Report: PINN for Solving the Biharmonic Equation with Neumann-Type Boundary Conditions (Phase-1)

1. Introduction

The biharmonic equation is a fourth-order partial differential equation (PDE) that arises in various applications in physics and engineering.

The general biharmonic problem is stated as:

$$\boldsymbol{\Delta^2 u = f \ \text{ in } \ \Omega}$$

where $\Delta^2 = \Delta(\Delta u)$is the biharmonic operator, obtained by applying the Laplacian operator twice. For a two-dimensional domain $\Omega \subset \mathbb{R}^2$, this expands to:

$$\Delta^2 u = \frac{\partial^4 u}{\partial x_1^4} + 2\frac{\partial^4 u}{\partial x_1^2 \, \partial x_2^2} + \frac{\partial^4 u}{\partial x_2^4}$$

2. Problem Formulation

We consider the biharmonic equation in the unit square $\Omega = (0,1)^2$:

$$\Delta^2 u = f \quad \text{in } \ \Omega,$$

with mixed boundary conditions:

- Essential (Dirichlet): $u = g_1$on $\partial\Omega$
- Neumann-type: $\frac{\partial^2 u}{\partial n^2} = g_2$on $\partial\Omega$

The second normal derivative is defined as:

$$\frac{\partial^2 u}{\partial n^2} = \mathrm{n}^T \cdot D^2 u \cdot \mathrm{n} = n_1^2 \frac{\partial^2 u}{\partial x_1^2} + 2n_1 n_2 \frac{\partial^2 u}{\partial x_1 \, \partial x_2} + n_2^2 \frac{\partial^2 u}{\partial x_2^2}$$

where $\mathrm{n} = (n_1, n_2)$is the outward normal vector and $D^2 u$is the Hessian matrix of second derivatives.

Empirical Loss: $L(\theta) = \lambda_{int}\mathcal{L}_{int}(\theta) + \lambda_{bc}\mathcal{L}_{bc}(\theta)$

Where:

- **Interior Loss:** $\mathcal{L}_{int}(\theta) = \frac{1}{N_i}\sum_{j=1}^{N_i}\left|\Delta^2 u_\theta(x_j) - f(x_j)\right|^2$
- **Boundary Loss:** $\mathcal{L}_{bc}(\theta) = \frac{1}{N_b}\sum_{k=1}^{N_b}[|u_\theta y_k|^2 + |\Delta u_\theta y_k - g_2(y_k)|^2]$
-

Example 3.1: We considered the problem in two-dimensional domain $\Omega = (0,1)^2$, with known exact solution $u = \frac{1}{2\pi^2}\sin(\pi x_1)\sin(\pi x_2)$. The Boundary condition is unchanged.

Example 3.2: We considered the problem in two-dimensional domain $\Omega = (0,1)^2$, with known exact solution $u = x_1^2 x_2^2 (1 - x_1)^2 (1 - x_2)^2$. The Boundary condition is unchanged.

3. Neural Network Setup

3.1 Architecture

The solution $\mathrm{u}(\mathrm{x}_1, \mathrm{x}_2)$is approximated by a fully connected feed-forward neural network (multi-layer perceptron), implemented in the class U_FCN. The architecture used in this work consists of:

- Input layer: 2 neurons (coordinates $\mathrm{x}_1, \mathrm{x}_2$)
- Hidden layers: 6 layers with widths 100 each.
- Activation: Hyperbolic Tangent(tanh) at each hidden layer
- Output layer: scalar output representing $u_\theta(x_1, x_2)$

Total number of parameters:

Choice of activation:

The activation function **tanh** gave results:

1. It is smooth and infinitely differentiable.
2. It allows stable computation of second-order and fourth-order derivatives using automatic differentiation.
3. ReLU is not suitable for high-order PDEs, as its second derivative vanishes almost everywhere.

Also tried with Sigmoid but the results was a step like function.

3.2 Sampling Strategy

Training requires interior points to enforce the PDE and boundary points to enforce boundary conditions.

Interior sampling:

We draw N points uniformly inside the computational domain $\Omega = (0,1)^2$

As $\mathrm{X_i} \sim \mathcal{U}([0,1]^2), \mathrm{i} = 1, \ldots, \mathrm{N}.$

We chose N = 5000 interior points

Boundary sampling:

The boundary $\partial\Omega$consists of four edges. We draw $\mathrm{M}/4$points uniformly on each edge (with total boundary points $\mathrm{M} = 1000$). For each edge, we associate the outward unit normal vector:

- Bottom edge ($\mathrm{x}_2 = 0$): $\mathrm{n} = (0, -1)$
- Right edge ($\mathrm{x}_1 = 1$): $\mathrm{n} = (1,0)$
- Top edge ($\mathrm{x}_2 = 1$): $\mathrm{n} = (0,1)$
- Left edge ($\mathrm{x}_1 = 0$): $\mathrm{n} = (-1,0)$

Resampling:

- During Adam optimization, interior and boundary points are resampled every iteration.
  This reduces overfitting and improves generalization.
- During L-BFGS optimization, a fixed set of points is used for stability.

