A Unified Hard-Constraint Framework for Solving Geometrically Complex PDEs

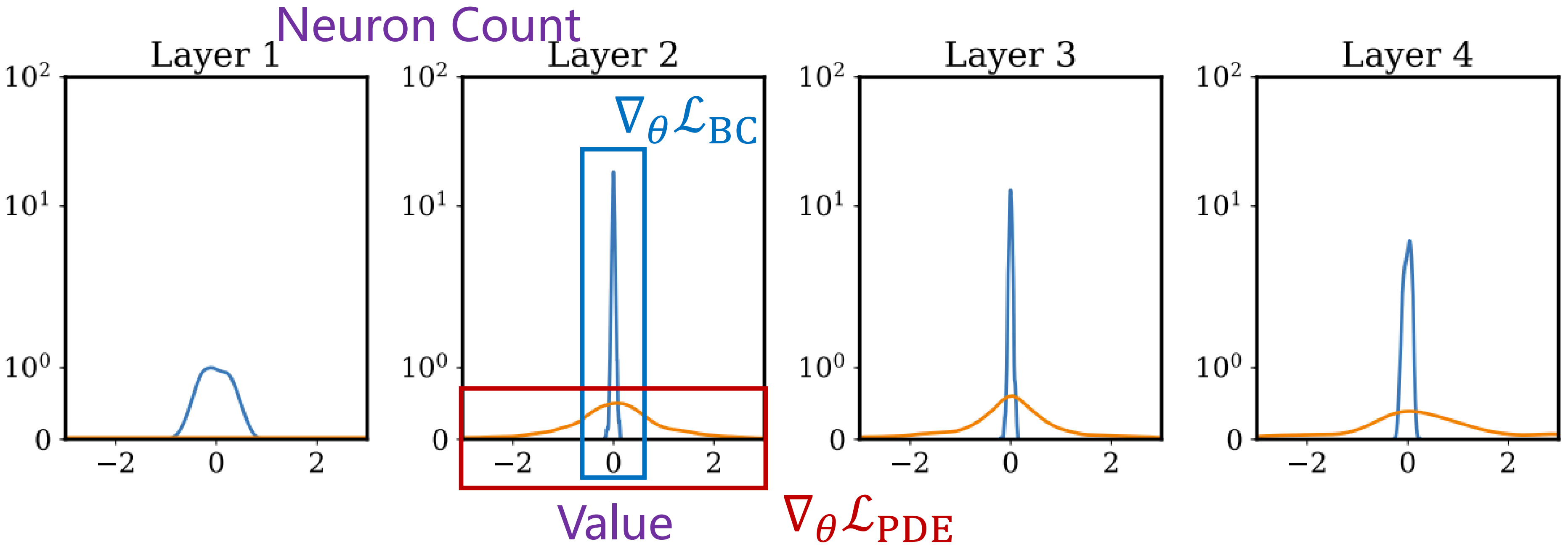
Songming Liu, Zhongkai Hao, Chengyang Ying,
Hang Su, Jun Zhu, Ze Cheng

Background

Solving PDEs via Neural Networks

- **PINNs → PDEs with Boundary Conditions (BCs)**

$$\hat{u} \coloneqq \mathrm{NN}(\,\cdot\,;\boldsymbol{\theta}), \qquad \mathcal{L}(\boldsymbol{\theta}) \coloneqq \mathcal{L}_{\mathrm{PDE}}(\boldsymbol{\theta}) + \mathcal{L}_{\mathrm{BC}}(\boldsymbol{\theta})$$

