Neural Domain Decomposition for Variable Coefficient Poisson Solvers

Sebastian Barschkis*, Zitong Li*, Hengjie Wang, Aparna Chandramowlishwaran

University of California, Irvine

UCI University of California, Irvine

Motivation

Why speed up the Poisson solving step?

- It's the bottleneck of many flow simulations. MFiX's [2] fluid solver, for example, spends more than 80% of its time on solving the Poisson equation.

Why use Neural PDE solvers?

- Predicting PDE solutions is generally faster than solving PDEs numerically.
- Training is compute intensive - but only needs to be done once. Trained models can predict solutions subject to any input boundary condition (BC).

Why choose a U-Net?

- Convolutions are very efficient for large input domains. A fully-connected model would need very wide layers to map the right hand side (RHS) and the variable coefficient to the solution.
- Can capture the correlation of adjacent cells (similar to finite differences (FD)).

Variable Coefficient Poisson Equation

We consider the variable coefficient 2D Poisson equation:

$$\begin{aligned} \nabla \cdot (a \nabla p(\boldsymbol{x})) &= f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega \\ p(\boldsymbol{x}) &= g(\boldsymbol{x}), \quad \boldsymbol{x} \in \partial\Omega \end{aligned} \tag{1}$$

For our numerical ground truth data, we solve the equation by discretizing it with FD and solving the corresponding linear system

$$A(a)p = b(a, f)$$

with PCG. For a constant coefficient, the system becomes

$$Ap = b$$

where A only depends on size of the domain.

Superposition Principle

Poisson's equation (zero BC):

$$\begin{aligned} \nabla^2 p^p(\boldsymbol{x}) &= f(\boldsymbol{x}), \quad && \boldsymbol{x} \in \Omega \\ p^p(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \partial\Omega \end{aligned}$$

Laplace's equation:

$$\begin{aligned} \nabla^2 p^l(\boldsymbol{x}) &= 0, \quad && \boldsymbol{x} \in \Omega \\ p^l(\boldsymbol{x}) &= g(\boldsymbol{x}), && \boldsymbol{x} \in \partial\Omega \end{aligned}$$

Poisson's equation (non-zero BC):

$$p(\boldsymbol{x}) = p^p(\boldsymbol{x}) + p^l(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega$$

Neural PDE solver

- **Architecture:** U-Net [3] with skip connections.
- **Optimizations:** Batch-normalization layers after each convolution, 'LeakyRelu' activations in contracting and expanding paths.

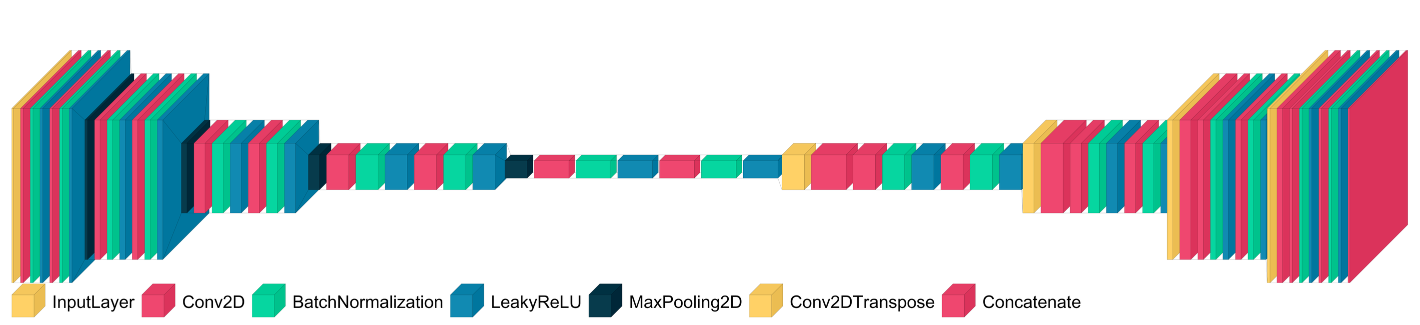


Figure 1. Modified U-Net architecture for fast Poisson solver

Training

- **Loss:** Supervised data and analytical PDE components:

$$\mathcal{L}_{Total} = \mathcal{L}_{Data} + \alpha \cdot \mathcal{L}_{PDE}$$

- **Error metric:** Mean-squared-error for both $\mathcal{L}_{Data}$ and $\mathcal{L}_{PDE}$.
- **PDE residual:** $\mathcal{L}_{PDE}$ minimizes the residual

$$\begin{aligned} &[(a_{i+1,j} + a_{i,j})(p_{i+1,j} - p_{i,j}) - (a_{i,j} + a_{i-1,j})(p_{i,j} - p_{i-1,j}) \\ &+ (a_{i,j+1} + a_{i,j})(p_{i,j+1} - p_{i,j}) - (a_{i,j} + a_{i,j-1})(p_{i,j} - p_{i,j-1}) - 2f_{i,j}h^2]^2 = 0 \end{aligned}$$

resulting from discretization of Eq. 1 with FD on a uniform grid with height h.

- **Model inputs:** n 32x32 randomly/uniformly generated grids for BC g, variable coefficient a, and RHS f.
- **Labels:** Solution p pre-computed with PCG solver.
- **Training:** 20k samples/epoch, 90/10 train/valid split, (mini) batchsize 64.

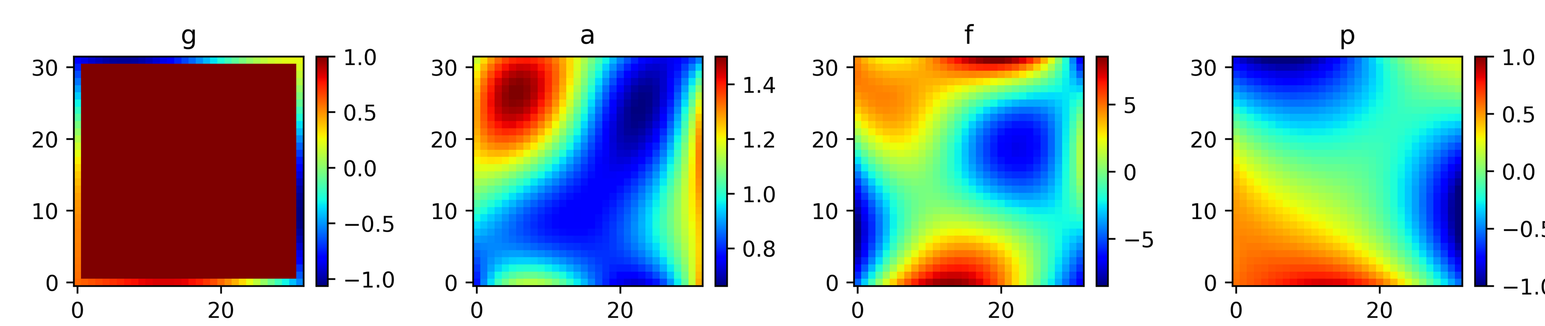
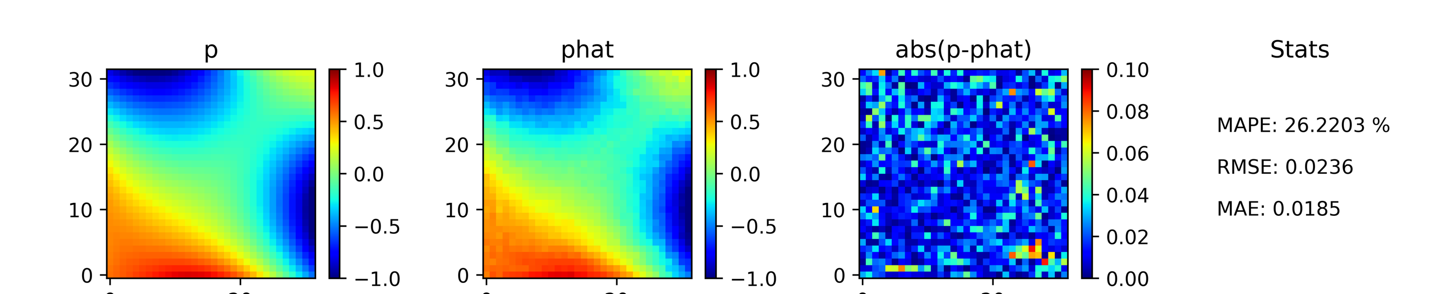