4. Uncertainty Based Loss Weighting

We introduce learnable log-variances: log_var_interior & log_var_boundary

and defined the total loss as:

$$\mathcal{L}_\theta = \frac{1}{2}[e^{-s_1}\mathcal{L}_{\text{interior}} + s_1 + e^{-s_2}\mathcal{L}_{\text{boundary}} + s_2]$$

Where s_1 = log_var_interior and s2 = log_var_boundary

5. Training Procedure

- At every epoch, new interior and boundary samples are drawn.
- Adam optimizer updates the parameters.
- Uncertainty parameters (log_var_*) are updated together with the network
- After 30000 Adam epochs, switched the optimize to LBFGS(for 10000 epochs)
- Losses and validation error are logged.

6. Error Computation and Validation

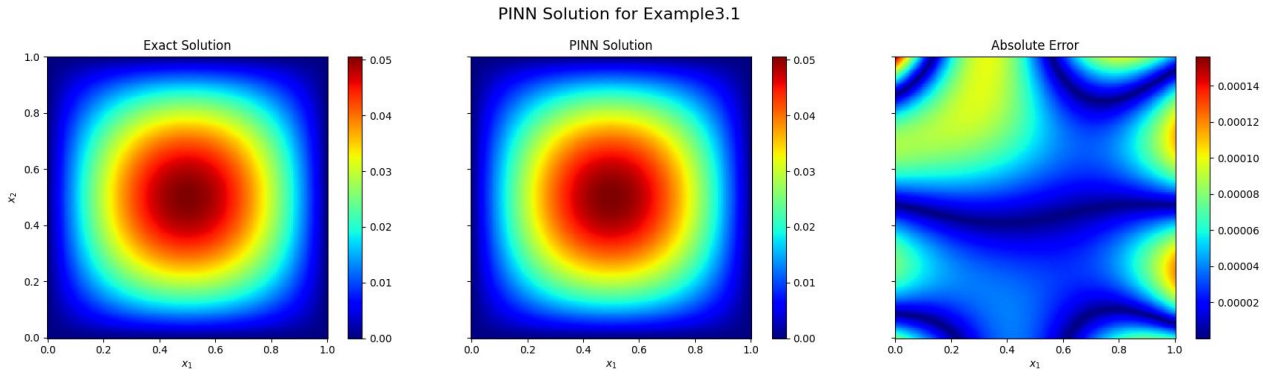
In both examples, computed L^2 – error $\left|\left|u - u_\theta\right|\right|_{L^2(\Omega)}$, L^2 relative - $\frac{\left|\left|u-u_\theta\right|\right|_{L^2(\Omega)}}{\left|\left|u\right|\right|_{L^2}}$

Energy error (H1) - $\left|\left|u - u_\theta\right|\right|_{L^2(\Omega)}$ + $\left|\left|\nabla(u - u_\theta)\right|\right|_{L^2}$

H1 Relative - $\frac{||u-u_\theta||_{L^2(\Omega)}+||\nabla(u-u_\theta)||_{L^2}}{||u||_{L^2(\Omega)}+||\nabla||_{L^2(\Omega)}}$

