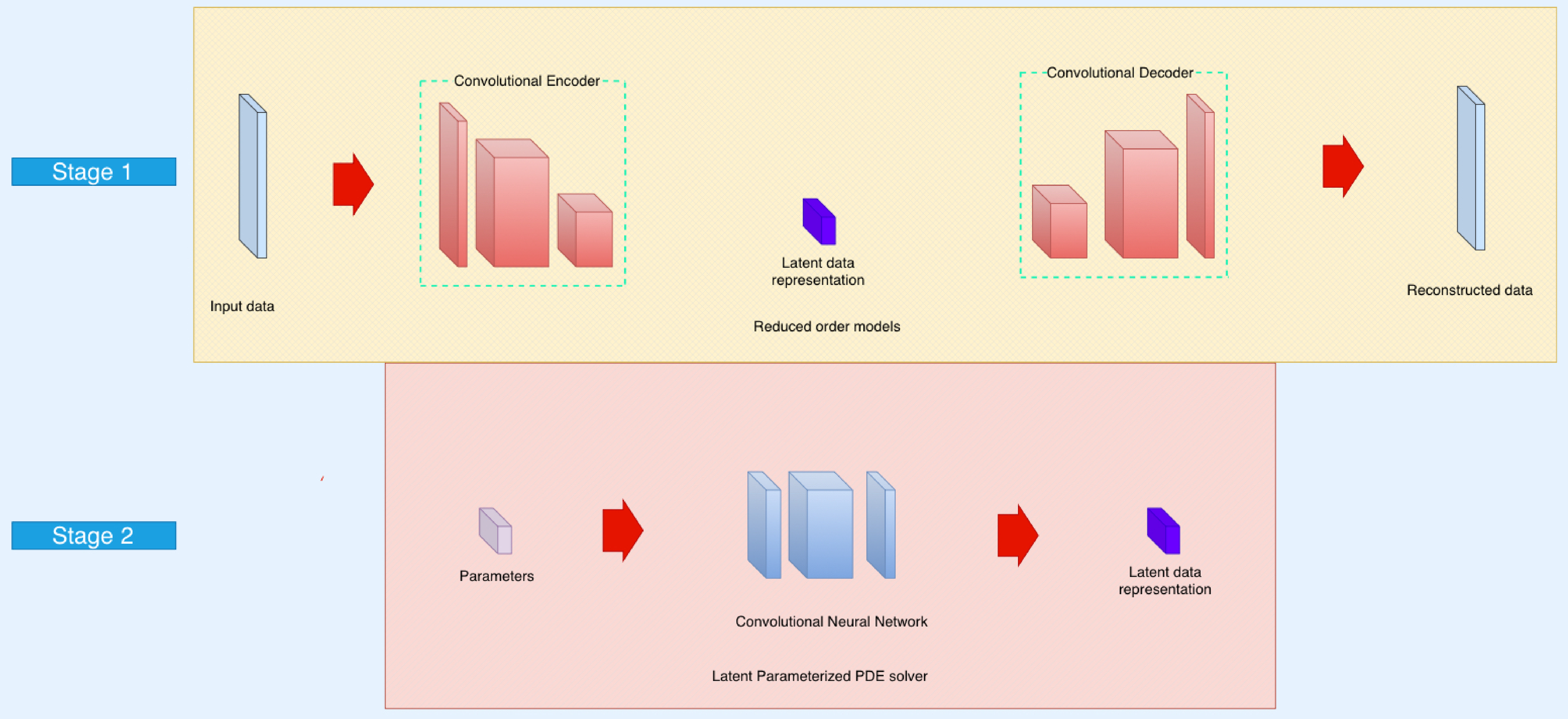
Convolutional Neural Network (CNN): CNN utilizes convolution operations effectively capture spatial features. It mainly contains:

- **Convolutional layer:** It uses learnable filters to capture hierarchical features.
- **Pooling layer:** It downsamples feature map obtained from the convolutional layer, which helps reduce dimension.

Notice: We use CNN as the basic structure to solve parameterized PDE.

Proposed Parameterized PDE Solver

Framework:



Stage1: Reduced order models via CAE

Convolutional Autoencoder (CAE): It serves as a data-driven **nonlinear** reduced order models (ROMs). Encoder compresses the data into a **low-dimensional latent representation** and decoder reconstruct it back to high-dimensions.

