siam In cooperation with Society for Industrial and Applied Mathematics

JOHNS HOPKINS WHITING SCHOOL of ENGINEERING

Pushing the Boundaries of Surrogate Modeling: Neural Operators Integrated Numerical Simulators

Dibakar Roy Sarkar and Somdatta Goswami
Civil and Systems Engineering
Johns Hopkins University

SIAM Conference on Computational Science and Engineering (SIAM-CSE25) March 3, 2025

Physics-based Models

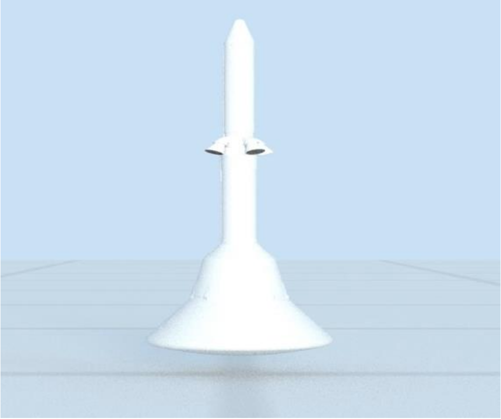
Can represent the **Processes of Nature**

- Physics-based models are approximated via **ODEs/PDEs**

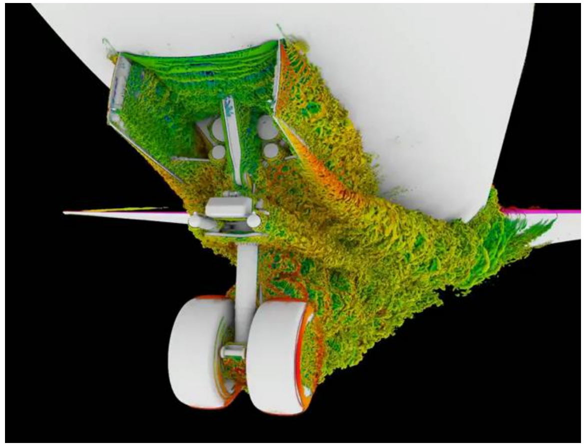
  To model earthquake: $m\frac{d^2u}{dt^2}+\mathrm{k}\frac{du}{dt}+F_0=0$

  To model waves: $\frac{\partial^2 u}{\partial t^2}-\nu^2\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)=0$

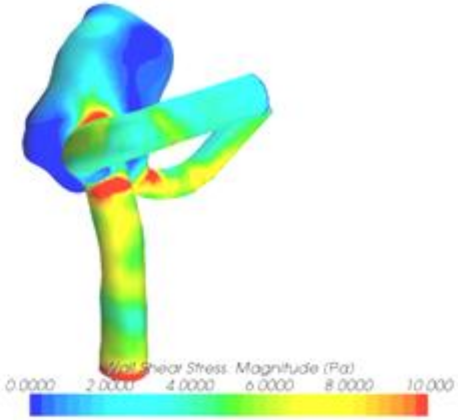
- Computational Mechanics helps us simulate these equations.



Simulation of Orion Spacecraft Launch Abort System (NASA Ames)



Detailed flow around an Aircraft's landing gear (NASA Ames)



CFD Simulation of a Patient-Specific Intracranial Aneurysm

Challenges

Traditional approaches in computational mechanics have undergone remarkable progress, but still, they operate under stringent requirements

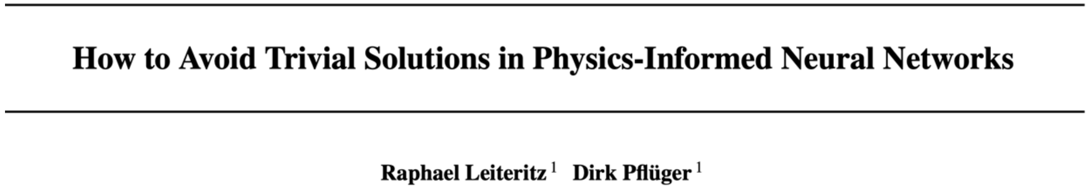
- Require precise knowledge of the governing equations
- Require knowledge of boundary conditions
- Arduous workflows (e.g., mesh generation) and long simulation times
- Assimilating observational data is costly.
- Solving inverse problems or discovering missing physics can be prohibitively expensive.
- Series of assumptions introduce many free parameters and sources of uncertainty.

Leveraging ML-based solvers

With the growing interest in ML-based PDE solvers, this is definitely an option. However, this has its own challenges:

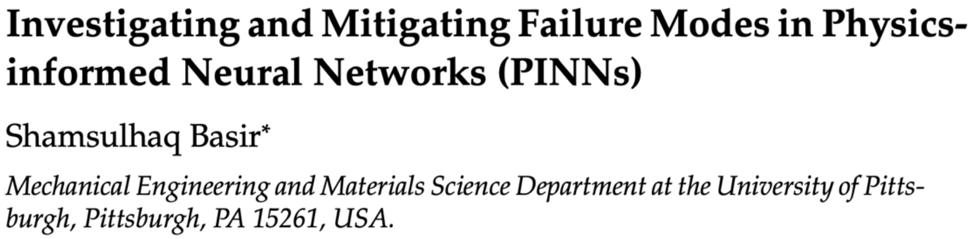
- For data driven ML Model: requires voluminous amount of high-fidelity training dataset – extensive parametric sweep on the numerical solvers.
- For physics-informed ML solvers -
  - PINNs – Not generalizable
  - Physics-Informed Neural Operators – extremely expensive, no proofs on error boundedness for generalization accuracy.

Physics-Informed ML models



How to Avoid Trivial Solutions in Physics-Informed Neural Networks

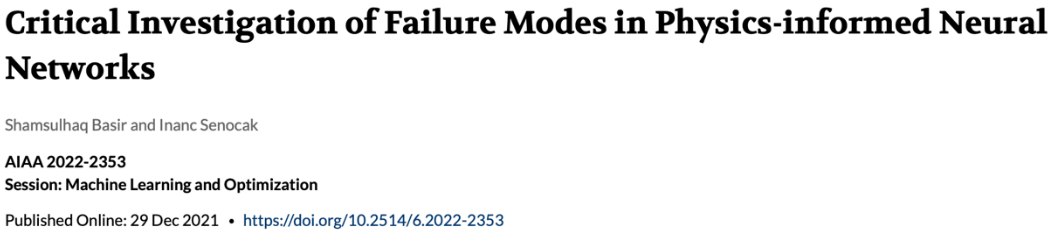
Raphael Leiteritz [1] Dirk Pflüger [1]



Investigating and Mitigating Failure Modes in Physics-informed Neural Networks (PINNs)

Shamsulhaq Basir*

Mechanical Engineering and Materials Science Department at the University of Pittsburgh, Pittsburgh, PA 15261, USA.