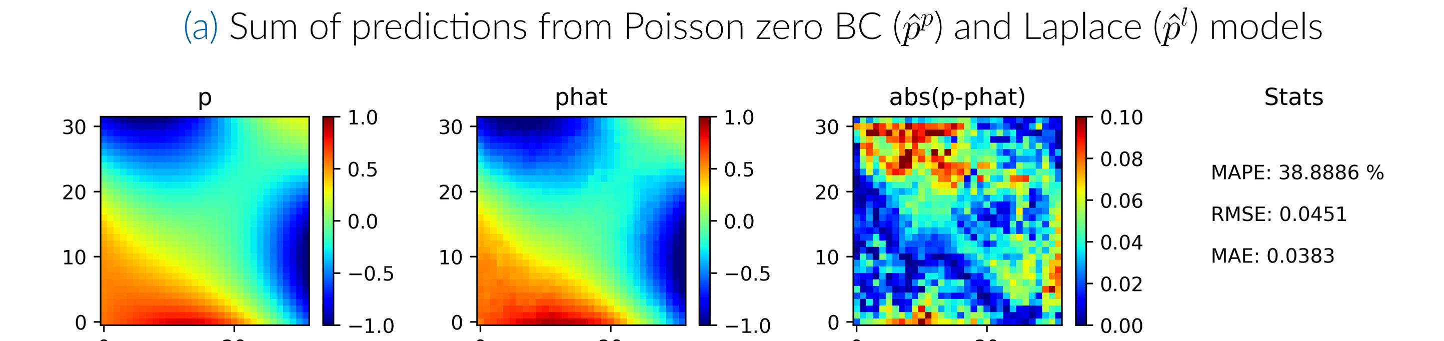
Figure 2. Example of one training sample: Inputs g, a, f, and label p (inhomogeneous Poisson)

Accuracy

- The sum of predictions $\hat{p}^p + \hat{p}^l$ yields a lower error than directly predicting $\hat{p}$



(a) Sum of predictions from Poisson zero BC ($\hat{p}^p$) and Laplace ($\hat{p}^l$) models