Mathematics: Encoder $\Psi(\psi)$: $\mathbb{R}^N \to \mathbb{R}^H$, Decoder $\Phi(\phi)$: $\mathbb{R}^H \to \mathbb{R}^N$

$$\hat{u}(\xi) = \Phi(\Psi(u(\xi);\psi);\phi)$$
$$h(\xi) = \Psi(u(\xi);\psi)$$
$$\hat{u}(\xi) = \Phi(h(\xi);\phi)$$

Here $H \ll N$ and $u(\xi) \in \mathbb{R}^N$

Reconstruction loss:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N} |u_i(\xi) - \Phi(\Psi(u_i(\xi);\psi);\phi)|^2$$

Notice: Encoder and decoder are trained together.

Stage2: Regression via CNN

Mapping: CNN is implemented to learn a mapping from parameters to latent solutions in the low-dimensional latent space.

Mathematics: Latent parameterized PDE solver $\Theta(\theta)$: $\mathbb{R}^P \to \mathbb{R}^H$

$$\hat{h}(\xi) = \Theta(\xi;\theta)$$

Reconstruction loss:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N} |h_i(\xi) - \Theta(\xi;\theta)|^2$$

Here $h_i(\xi)$ is latent representation from convolutional encoder $\Psi(\psi)$.
Notice: For time-dependent PDEs, t as an additional parameter.

Training strategy: We take separated training strategy instead of coupled training strategy for its advantages:

- Fast convergence
- Computationally efficiency

Testing strategy: For a given new parameter q with well-trained neural networks:

$$\hat{u}(q) = \Psi(\Theta(q;\theta);\beta)$$

Examples

2D Buckley Leverett equation

$$\begin{aligned} \partial_t u + \partial_x(f_1(u)) + \partial_y(f_2(u)) - \mu\nabla^2 u &= 0, \quad \text{for } x,y \in \Omega, \ t \in (0,T] \\ u &= 0, \quad \text{for } x,y \in \partial\Omega, \end{aligned}$$

where the initial condition is

$$u(x,y,0) = \begin{cases} 1, & \text{for } x^2+y^2 < 0.5, \\ 0, & \text{otherwise,} \end{cases}$$

Additionally, f_1 and f_2 are the fluxes that are non-linear functions of the field variable u,

$$f_1(u) = \frac{u^2}{u^2+(1-u)^2}, \quad f_2(u) = f_1(u)(1-5(1-u)^2)$$

Here is the parameter $\omega = (-1.5,1.5)^2$, $\mu \in [0.01,0.1]$, and $T = 0.5$.

Data preparation: 10 PDEs are numerically solved with uniformly sampled parameters, mesh size $(h_x,h_y) = (\frac{3}{255},\frac{3}{255})$ and timesteps $= 0.05$. Randomly sampled another 11 PDEs are used as test data.

Numerical Results

Prediction: Latent dimension $= 6$, compared with original dimension $= 256^2 = 65536$.