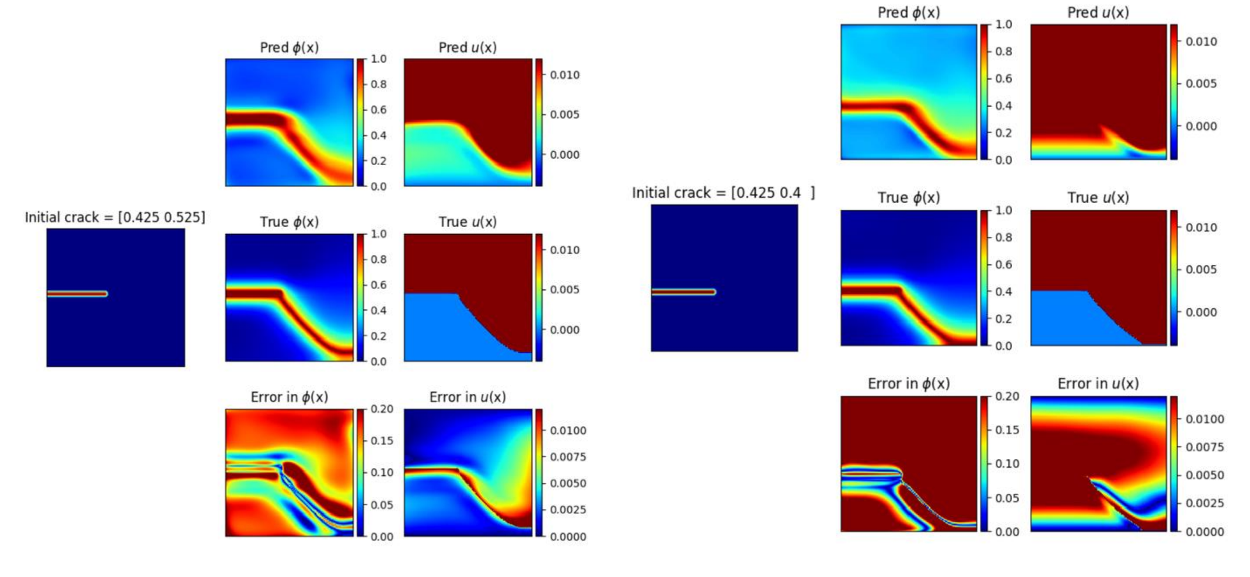


Critical Investigation of Failure Modes in Physics-informed Neural Networks

Shamsulhaq Basir and Inanc Senocak

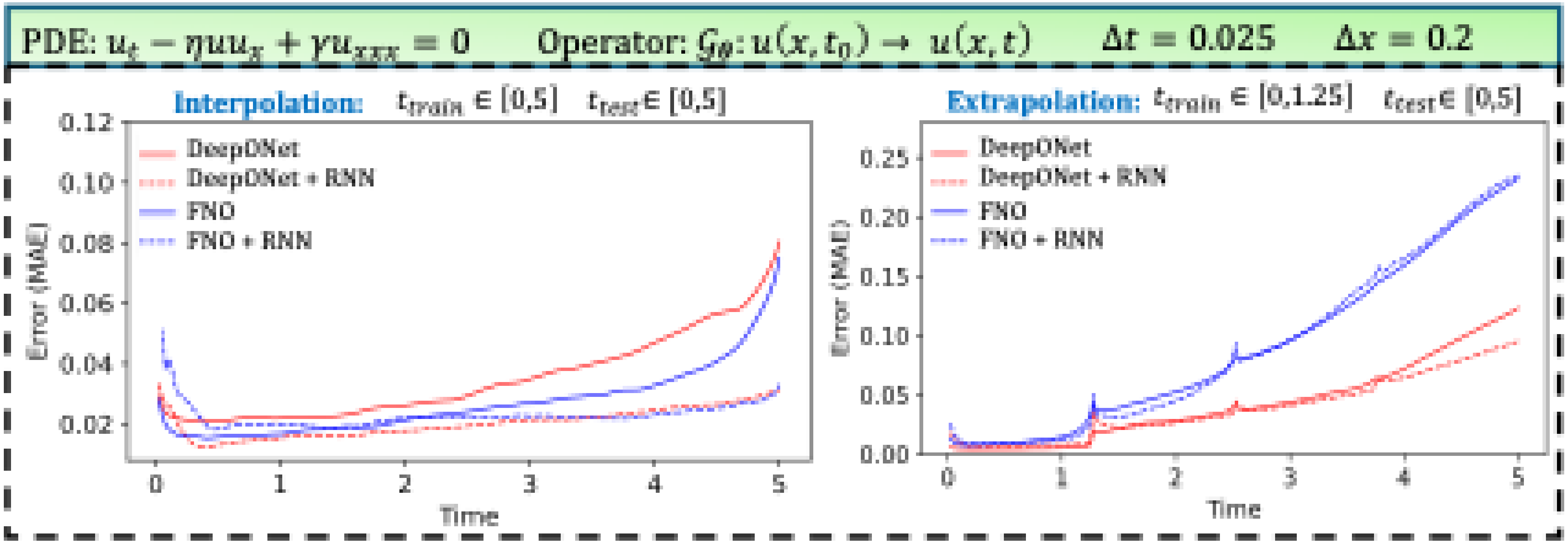
AIAA 2022-2353
Session: Machine Learning and Optimization

Published Online: 29 Dec 2021 • https://doi.org/10.2514/6.2022-2353



A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials – Goswami et. al, CMAME, 2022

Data-driven ML Models for Dynamical Systems

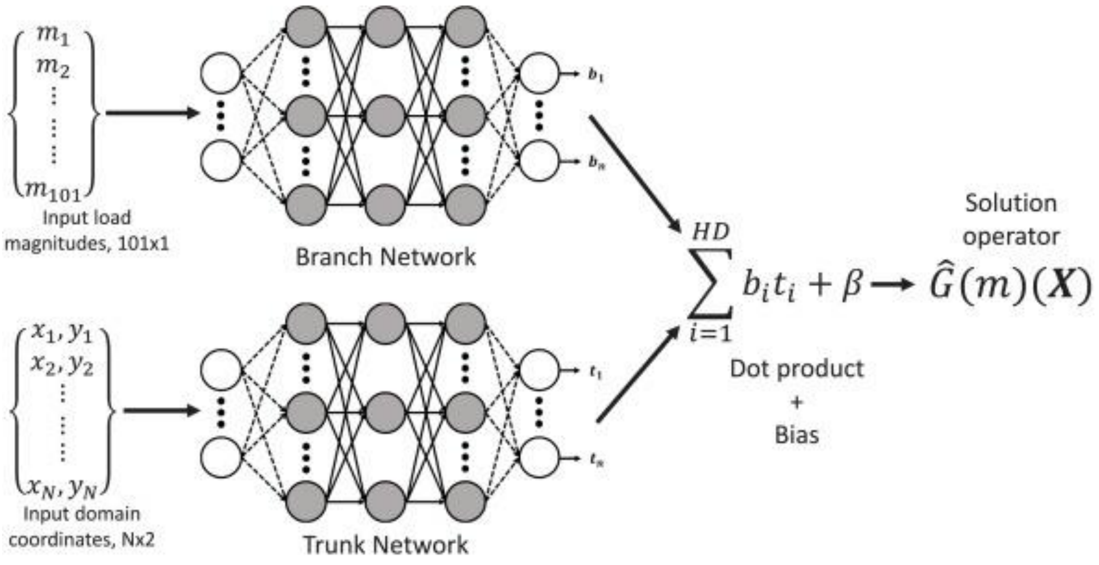


Michałowska, K., Goswami, S., Karniadakis, G. E., & Riemer-Sørensen, S. (2024, June). Neural operator learning for long-time integration in dynamical systems with recurrent neural networks. In *2024 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8). IEEE.

Our goal is to develop Physics-Informed ML-Integrated Numerical Simulators

Physics-Informed ML-Integrated Numerical Simulators

Physics-Informed ML models



m_1
m_2
m_{101}
Input load magnitudes, 101x1
Branch Network
b_1
b_n
x_1, y_1
x_2, y_2
x_N, y_N
Input domain coordinates, Nx2
Trunk Network
t_1
t_n
Solution operator
$\sum_{i=1}^{HD} b_i t_i + \beta \rightarrow \hat{G}(m)(\boldsymbol{X})$
Dot product
+
Bias

1. Forcing function.
2. Advection velocity.
3. Boundary conditions.
4. …

Numerical Solvers

FEniCS project

LAMMPS

SIMULIA

ABAQUS

Ansys

Cantera

Physics-Informed ML-integrated Numerical Simulators

Part – I: Separable Physics-Informed Neural Operators

Part – II: Domain Decomposition Approaches for Integration

Physics-Informed ML-integrated Numerical Simulators

Part – I: Separable Physics-Informed Neural Operators

Part – II: Domain Decomposition Approaches for Integration

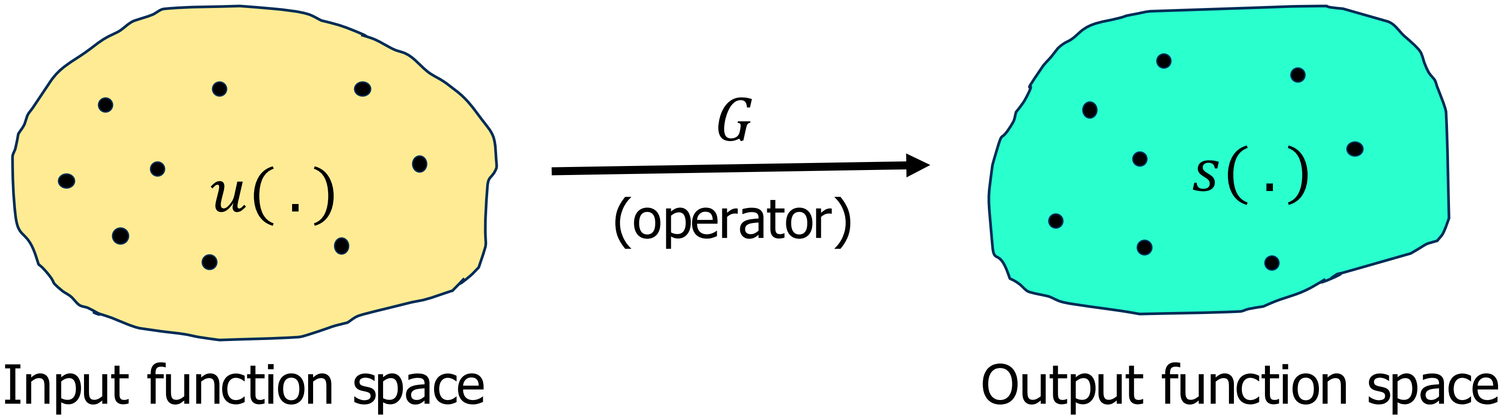
Operator Learning Framework

Input-output map

$\Phi: \mathcal{U} \rightarrow \mathcal{S}$

Data $\{\mathcal{U}_n, \mathcal{S}_n\}_{n=1}^{N}$and/or Physics

$\mathcal{S}_n = \Phi(\mathcal{F}_n)$, $\mathcal{F}_n \sim \mu \; i.i.d$



$u(.)$

G

(operator)

$s(.)$

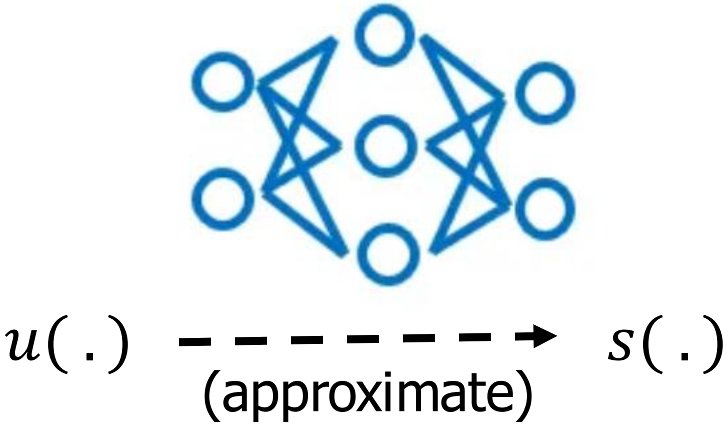
Input function space

Output function space

Operator learning

$\Psi: \times\, \Theta \rightarrow \mathcal{S}$ such that $\Psi(.,\theta^*) \approx \Phi$