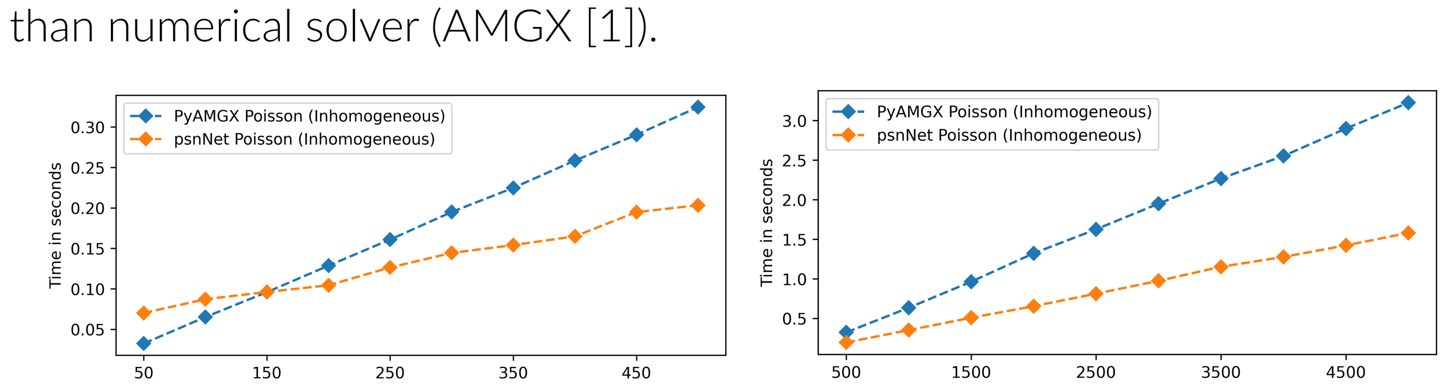


(b) Prediction from Poisson non-zero BC ($\hat{p}$)

Figure 3. Comparison of ground truth p, predictions $\hat{p}$ and errors.

Performance of AMG vs Neural PDE solver

- **Solving many samples ($n > 150$):** Neural Operator yields better performance than numerical solver (AMGX [1]).



(a) Solving a *few* samples (up to 500) (b) Solving *many* samples (up to 5,000)

Figure 4. Time to solve (`pyamgx.solve()`) and predict (`tf.predict()`) n (32x32) inhomogeneous Poisson instances using one V100 GPU. Excluding GPU data upload / download times.

Domain Decomposition Method (DDM)

- **Subdomain data:** Generated from synthetic samples of spatial size 2x2 (64x64 resolution). These are divided into 4 1x1 subdomains.
- **Step 1:** Solve Poisson's eq. with zero BC (p^p) on each subdomain:

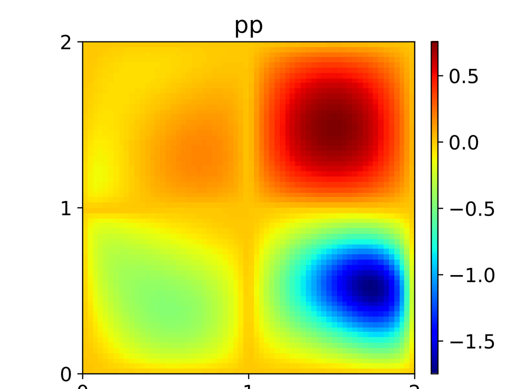


Figure 5. 2x2 subdomains for Poisson's equation with zero BC.

- **Step 2:** Solve the Schur system to find the BC for Laplace (MSE ≈6e-3):

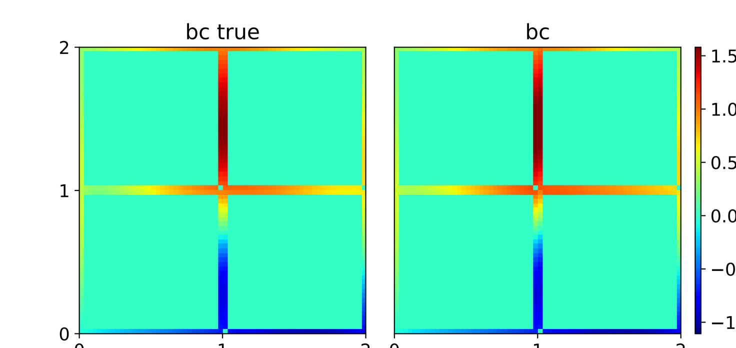


Figure 6. BC for Laplace. Left: Ground truth. Right: Computed solution.

- **Step 3:** Solve Laplace's eq. (p^l) on each subdomain using the BC from step 2:

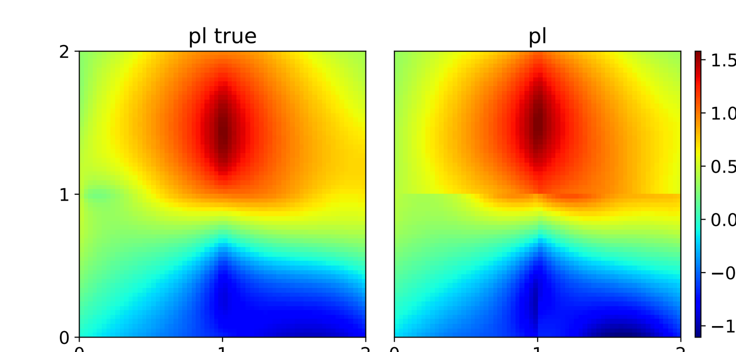


Figure 7. 2x2 subdomains for Laplace's equation. Left: Ground truth. Right: Computed solution.