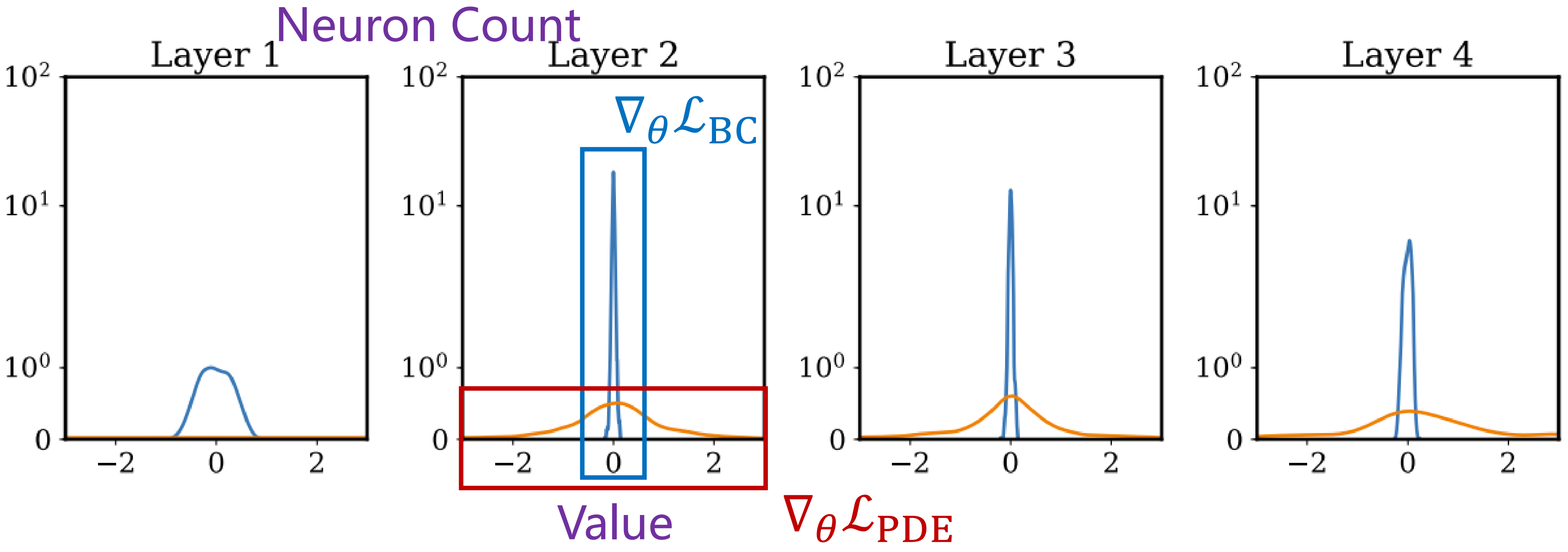


Unbalanced Competition[1]: $|\nabla_\theta \mathcal{L}_{\mathrm{PDE}}| \gg |\nabla_\theta \mathcal{L}_{\mathrm{BC}}|$!