7. Visualization of Results

Results Example 3.1



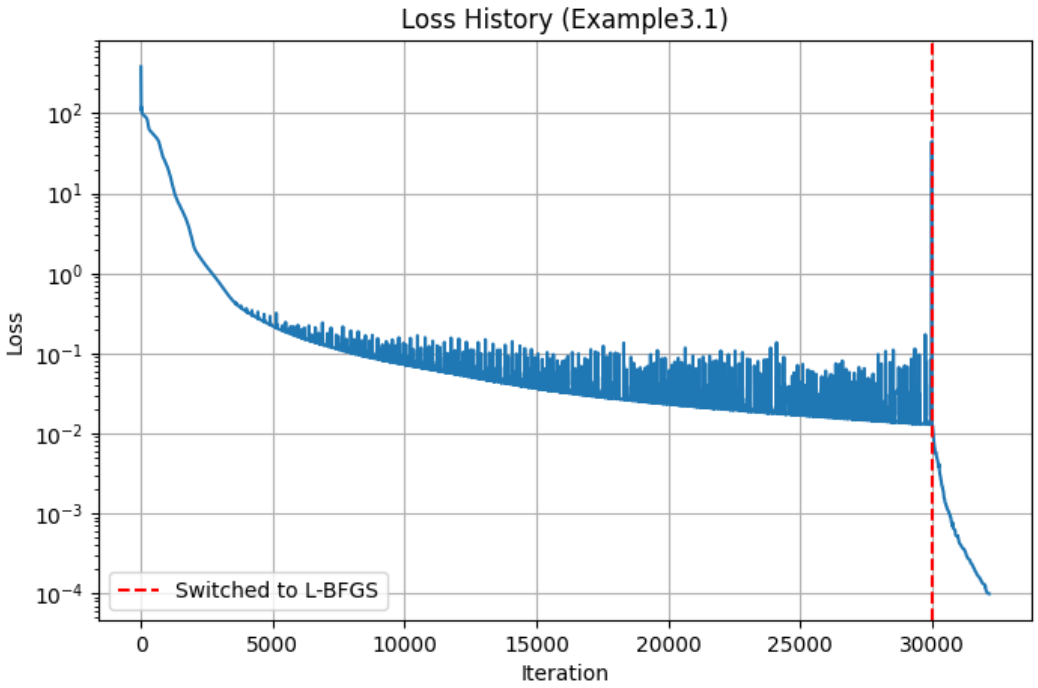
Final Loss: 9.9044e-05

L2 Relative Error: 1.9286e-03

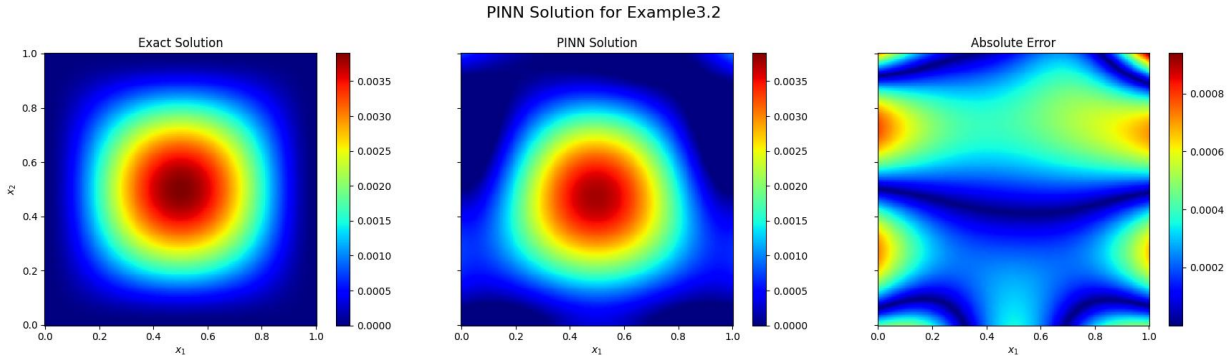
H1 Relative Error: 3.6912e-03

H2 Relative Error: 6.5490e-03

Loss Curve



Results Example 3.2



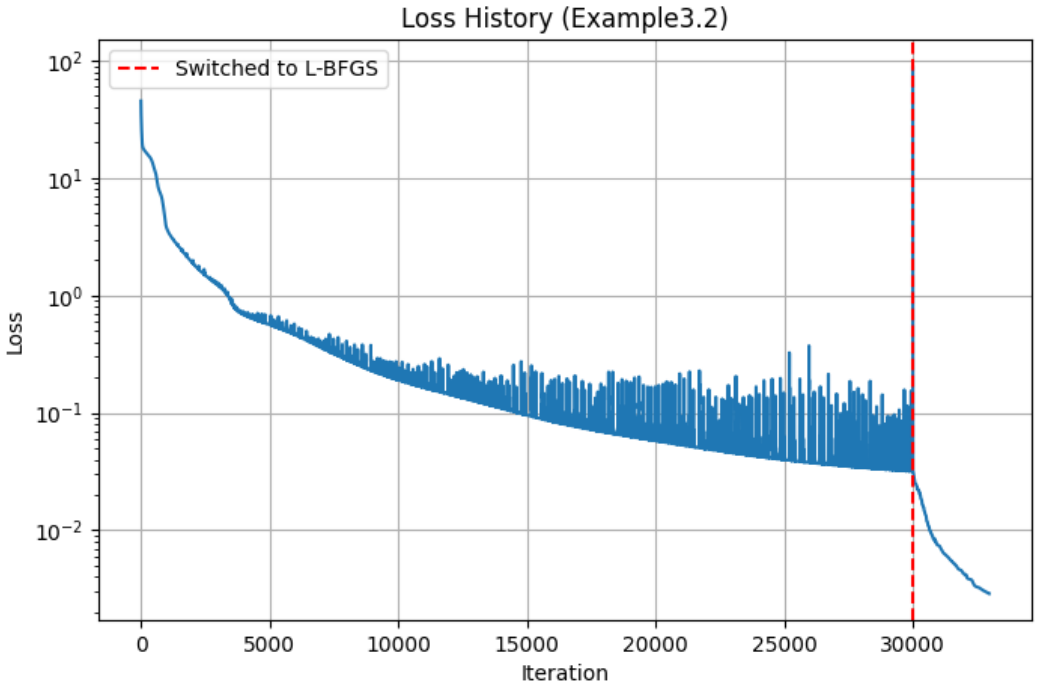
Final Loss: 2.8825e-03

L2 Relative Error: 2.0665e-01

H1 Relative Error: 3.5027e-01

H2 Relative Error: 3.5480e-01

Loss Curve



Contributions

Every One in team has equally Contributed to this Project Completion.