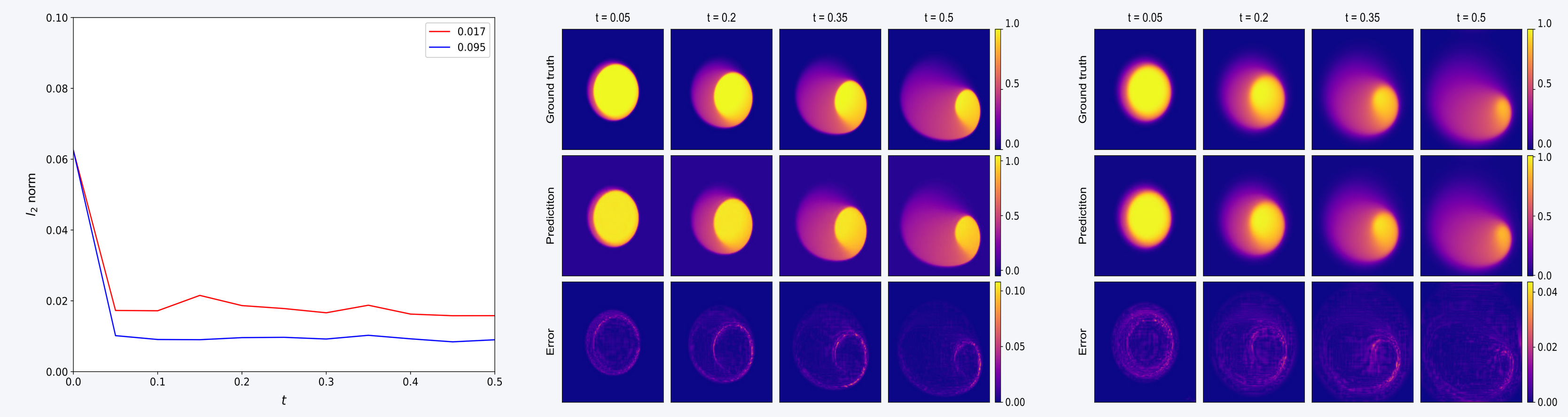


Figure 1. From left to right: l_2 norm for $\mu = 0.017$, $\mu = 0.095$ and corresponding absolute error evaluation

Comparison with POD for compression ability:

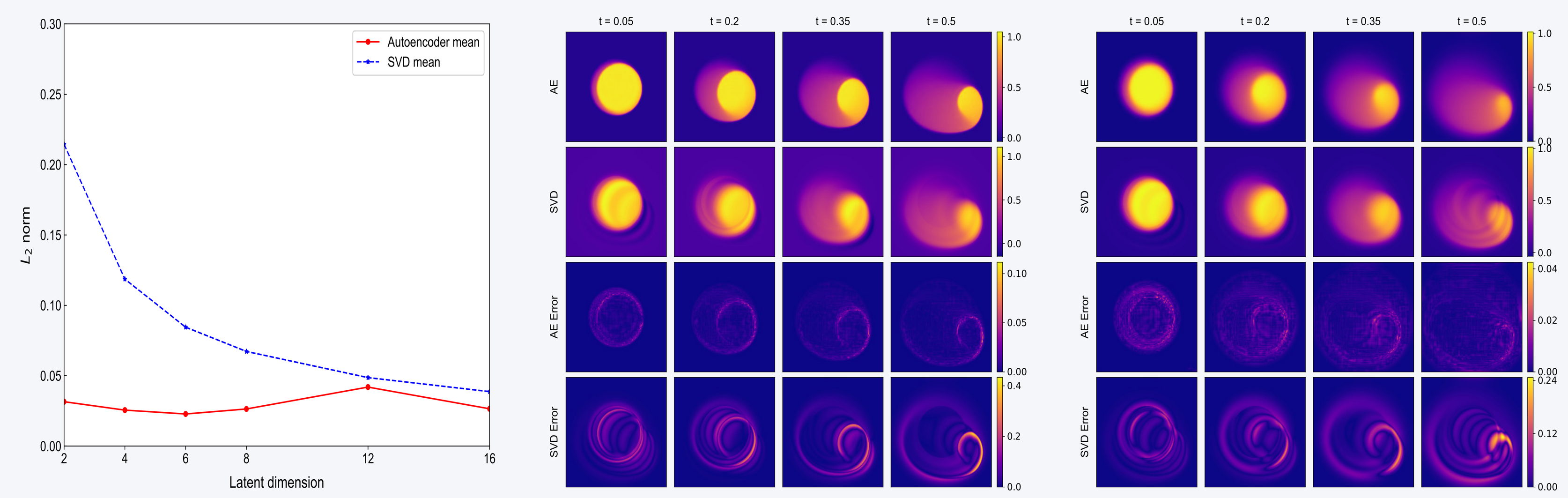


Figure 2. From left to right: Comparison with POD among parameter space and sample at $\mu = 0.017$, $\mu = 0.095$

Results Analysis and Summary

- **Our proposed framework:** Learning a latent parameterized PDE solver with CAE can give competitive results with computational efficiency.
- Decoupled training strategy is more efficient and appropriate.
- Convolutional autoencoder outperform traditional reduced order models (POD), with a low reconstruction error and low latent dimension.
- The optimal latent dimension need to be determined, usually determined by reconstruction loss empirically. But POD can be as a upper bound reference.
- Our proposed method can predict solutions in arbitrary time for time-dependent parameterized PDEs, not constrained by training timestep.

References

[1] R. Maulik, B. Lusch, and P. Balaprakash. Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders. *Physics of Fluids*, 33(3):037106, 2021.

[2] Y. Wang, Y. Zhang, and S. Zhou. Parametric model reduction with convolutional neural network. to be submitted (March,2024).