Solving PDEs via Neural Networks

- **PINNs → PDEs with Boundary Conditions (BCs)**

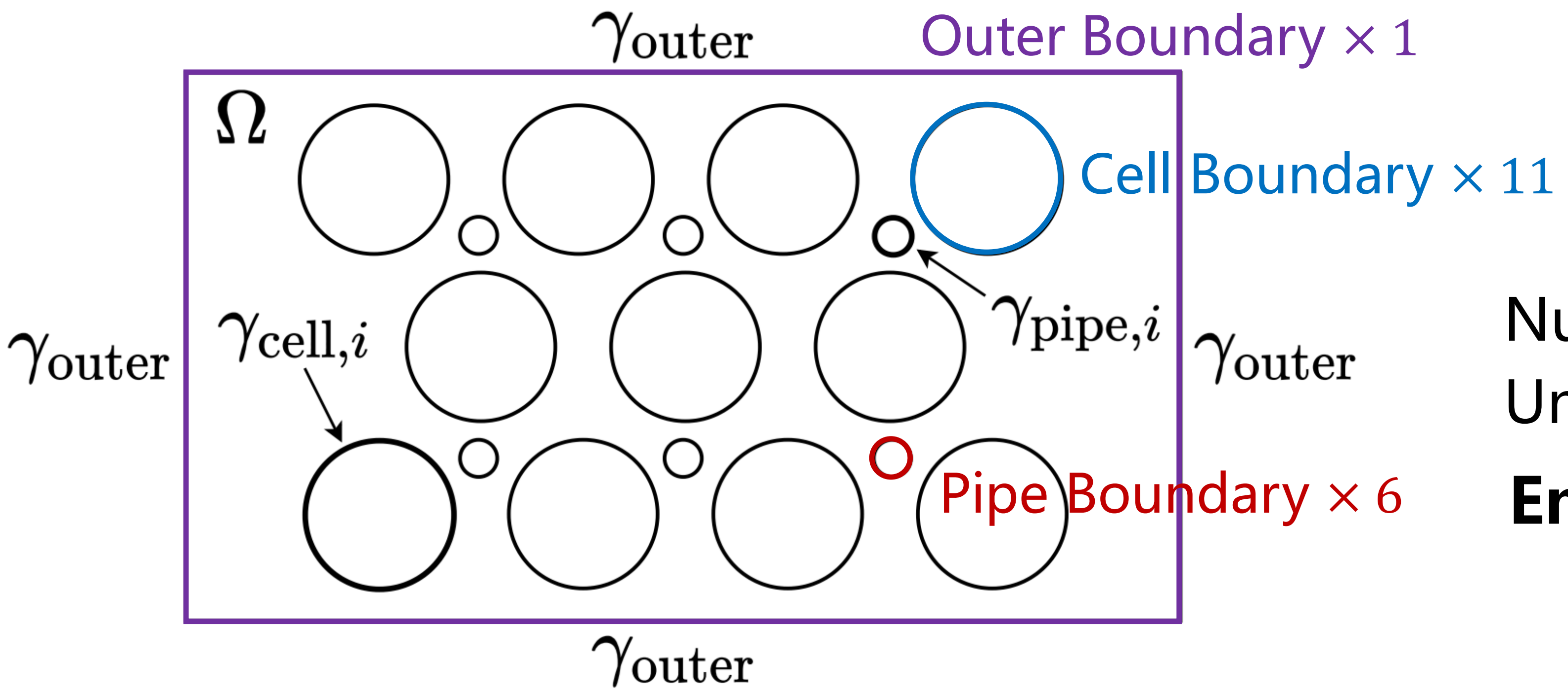
$$\hat{u} := \mathrm{NN}(\,\cdot\,;\boldsymbol{\theta}), \qquad \mathcal{L}(\boldsymbol{\theta}) := \mathcal{L}_{\mathrm{PDE}}(\boldsymbol{\theta}) + \mathcal{L}_{\mathrm{BC}}(\boldsymbol{\theta})$$



Unbalanced Competition[1]: $|\nabla_\theta \mathcal{L}_{\mathrm{PDE}}| \gg |\nabla_\theta \mathcal{L}_{\mathrm{BC}}|$!

Geometrically Complex PDEs

- **Example:** a 2D battery pack



Number of BCs ↑

Unbalance Competition↑

Embed BCs into Ansatz!

Previous Work

Methods	Neumann BCs	Robin BCs	Complex Geometries	High Dimension	Meshless
Variational PINNs[2]	✗*	✗	✗	✓	✓
Hard-Constraint PINNs[3]	✗	✗	✗	✓	✓
Deep TFCs[4]	✓	✓	✗	✗	✓
Hard-Constraint CNNs[5]	✓	✓	✓	✗	✗
Our Proposed Method	✓	✓	✓	✓	✓

*Only applicable to homogeneous Neumann BCs

Method

Problem Formulation

- **PDE:**

$$\mathcal{F}[u(\boldsymbol{x})] = 0, \qquad \boldsymbol{x} = (x_1, \ldots, x_d) \in \Omega$$

with Dirichlet, Neumann, Robin BCs:

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \nabla u) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i, \qquad i = 1, \ldots, m,$$

where $\cup_{i=1}^{m} \gamma_i = \partial\Omega$, $\boldsymbol{n}$ is the (outward facing) unit normal.