- **Step 4:** Find the final solution $p = p^l + p^p$ (MAE ≈3e-2):

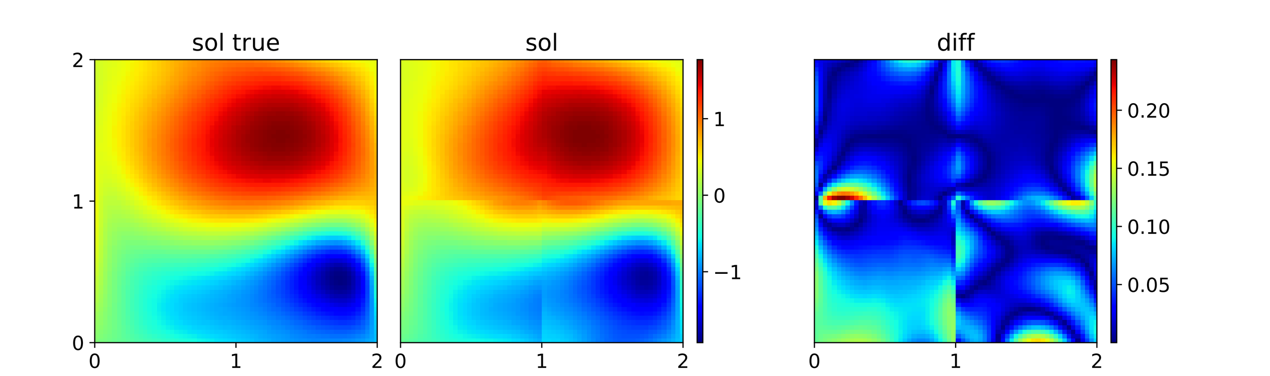


Figure 8. Final Poisson solution. Left: Ground truth. Center: Computed solution. Right: Error.

Schur Complement

$$\begin{aligned} p_0 &= p_0^l + p_0^p = c_0^T g_0 + p_0^p \\ p_1 &= p_1^l + p_1^p = c_1^T g_1 + p_1^p \\ g &= \frac{p_0 + p_1}{2} \end{aligned}$$

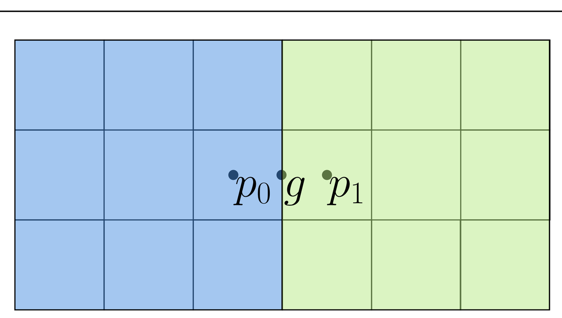


Figure 9. The BC is on the cell interface rather than cell center. The equation shown above is for a point on the subdomain interface.

References

[1] Algebraic Multigrid Solver (AmgX) Library, `https://github.com/NVIDIA/AMGX`, (2023-08-01).

[2] MFIX-Exa, `https://amrex-codes.github.io/MFIX-Exa/`, (2023-08-01).

[3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, 2015.

[4] Hengjie Wang, Robert Planas, Aparna Chandramowlishwaran, and Ramin Bostanabad. Mosaic flows: A transferable deep learning framework for solving pdes on unseen domains. *Computer Methods in Applied Mechanics and Engineering*, 389:114424, 2022.