Training $\theta^* = \mathrm{argmin}_{\theta}\, l(\{\mathcal{U}_n, \Psi(\mathcal{S}_n, \theta)\})$



$u(.)$ - - - - - - → $s(.)$

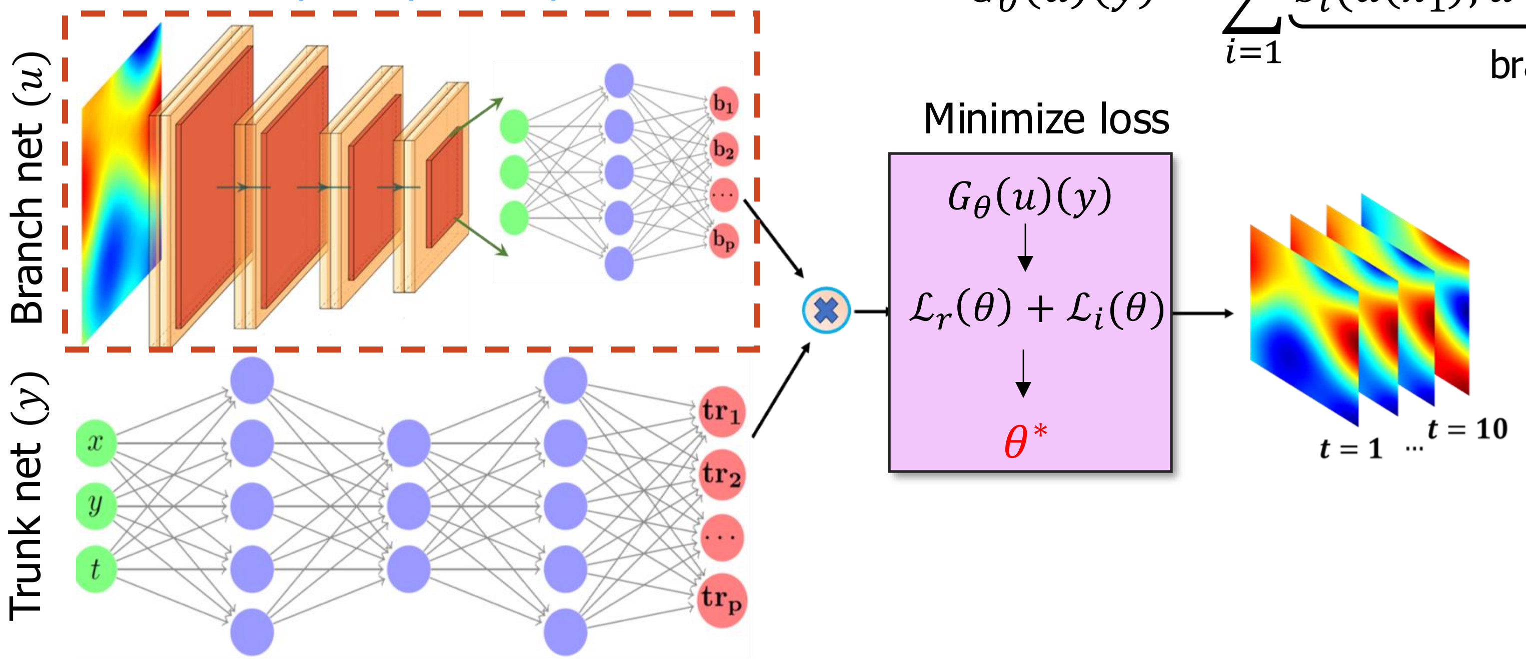
(approximate)

Deep Operator Network (DeepONet)

- Generalized Universal Approximation Theorem for Operator [Chen ’95, Lu et al. ’19]
- **Branch net**: Input $\{u(x_i)\}_{i=1}^{m}$, output: $\left[b_1, b_2, .., b_p\right]^T \in \mathbb{R}^p$
- **Trunk net**: Input y, output: $\left[t_1, t_2, .., t_p\right]^T \in \mathbb{R}^p$
- Input u is evaluated at the fixed locations $\{y_i\}_{i=1}^{m}$

$$G_\theta(u)(y) = \sum_{i=1}^{p} \underbrace{b_i(u(x_1), u\,(x_2), \ldots, u\,(x_m))}_{\text{branch net}} \cdot \underbrace{tr_i(y)}_{\text{trunk net}}$$

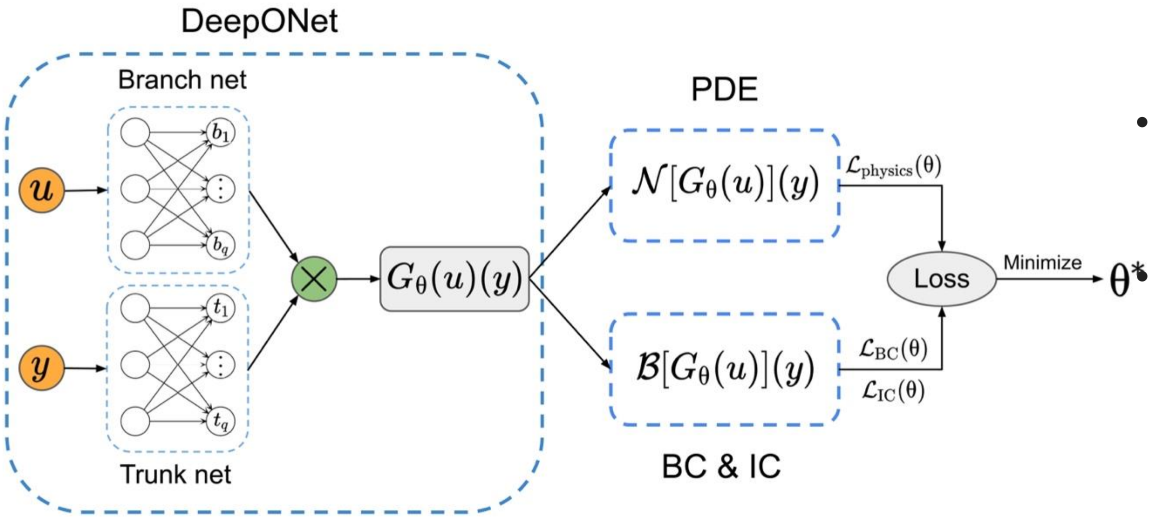
Can be FNN/CNN/U-Net/LSTM etc.



Minimize loss

$$G_\theta(u)(y) \downarrow \mathcal{L}_r(\theta) + \mathcal{L}_i(\theta) \downarrow \theta^*$$

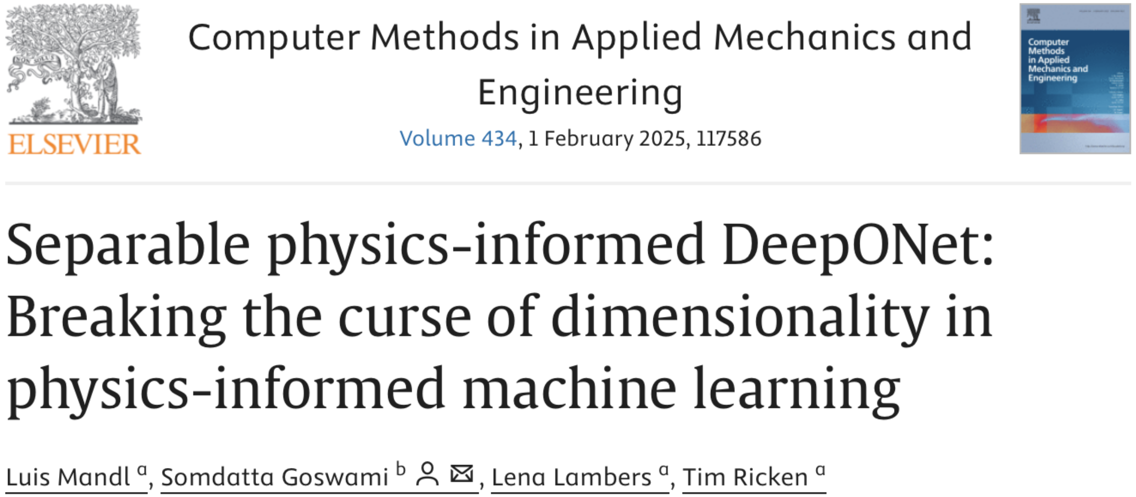
Physics-Informed DeepONet


DeepONet
Branch net
u
b_1
b_q
y
t_1
t_q
Trunk net
$G_\theta(u)(y)$
PDE
$\mathcal{N}[G_\theta(u)](y)$
$\mathcal{L}_{\text{physics}}(\theta)$
Loss
Minimize
θ^*
$\mathcal{B}[G_\theta(u)](y)$
$\mathcal{L}_{\text{BC}}(\theta)$
$\mathcal{L}_{\text{IC}}(\theta)$
BC & IC

- Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. Science Advances, 7(40), 2021.
- Somdatta Goswami, Yin, M., Yu, Y., & Karniadakis, G. E. (2022). A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials. Computer Methods in Applied Mechanics and Engineering, 391, 114587.

Extremely compute expensive; therefore, difficult to scale to high-dimensional systems.