- **Extra Fields:** $\boldsymbol{p} = \nabla u, \quad x \in \Omega \cup \partial\Omega$

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \boldsymbol{p}) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i, \qquad i = 1, \ldots, m.$$

- A **linear equation** w.r.t. $(u, \boldsymbol{p})$!

Problem Formulation

- **PDE:**

$$\mathcal{F}[u(\boldsymbol{x})] = 0, \qquad \boldsymbol{x} = (x_1, \dots, x_d) \in \Omega$$

with Dirichlet, Neumann, Robin BCs:

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \nabla u) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i, \qquad i = 1, \dots, m,$$

where $\cup_{i=1}^{m} \gamma_i = \partial\Omega$, $\boldsymbol{n}$ is the (outward facing) unit normal.

- **Extra Fields:** $\boldsymbol{p} = \nabla u, \quad x \in \Omega \cup \partial\Omega$

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \boldsymbol{p}) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i, \qquad i = 1, \dots, m.$$

- A **linear equation** w.r.t. $(u, \boldsymbol{p})$!

Problem Formulation

- **PDE:**

$$\mathcal{F}[u(\boldsymbol{x})] = 0, \qquad \boldsymbol{x} = (x_1, \dots, x_d) \in \Omega$$

with Dirichlet, Neumann, Robin BCs:

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \nabla u) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i, \qquad i = 1, \dots, m,$$

where $\cup_{i=1}^{m} \gamma_i = \partial\Omega$, $\boldsymbol{n}$ is the (outward facing) unit normal.

- **Extra Fields:** $\boldsymbol{p} = \nabla u, \quad x \in \Omega \cup \partial\Omega$

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \boldsymbol{p}) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i, \qquad i = 1, \dots, m.$$

- A **linear equation** w.r.t. $(u, \boldsymbol{p})$!

Problem Formulation

- **PDE:**

$$\mathcal{F}[u(\boldsymbol{x})] = 0, \qquad \boldsymbol{x} = (x_1, \dots, x_d) \in \Omega$$

with Dirichlet, Neumann, Robin BCs:

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \nabla u) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i, \qquad i = 1, \dots, m,$$

where $\cup_{i=1}^{m} \gamma_i = \partial\Omega$, $\boldsymbol{n}$ is the (outward facing) unit normal.

- **Extra Fields:** $\boldsymbol{p} = \nabla u, \quad x \in \Omega \cup \partial\Omega$

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \boldsymbol{p}) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i, \qquad i = 1, \dots, m.$$

- A **linear equation** w.r.t. $(u, \boldsymbol{p})$!

General Solution at Boundaries

- **Reformulated BC:**

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \boldsymbol{p}) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i,$$

- **The General Solutions Go With:**

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}) = \boldsymbol{B}(\boldsymbol{x})\mathrm{NN}^{\gamma_i}(\boldsymbol{x}; \boldsymbol{\theta}^{\gamma_i}) + \tilde{\boldsymbol{n}}(\boldsymbol{x})\tilde{g}_i(\boldsymbol{x}),$$

where $\tilde{\boldsymbol{n}} = (a_i, b_i\boldsymbol{n})/\left(a_i^2 + b_i^2\right)^{1/2}$, $\tilde{g}_i = g_i/\left(a_i^2 + b_i^2\right)^{1/2}$,
$\mathrm{NN}^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}^{d+1}$, $\boldsymbol{B}(\boldsymbol{x}) = \boldsymbol{I}_{d+1} - \tilde{\boldsymbol{n}}(\boldsymbol{x})\tilde{\boldsymbol{n}}(\boldsymbol{x})^{\top}$.

- It strictly **satisfies BC** and retains **expressive power**.

General Solution at Boundaries

- **Reformulated BC:**

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \boldsymbol{p}) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i,$$

- **The General Solutions Go With:**

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}) = \boldsymbol{B}(\boldsymbol{x})\mathrm{NN}^{\gamma_i}(\boldsymbol{x}; \boldsymbol{\theta}^{\gamma_i}) + \tilde{\boldsymbol{n}}(\boldsymbol{x})\tilde{g}_i(\boldsymbol{x}),$$

where $\tilde{\boldsymbol{n}} = (a_i, b_i\boldsymbol{n})/\left(a_i^2 + b_i^2\right)^{1/2}$, $\tilde{g}_i = g_i/\left(a_i^2 + b_i^2\right)^{1/2}$,
$\mathrm{NN}^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}^{d+1}$, $\boldsymbol{B}(\boldsymbol{x}) = \boldsymbol{I}_{d+1} - \tilde{\boldsymbol{n}}(\boldsymbol{x})\tilde{\boldsymbol{n}}(\boldsymbol{x})^\top$.

- It strictly **satisfies BC** and retains **expressive power**.

General Solution at Boundaries

- **Reformulated BC:**

$$a_i(\boldsymbol{x})u + b_i(\boldsymbol{x})(\boldsymbol{n}(\boldsymbol{x}) \cdot \boldsymbol{p}) = g_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \gamma_i,$$

- **The General Solutions Go With:**

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}) = \boldsymbol{B}(\boldsymbol{x})\mathrm{NN}^{\gamma_i}(\boldsymbol{x}; \boldsymbol{\theta}^{\gamma_i}) + \tilde{\boldsymbol{n}}(\boldsymbol{x})\tilde{g}_i(\boldsymbol{x}),$$

where $\tilde{\boldsymbol{n}} = (a_i, b_i\boldsymbol{n})/\left(a_i^2 + b_i^2\right)^{1/2}$, $\tilde{g}_i = g_i/\left(a_i^2 + b_i^2\right)^{1/2}$,
$\mathrm{NN}^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}^{d+1}$, $\boldsymbol{B}(\boldsymbol{x}) = \boldsymbol{I}_{d+1} - \tilde{\boldsymbol{n}}(\boldsymbol{x})\tilde{\boldsymbol{n}}(\boldsymbol{x})^\top$.

- It strictly **satisfies BC** and retains **expressive power**.

A Unified Framework

- Gather the general solution at γ_i

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}), \qquad i = 1, \dots, m$$

- **Our Final Ansatz:**

$$(\hat{u}, \widehat{\boldsymbol{p}}) = l^{\partial\Omega}(\boldsymbol{x})\mathrm{NN}(\boldsymbol{x}, \boldsymbol{\theta}) + \sum_{i=1}^{m} \exp[-\alpha_i l^{\gamma_i}(\boldsymbol{x})](u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$$

where $l^{\partial\Omega}, l^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}$ are the distance function to $\partial\Omega$, γ_i, $\mathrm{NN}: \mathbb{R}^d \to \mathbb{R}^{d+1}$.

- **Automatically satisfy BCs:** $x \in \gamma_i \Rightarrow (\hat{u}, \widehat{\boldsymbol{p}}) = (u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$!

A Unified Framework

- Gather the general solution at γ_i

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}), \qquad i = 1, \dots, m$$

- **Our Final Ansatz:**

$$(\hat{u}, \hat{\boldsymbol{p}}) = l^{\partial\Omega}(\boldsymbol{x})\mathrm{NN}(\boldsymbol{x}, \boldsymbol{\theta}) + \sum_{i=1}^{m} \exp[-\alpha_i l^{\gamma_i}(\boldsymbol{x})](u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$$

where $l^{\partial\Omega}, l^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}$ are the distance function to $\partial\Omega$, γ_i, $\mathrm{NN}: \mathbb{R}^d \to \mathbb{R}^{d+1}$.

- **Automatically satisfy BCs:** $x \in \gamma_i \Rightarrow (\hat{u}, \hat{\boldsymbol{p}}) = (u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$!

A Unified Framework

- Gather the general solution at γ_i

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}), \qquad i = 1, \dots, m$$

- **Our Final Ansatz:**

Vanishes at $\partial\Omega$ — Vanishes at $\partial\Omega \setminus \gamma_i$

$$(\hat{u}, \hat{\boldsymbol{p}}) = \boxed{l^{\partial\Omega}(\boldsymbol{x})}\mathrm{NN}(\boldsymbol{x}, \boldsymbol{\theta}) + \sum_{i=1}^{m} \boxed{\exp[-\alpha_i l^{\gamma_i}(\boldsymbol{x})]}(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$$

where $l^{\partial\Omega}, l^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}$ are the distance function to $\partial\Omega$, γ_i, $\mathrm{NN}: \mathbb{R}^d \to \mathbb{R}^{d+1}$.