Our Proposed Framework



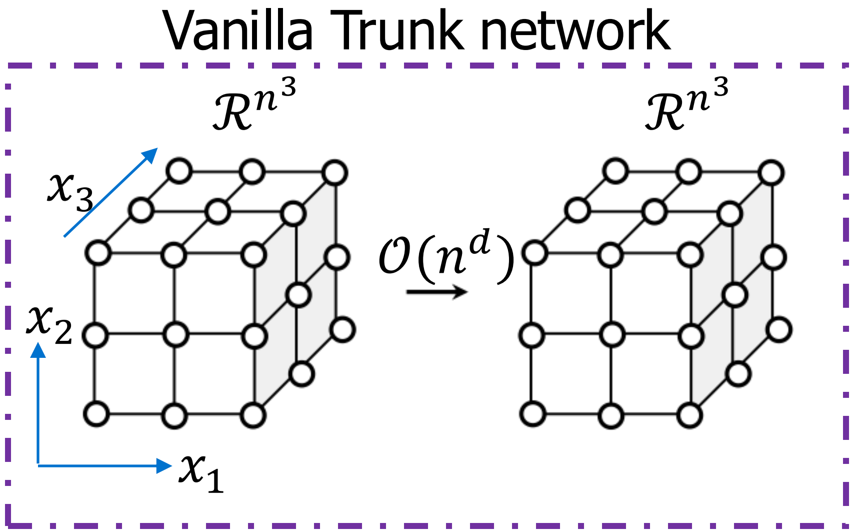
Computer Methods in Applied Mechanics and Engineering

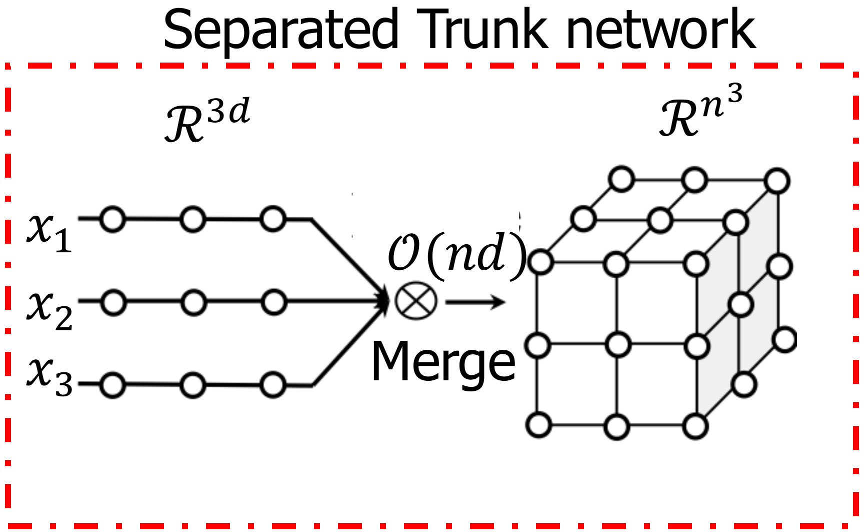
Volume 434, 1 February 2025, 117586

Separable physics-informed DeepONet: Breaking the curse of dimensionality in physics-informed machine learning

Luis Mandl [a], Somdatta Goswami [b], Lena Lambers [a], Tim Ricken [a]

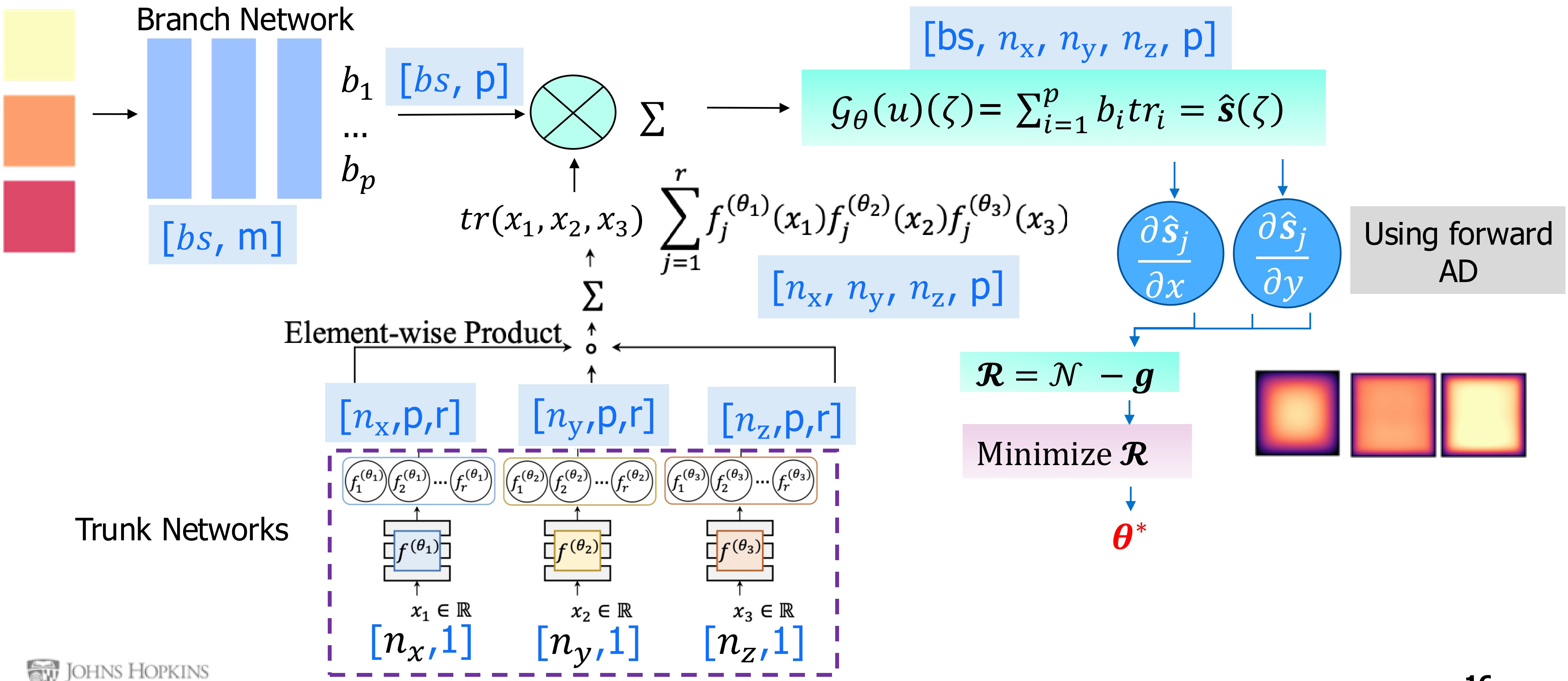
Introducing Separation of Variables





First Introduced in PINNs : Cho, J., Nam, S., Yang, H., Yun, S. B., Hong, Y., & Park, E. (2022). Separable PINN: Mitigating the curse of dimensionality in physics-informed neural networks. ArXiv preprint - 2211.08761.

Separable DeepONet Framework



Branch Network

$[bs, m]$

b_1 … b_p

$[bs, p]$

$\sum$

$[bs, n_x, n_y, n_z, p]$

$\mathcal{G}_\theta(u)(\zeta) = \sum_{i=1}^{p} b_i tr_i = \hat{\boldsymbol{s}}(\zeta)$

$tr(x_1, x_2, x_3) \sum_{j=1}^{r} f_j^{(\theta_1)}(x_1) f_j^{(\theta_2)}(x_2) f_j^{(\theta_3)}(x_3)$

$[n_x, n_y, n_z, p]$

$\frac{\partial \hat{\boldsymbol{s}}_j}{\partial x}$ $\frac{\partial \hat{\boldsymbol{s}}_j}{\partial y}$

Using forward AD

$\boldsymbol{\mathcal{R}} = \mathcal{N} - \boldsymbol{g}$

Minimize $\boldsymbol{\mathcal{R}}$

$\boldsymbol{\theta}^*$

$\sum$

Element-wise Product $\circ$

$[n_x,p,r]$ $[n_y,p,r]$ $[n_z,p,r]$

Trunk Networks

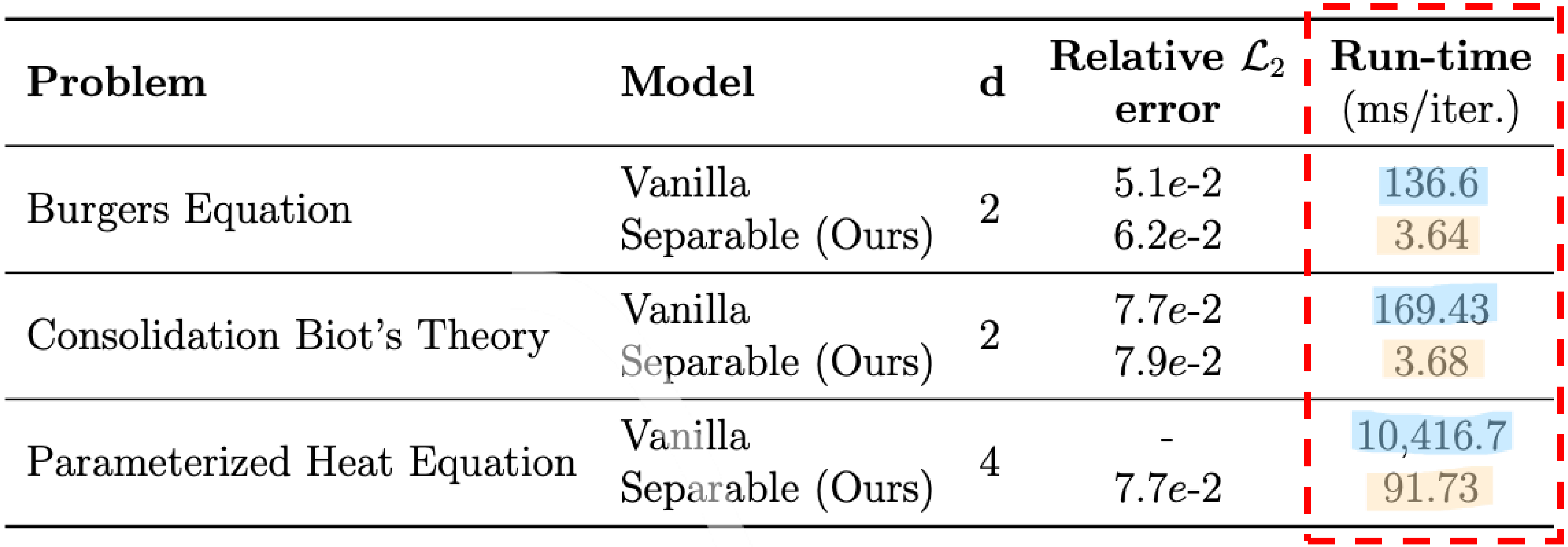
$f_1^{(\theta_1)} f_2^{(\theta_1)} \cdots f_r^{(\theta_1)}$ $f_1^{(\theta_2)} f_2^{(\theta_2)} \cdots f_r^{(\theta_2)}$ $f_1^{(\theta_3)} f_2^{(\theta_3)} \cdots f_r^{(\theta_3)}$