- **Automatically satisfy BCs:** $x \in \gamma_i \Rightarrow (\hat{u}, \hat{\boldsymbol{p}}) = (u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$!

A Unified Framework

- Gather the general solution at γ_i

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}), \qquad i = 1, \dots, m$$

- **Our Final Ansatz:**

$$(\hat{u}, \hat{\boldsymbol{p}}) = l^{\partial\Omega}(\boldsymbol{x})\boxed{\mathrm{NN}(\boldsymbol{x}, \boldsymbol{\theta})} + \sum_{i=1}^{m} \exp[-\alpha_i l^{\gamma_i}(\boldsymbol{x})](u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$$

Freedom Term

where $l^{\partial\Omega}, l^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}$ are the distance function to $\partial\Omega$, γ_i, $\mathrm{NN}: \mathbb{R}^d \to \mathbb{R}^{d+1}$.

- **Automatically satisfy BCs:** $x \in \gamma_i \Rightarrow (\hat{u}, \hat{\boldsymbol{p}}) = (u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$!

A Unified Framework

- Gather the general solution at γ_i

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}), \qquad i = 1, \dots, m$$

- **Our Final Ansatz:**

$$(\hat{u}, \hat{\boldsymbol{p}}) = l^{\partial\Omega}(\boldsymbol{x})\mathrm{NN}(\boldsymbol{x}, \boldsymbol{\theta}) + \sum_{i=1}^{m} \exp[-\alpha_i l^{\gamma_i}(\boldsymbol{x})]\boxed{(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})}$$

General Solution at γ_i

where $l^{\partial\Omega}, l^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}$ are the distance function to $\partial\Omega$, γ_i, $\mathrm{NN}: \mathbb{R}^d \to \mathbb{R}^{d+1}$.

- **Automatically satisfy BCs:** $x \in \gamma_i \Rightarrow (\hat{u}, \hat{\boldsymbol{p}}) = (u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$!

A Unified Framework

- Gather the general solution at γ_i

$$(u^{\gamma_i}, \boldsymbol{p}^{\gamma_i}), \qquad i = 1, \ldots, m$$

- **Our Final Ansatz:**

Vanishes at $\partial\Omega$ — Vanishes at $\partial\Omega \setminus \gamma_i$

$$(\hat{u}, \hat{\boldsymbol{p}}) = l^{\partial\Omega}(\boldsymbol{x}) \mathrm{NN}(\boldsymbol{x}, \boldsymbol{\theta}) + \sum_{i=1}^{m} \exp[-\alpha_i l^{\gamma_i}(\boldsymbol{x})] (u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$$

Freedom Term — General Solution at γ_i

where $l^{\partial\Omega}, l^{\gamma_i}: \mathbb{R}^d \to \mathbb{R}$ are the distance function to $\partial\Omega$, γ_i, $\mathrm{NN}: \mathbb{R}^d \to \mathbb{R}^{d+1}$.

- **Automatically satisfy BCs:** $\boldsymbol{x} \in \gamma_i \Rightarrow (\hat{u}, \hat{\boldsymbol{p}}) = (u^{\gamma_i}, \boldsymbol{p}^{\gamma_i})$!

Experiments

Experimental Results

- **Evaluation Metrics:**
  - Mean Absolute Error (**MAE**), Mean Absolute Percentage Error (**MAPE**)
  - Weighted Mean Absolute Percentage Error (**WMAPE**):

$$\text{WMAPE} = \Sigma|\text{error}_i| \,/\, \Sigma|\text{truth}_i|$$

- **Baselines:**
  - **PINN**[6]**:** vanilla PINN
  - **PINN-LA**[7] **& PINN-LA-2:** PINN with learning rate annealing
  - **xPINN**[8] **& FBPINN**[9]**:** PINN with domain decomposition for geometrically complex PDEs
  - **PFNN**[10] **& PFNN-2**[11]**:** hard-constraint methods based on the variational formulation of PDEs

Experimental Results

- **Evaluation Metrics:**
  - Mean Absolute Error (**MAE**), Mean Absolute Percentage Error (**MAPE**)
  - Weighted Mean Absolute Percentage Error (**WMAPE**):

$$\mathrm{WMAPE} = \Sigma|\mathrm{error}_i| \,/\, \Sigma|\mathrm{truth}_i|$$

- **Baselines:**
  - **PINN**[6]**:** vanilla PINN
  - **PINN-LA**[7] **& PINN-LA-2:** PINN with learning rate annealing
  - **xPINN**[8] **& FBPINN**[9]**:** PINN with domain decomposition for geometrically complex PDEs
  - **PFNN**[10] **& PFNN-2**[11]**:** hard-constraint methods based on the variational formulation of PDEs

Experimental Results

Result:

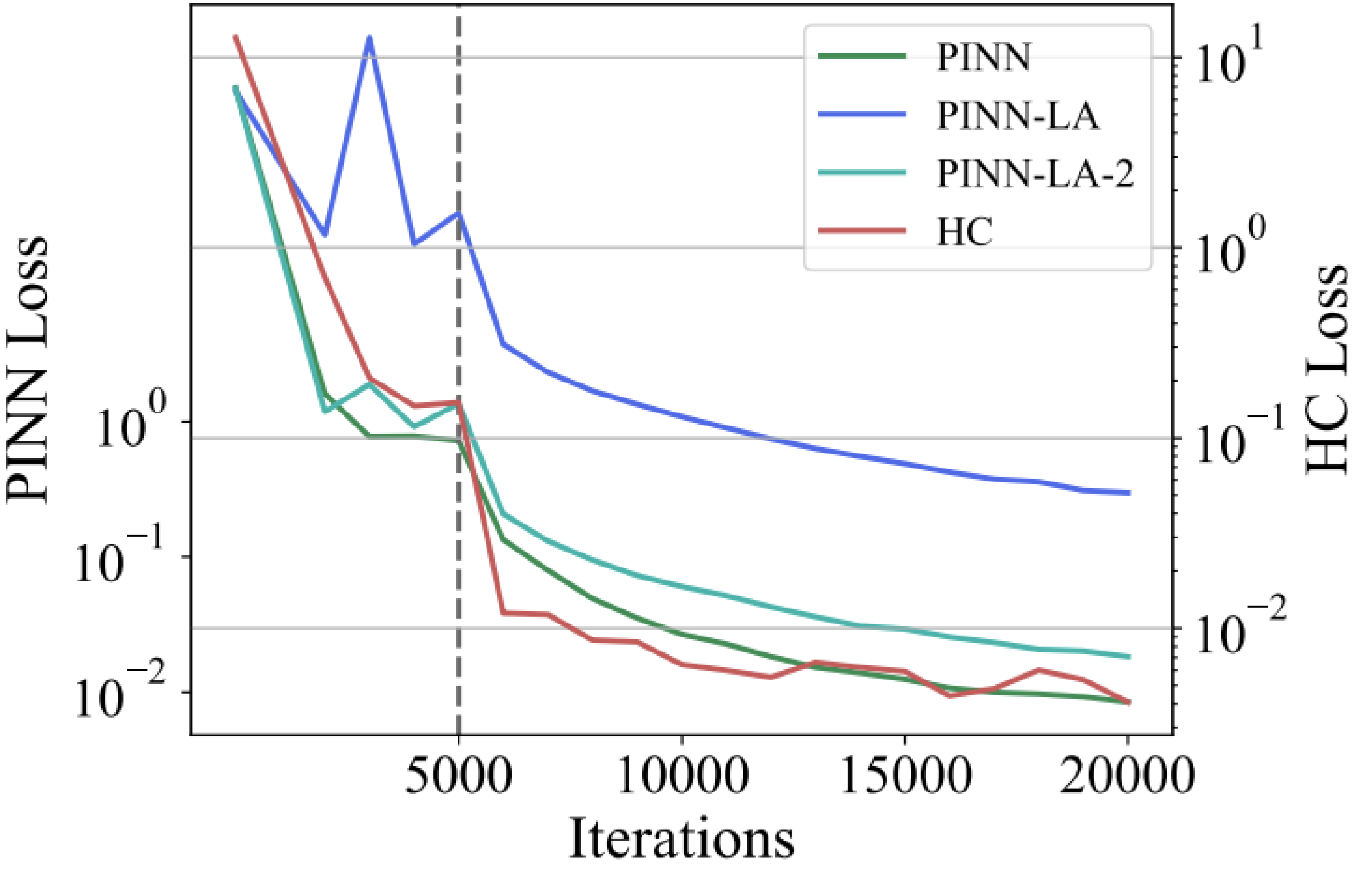
- Error $\downarrow > 70\%$
- Convergence $\uparrow$