$f^{(\theta_1)}$ $f^{(\theta_2)}$ $f^{(\theta_3)}$

$x_1 \in \mathbb{R}$ $x_2 \in \mathbb{R}$ $x_3 \in \mathbb{R}$

$[n_x,1]$ $[n_y,1]$ $[n_z,1]$

Numerical Examples



Problem	Model	d	Relative $\mathcal{L}_2$ error	Run-time (ms/iter.)
Burgers Equation	Vanilla Separable (Ours)	2	$5.1e$-2 $6.2e$-2	136.6 3.64
Consolidation Biot's Theory	Vanilla Separable (Ours)	2	$7.7e$-2 $7.9e$-2	169.43 3.68
Parameterized Heat Equation	Vanilla Separable (Ours)	4	- $7.7e$-2	10,416.7 91.73

Separable Physics – Informed DeepONet reduces compute time by two orders, when compared to Vanilla Physics – Informed DeepONet .

Physics-Informed ML-integrated Numerical Simulators

Part – I: Separable Physics-Informed Neural Operators

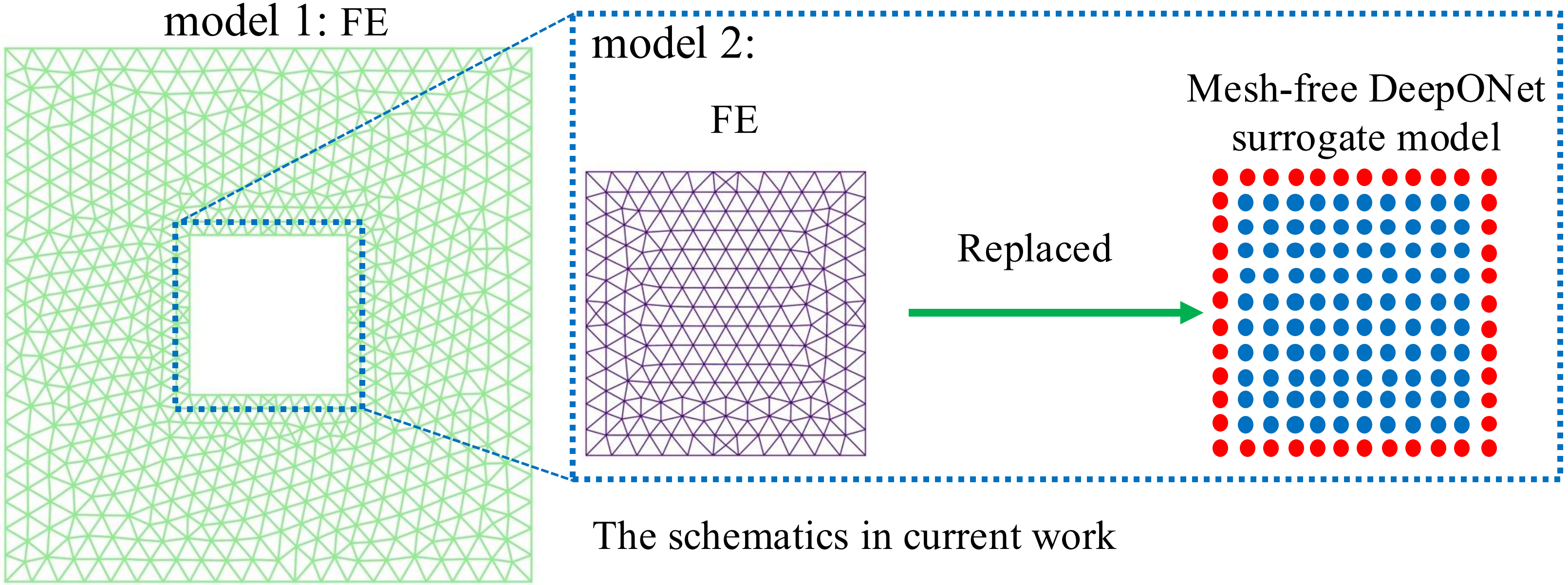
Part – II: Domain Decomposition Approaches for Integration

Our Proposed framework

Physics-Informed Neural Operator Coupled with Finite Element Methods: A High-Accuracy AI-Accelerated Framework for Time-Evolution Numerical Solutions (*Manuscript in Preparation*)

Domain Decomposition Framework

Decompose the computational domain; replace the computational expensive domain by PI-DeepONet.



model 1: FE

model 2:

FE

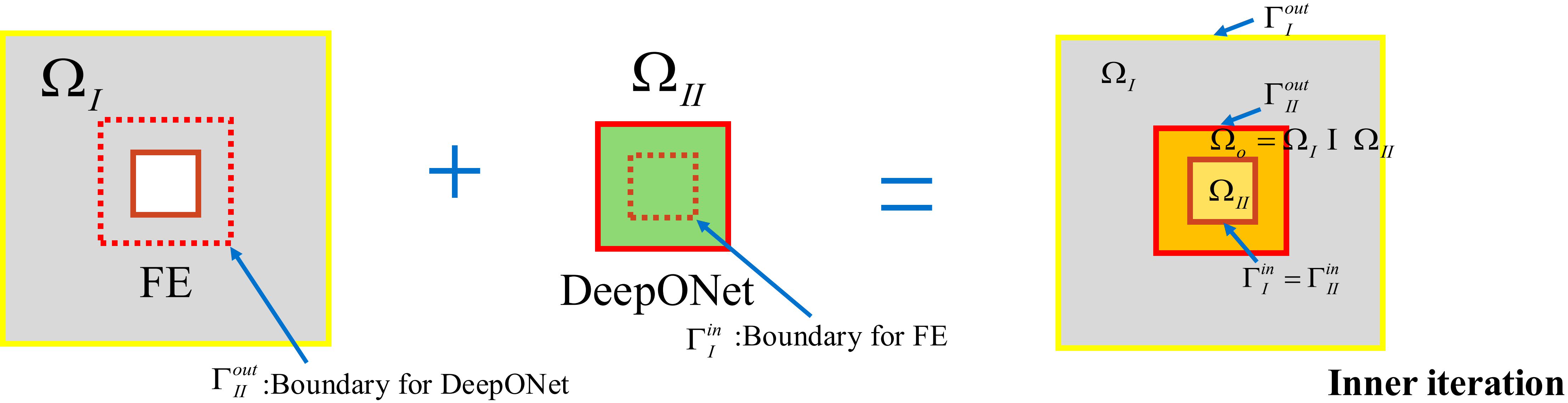
Replaced

Mesh-free DeepONet surrogate model

The schematics in current work

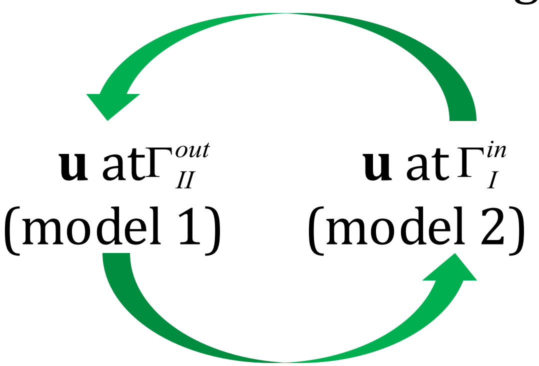
Establish the FE-DeepONet coupling framework → spatially and temporally couple the two solvers

Spatial Domain Coupling


Ω_I
FE
Γ_{II}^{out} :Boundary for DeepONet
+
Ω_{II}
DeepONet
Γ_I^{in} :Boundary for FE
=
Γ_I^{out}
Ω_I
Γ_{II}^{out}
$\Omega_o = \Omega_I$ I Ω_{II}
Ω_{II}
$\Gamma_I^{in} = \Gamma_{II}^{in}$
Inner iteration

Schwartz alternating method at overlapping boundary:

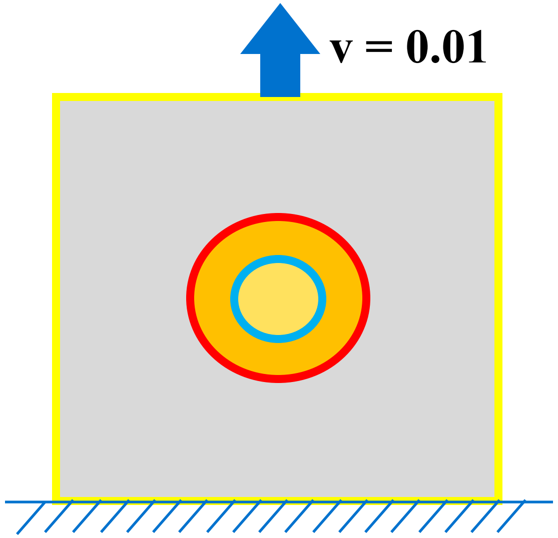
Information Exchange

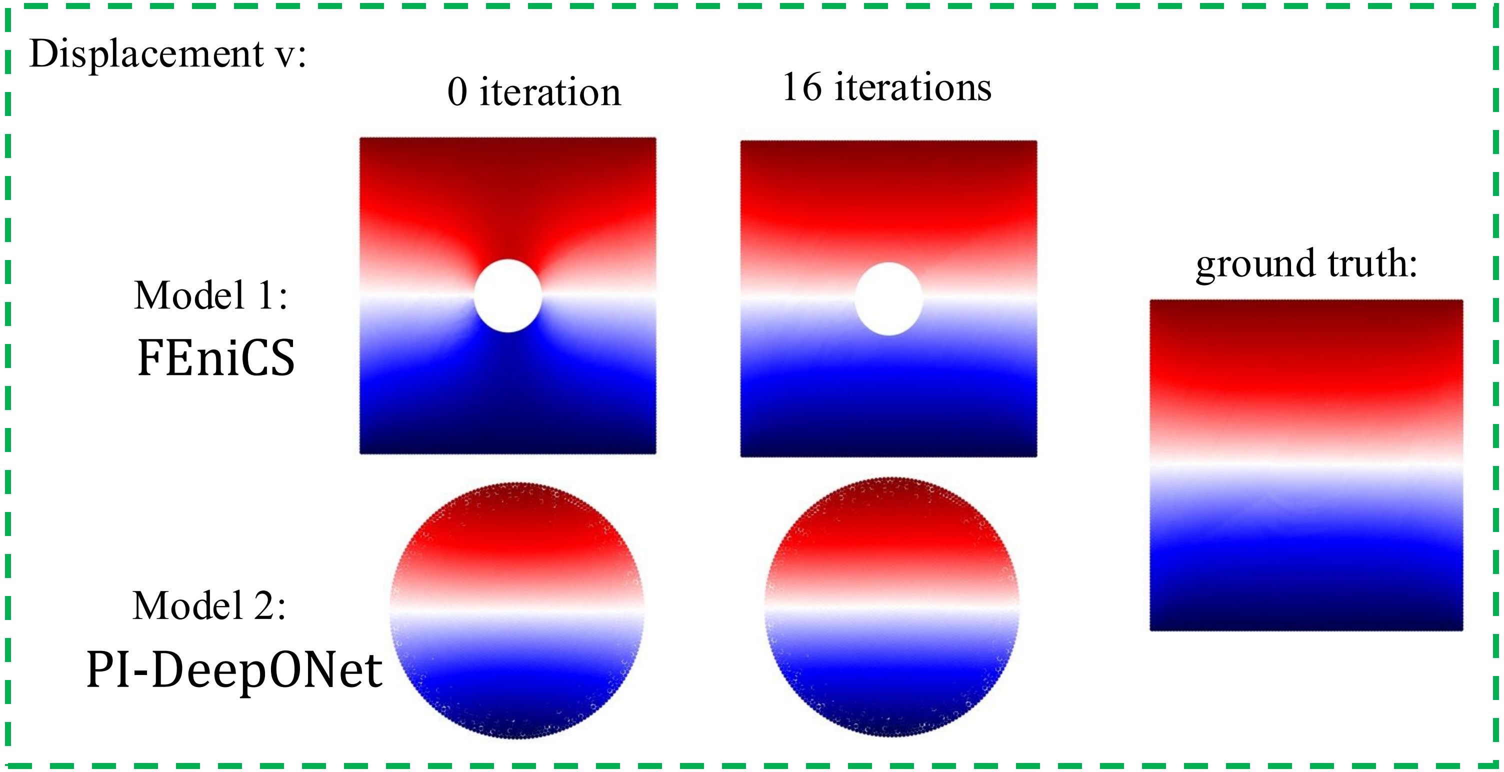


1. Receive the boundary conditions of Ω_I and obtain the displacement **u** at Γ_{II}^{out}, pass it to model 2 in Ω_{II}.
2. Receive the boundary conditions of Ω_{II} and obtain the displacement **u** at Γ_I^{in}, pass it to model 1 in Ω_I.
3. Obtain the results when the **u** difference from two successive iterations is smaller than the critical value.

Example: Linear Elastic Model in Static Regime

Static linear elastic example:

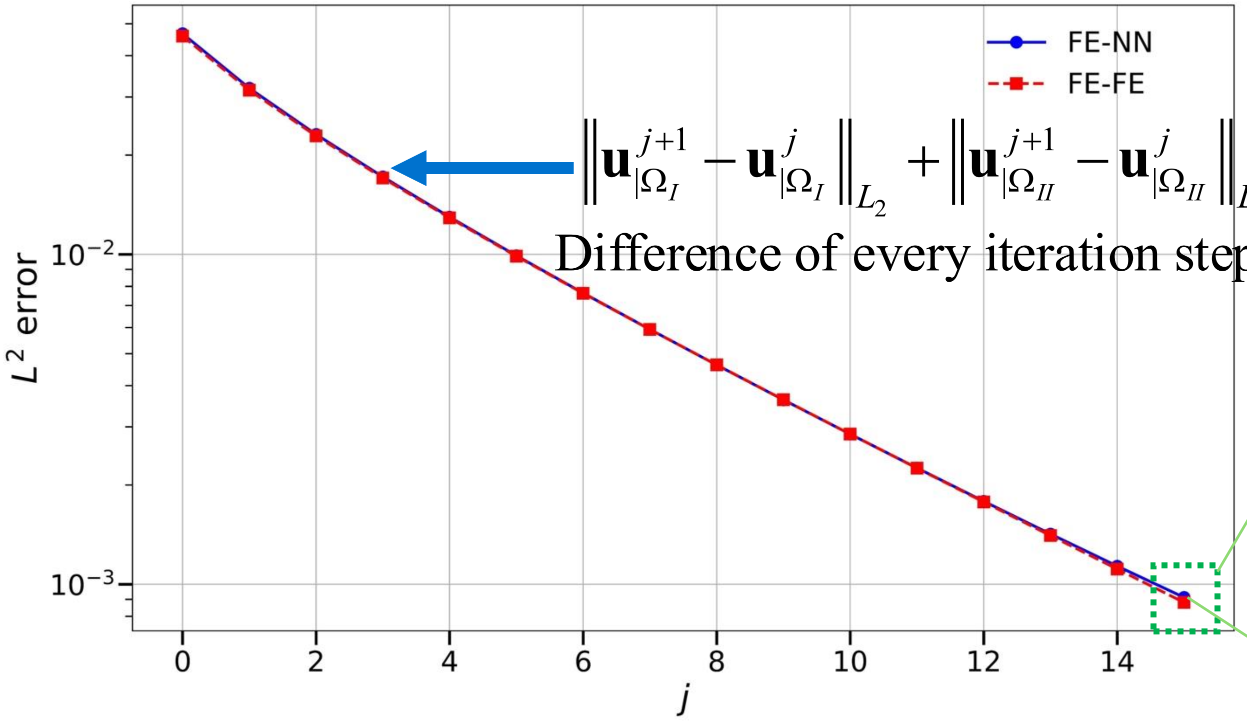




PI-DeepONet has been pre-trained with displacement boundary conditions generated as Gaussian Random Fields.

Accuracy Comparison with FE

L$_2$ error between two successive steps versus inner iterations steps

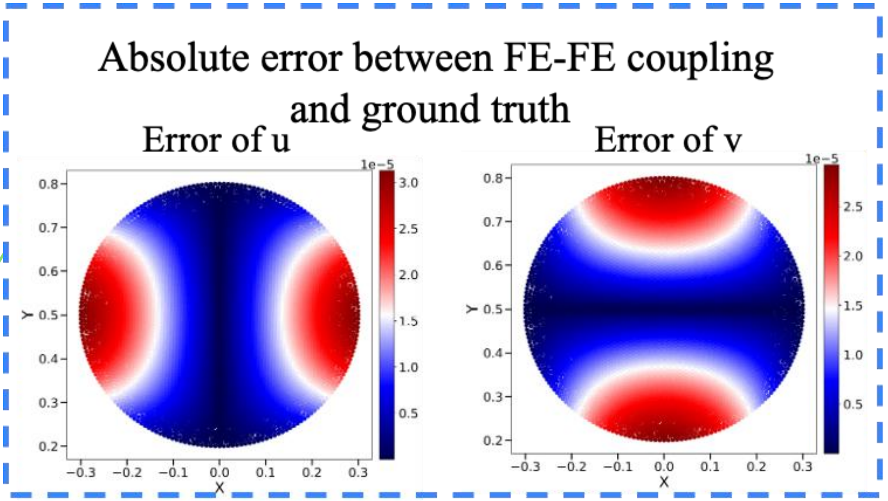


$$\left\|\mathbf{u}^{j+1}_{|\Omega_I} - \mathbf{u}^{j}_{|\Omega_I}\right\|_{L_2} + \left\|\mathbf{u}^{j+1}_{|\Omega_{II}} - \mathbf{u}^{j}_{|\Omega_{II}}\right\|_{L_2}$$

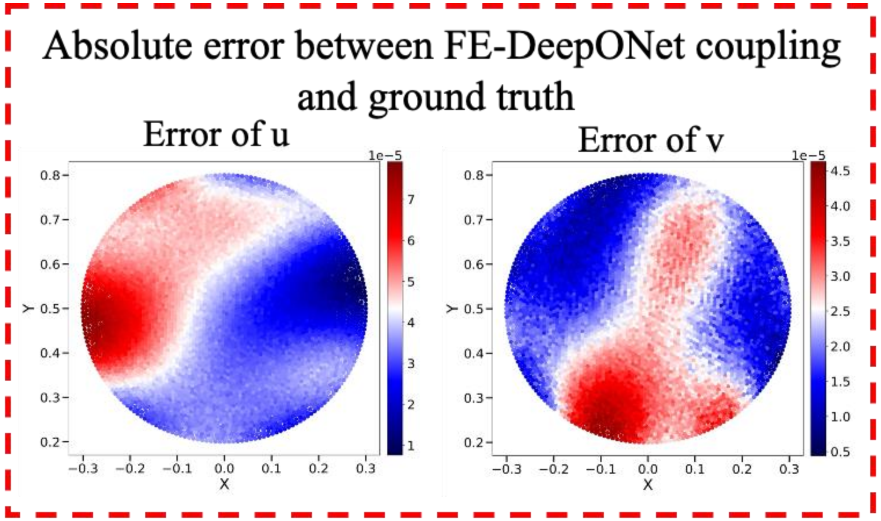
Difference of every iteration step

FE-DeepONet coupling is comparable with FE-FE coupling

Absolute error between FE-FE coupling and ground truth



Absolute error between FE-DeepONet coupling and ground truth



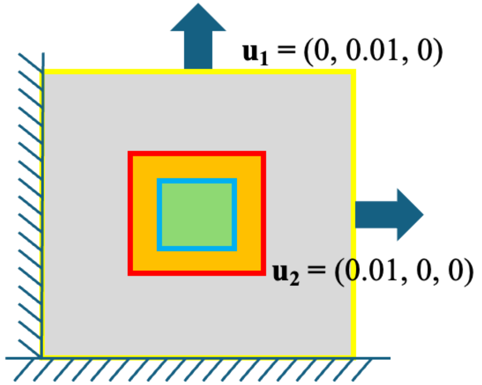
Results: Elastic plate subjected to Dynamic Loading

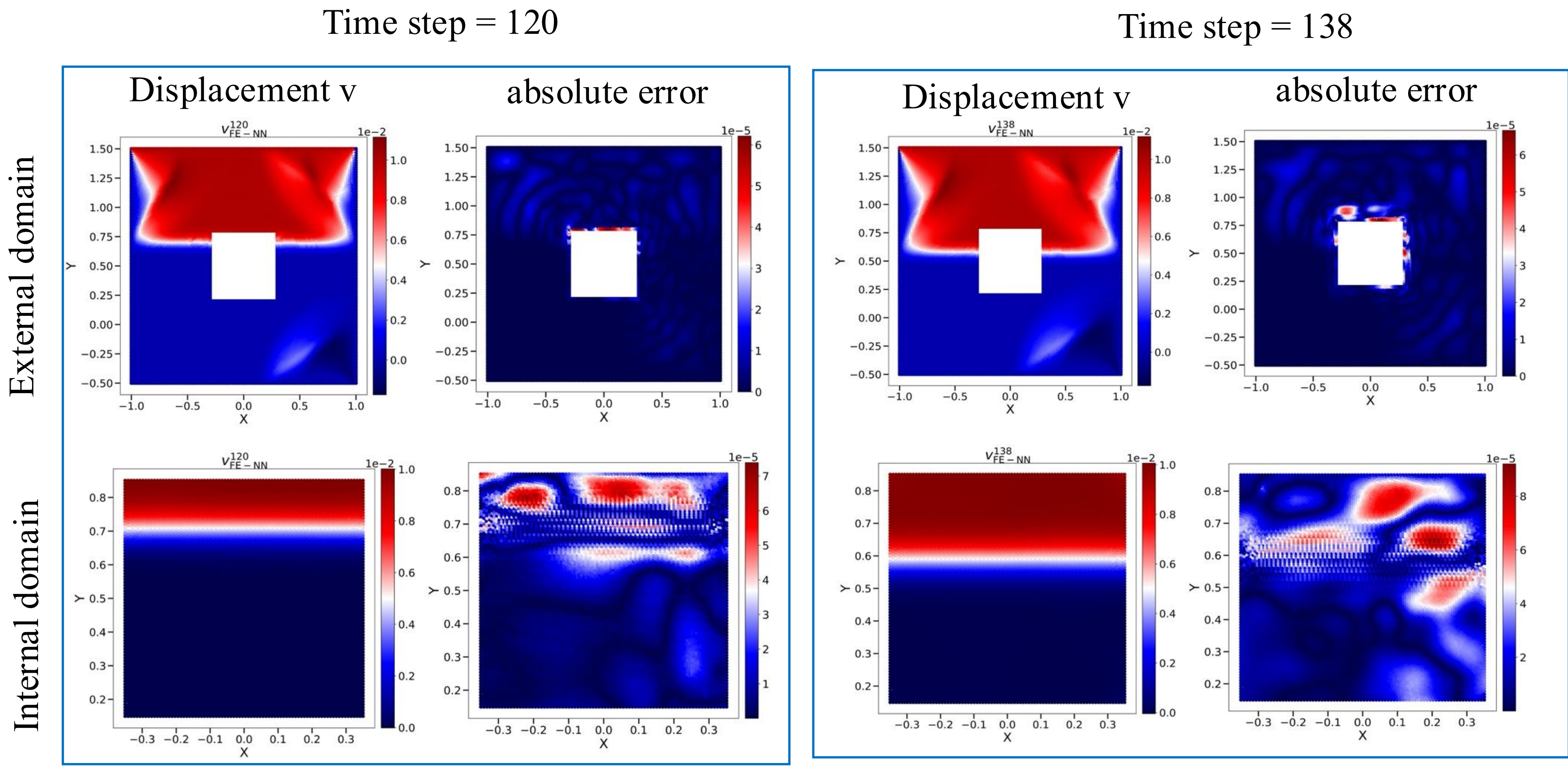
Newmark-beta time discretization method:

$$\dot{u}(t+dt)=\dot{u}+dt\left[(1-\gamma)\ddot{u}+\gamma\ddot{u}(t+dt)\right]$$

$$\ddot{u}(t+dt)=\frac{1}{(dt)^{2}\beta}\left[-u-\dot{u}\,dt+u(t+dt)\right]-\frac{(1-2\beta)}{2\beta}\ddot{u}$$

$$[M]\ddot{u}(t+dt)=[K]u(t+dt)+\left[F(t+dt)\right]$$




Time step = 120

Time step = 138

Displacement v

absolute error

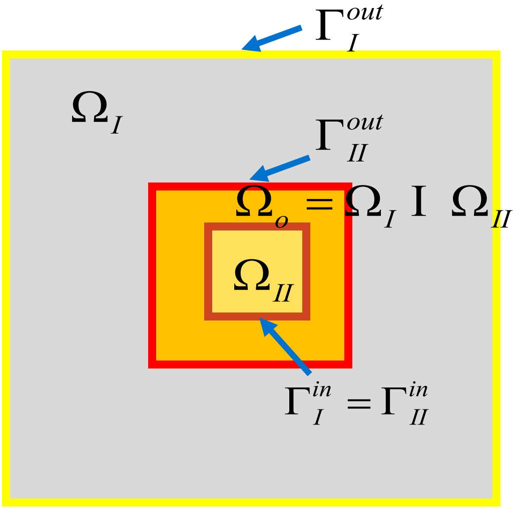
External domain

Internal domain

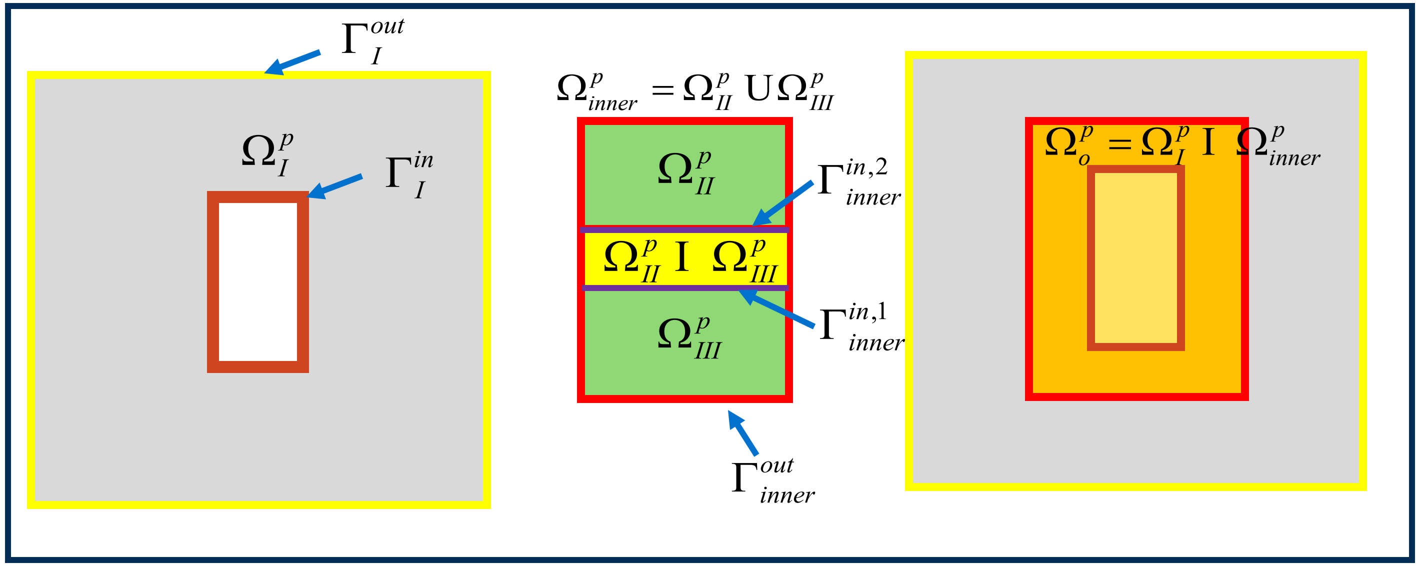
Now for a complex problem of realistic size, the subdomain employing the pre-trained solver has to expand as the solution evolves.