Methods	Real-world Problems					
	2D Battery Pack		Airfoil (u_1)		High Dimension	
	MAE	MAPE	MAE	WMAPE	MAE	MAPE
PINN	0.0539	24.82%	0.4682	0.5924	0.0582	1.99%
PINN-LA	0.0661	27.06%	0.4018	0.5084	0.0235	0.78%
PINN-LA-2	0.0402	19.76%	0.5047	0.6385	0.0466	1.49%
FBPINN	0.0343	14.74%	0.4058	0.5134	-	-
xPINN	0.1454	54.70%	0.7188	0.9095	-	-
PFNN	0.2758	68.29%	-	-	0.1425	4.64%
PFNN-2	0.3215	59.62%	-	-	-	-
HC (**Ours**)	**0.0221**	**5.10%**	**0.2689**	**0.3402**	**0.0026**	**0.10%**

Experimental Results

Result:

- Error ↓ > 70%
- Convergence ↑



References

[1] Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM Journal on Scientific Computing, 43(5):A3055–A3081, 2021.

[2] Kharazmi, E., Zhang, Z., & Karniadakis, G. E. (2019). Variational physics-informed neural networks for solving partial differential equations. arXiv preprint arXiv:1912.00873.

[3] Lu, L., Pestourie, R., Yao, W., Wang, Z., Verdugo, F., & Johnson, S. G. (2021). Physics-informed neural networks with hard constraints for inverse design. SIAM Journal on Scientific Computing, 43(6), B1105-B1132.

[4] Carl Leake and Daniele Mortari. Deep theory of functional connections: A new method for estimating the solutions of partial differential equations. Machine learning and knowledge extraction, 2(1):37–55, 2020.

References

[5] Gao, H., Sun, L., & Wang, J. X. (2021). PhyGeoNet: Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state PDEs on irregular domain. Journal of Computational Physics, 428, 110079.

[6] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378:686–707, 2019.

[7] Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. SIAM Journal on Scientific Computing, 43(5):A3055–A3081, 2021.

[8] Ameya D Jagtap and George Em Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Communications in Computational Physics, 28(5):2002–2041, 2020.

References

[9] Ben Moseley, Andrew Markham, and Tarje Nissen-Meyer. Finite basis physics-informed neural networks (fbpinns): a scalable domain decomposition approach for solving differential equations. arXiv preprint arXiv:2107.07871, 2021.

[10] Hailong Sheng and Chao Yang. Pfnn: A penalty-free neural network method for solving a class of second-order boundary-value problems on complex geometries. Journal of Computational Physics, 428:110085, 2021.

[11] Hailong Sheng and Chao Yang. Pfnn-2: A domain decomposed penalty-free neural network method for solving partial differential equations. arXiv preprint arXiv:2205.00593, 2022.

Thank You!

Paper Link: https://arxiv.org/pdf/2210.03526.pdf

Code Link: https://github.com/csuastt/HardConstraint