Adaptive extension of the ML –simulated subdomain

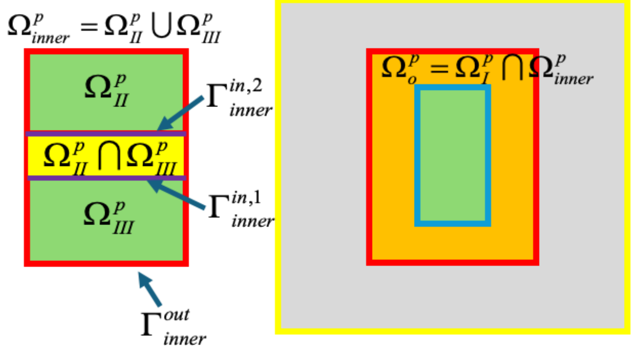
Initial decomposed domains


Γ_I^{out}
Ω_I
Γ_{II}^{out}
$\Omega_o = \Omega_I \; I \; \Omega_{II}$
Ω_{II}
$\Gamma_I^{in} = \Gamma_{II}^{in}$

Decomposed domain after a few timesteps


Γ_I^{out}
Ω_I^p
Γ_I^{in}
$\Omega_{inner}^p = \Omega_{II}^p \, U \Omega_{III}^p$
Ω_{II}^p
$\Gamma_{inner}^{in,2}$
$\Omega_{II}^p \; I \; \Omega_{III}^p$
$\Gamma_{inner}^{in,1}$
Ω_{III}^p
Γ_{inner}^{out}
$\Omega_o^p = \Omega_I^p \; I \; \Omega_{inner}^p$

Dynamic expansion of the ML simulated subdomain



$\Omega^p_{inner} = \Omega^p_{II} \cup \Omega^p_{III}$

Ω^p_{II}

$\Omega^p_{II} \cap \Omega^p_{III}$

Ω^p_{III}

$\Gamma^{in,2}_{inner}$

$\Gamma^{in,1}_{inner}$

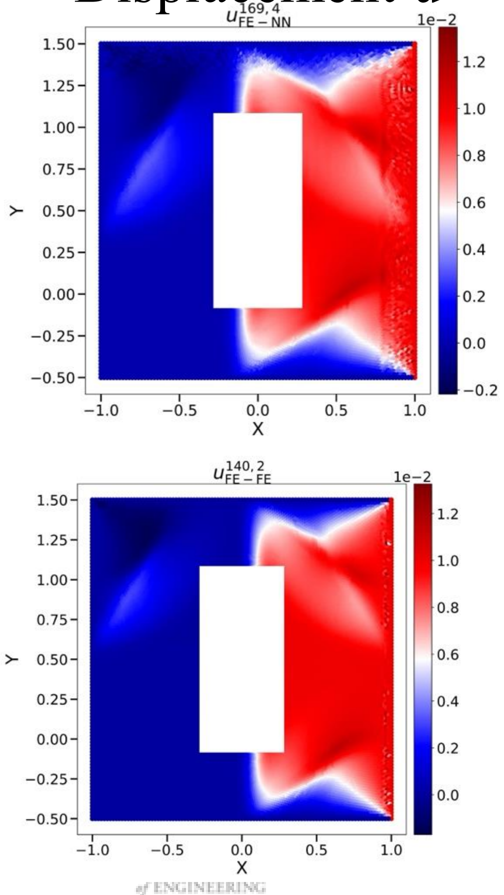
Γ^{out}_{inner}

$\Omega^p_o = \Omega^p_I \cap \Omega^p_{inner}$

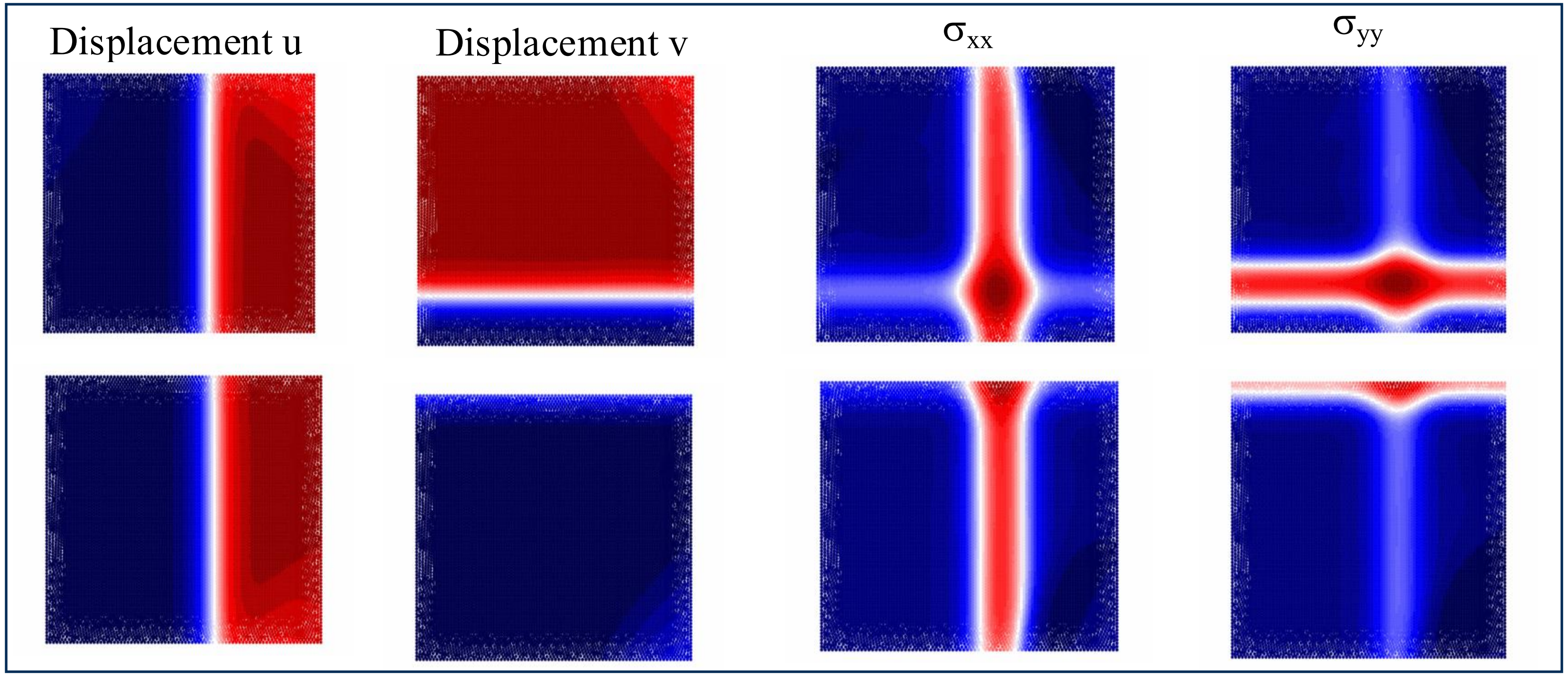
Outer Domain
Displacement u

FE-NN framework

FE-FE framework



Inner Domain



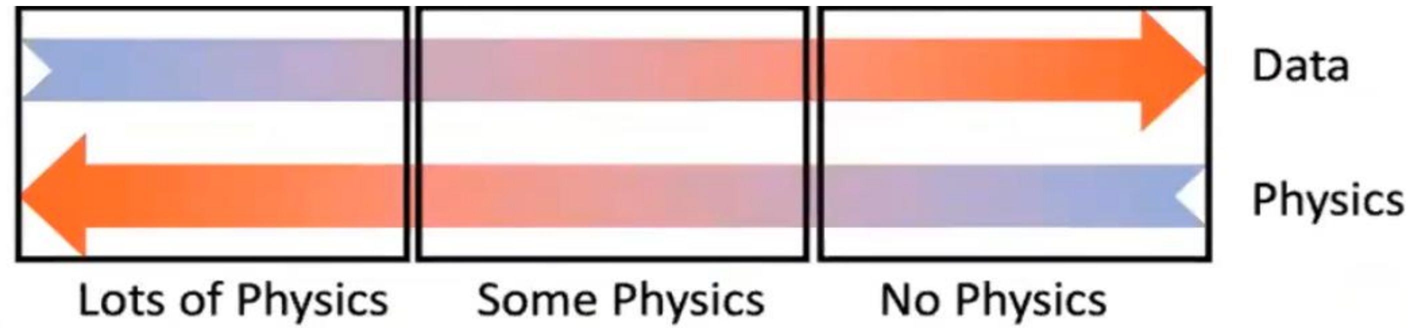
Displacement u

Displacement v

σ_{xx}

σ_{yy}

Key Takeaways



- These methods have a niche in real world problems, where partially physics in known and some measurements of quantities of interest are available.
- Separable architecture introduces the possibility of employing physics in neural operators.
- Learning the system by combining NOs with numerical simulators up the possibility of multiscale modeling and accelerated computational time with reliable predictions.

Acknowledgement

Funding

Thank you!