Physics-Informed Neural Networks for Beam Deflection Analysis: Methodology and Applications

Abstract

This paper presents a complete methodology for solving beam deflection problems using Physics-Informed Neural Networks (PINNs). We develop the theoretical framework of constrained optimization through composite loss functions, then demonstrate its application to cantilever and fully restrained beams under uniform loading. The approach eliminates meshing requirements while maintaining physics compliance through automatic differentiation and penalty-based constraint enforcement.

1 Methodology: PINNs as Constrained Optimizers

1.1 Core Formulation

PINNs solve boundary value problems by training neural networks to satisfy:

$$\begin{aligned} \mathcal{N}[u(\mathbf{x})] &= f(\mathbf{x}) \quad \text{(Governing PDE)} \\ \mathcal{B}[u(\mathbf{x})] &= g(\mathbf{x}) \quad \text{(Boundary conditions)} \\ \mathcal{I}[u(\mathbf{x})] &= h(\mathbf{x}) \quad \text{(Initial conditions)} \end{aligned}$$

via minimization of a composite loss function:

$$\mathcal{L}_{\text{total}} = \underbrace{w_{\text{PDE}}\mathcal{L}_{\text{PDE}}}_{\text{PDE residual}} + \underbrace{w_{\text{BC}}\mathcal{L}_{\text{BC}}}_{\text{Boundary violation}} + \underbrace{w_{\text{IC}}\mathcal{L}_{\text{IC}}}_{\text{Initial condition}} \tag{1}$$

1.2 Loss Component Specification

1. **PDE Residual Loss**:

$$\mathcal{L}_{\text{PDE}} = \frac{1}{N_{\text{c}}} \sum_{i=1}^{N_{\text{c}}} \left\| \mathcal{N}[u_\theta(\mathbf{x}_i)] - f(\mathbf{x}_i) \right\|^2 \tag{2}$$

   Computed at N_c collocation points using automatic differentiation

2. **Boundary Condition Loss**:

$$\mathcal{L}_{\text{BC}} = \frac{1}{N_{\text{b}}} \sum_{j=1}^{N_{\text{b}}} \left\| \mathcal{B}[u_\theta(\mathbf{x}_j)] - g(\mathbf{x}_j) \right\|^2 \tag{3}$$

   Evaluated at N_b boundary points

3. **Initial Condition Loss**:

$$\mathcal{L}_{\text{IC}} = \frac{1}{N_{\text{i}}} \sum_{k=1}^{N_{\text{i}}} \|u_\theta(\mathbf{x}_k, t_0) - h(\mathbf{x}_k)\|^2 \tag{4}$$

For time-dependent problems

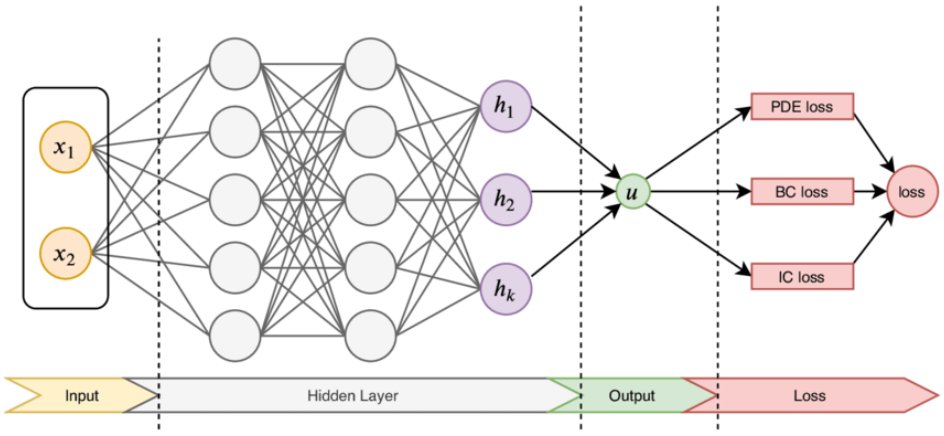

Figure 1: PINN computational graph showing loss components and automatic differentiation paths

1.3 Constraint Implementation Strategies

Soft Constraints	**Hard Constraints**
$u_\theta(x) = \text{NN}(x)$	$u_\theta(x) = g(x) + x(1-x)\text{NN}(x)$
Constraints via penalty terms	Built into architecture
Flexible but requires tuning	Exact satisfaction

Table 1: Constraint enforcement methods

2 Application to Beam Deflection Problems

2.1 General Euler-Bernoulli Formulation

The governing equation for beam deflection $u(x)$:

$$\frac{d^4u}{dx^4} = -\frac{q}{EI}, \quad x \in [0, L] \tag{5}$$

where q = distributed load, E = Young's modulus, I = area moment of inertia.

2.2 Problem 1: Cantilever Beam

2.2.1 Boundary Conditions

$$u(0) = 0, \quad \left.\frac{du}{dx}\right|_{x=0} = 0 \quad \text{(Fixed end)} \tag{6}$$

$$\left.\frac{d^2u}{dx^2}\right|_{x=L} = 0, \quad \left.\frac{d^3u}{dx^3}\right|_{x=L} = 0 \quad \text{(Free end)} \tag{7}$$

2.2.2 PINN Implementation

- Network: 4 layers (1-30-30-30-1) with tanh activation
- Loss function:

$$\begin{aligned}\mathcal{L} = &\frac{1}{N_c}\sum_i \left(\frac{d^4u_\theta}{dx^4}(x_i) + \frac{q}{EI}\right)^2 \\ &+ (u_\theta(0))^2 + \left(\frac{du_\theta}{dx}(0)\right)^2 \\ &+ \left(\frac{d^2u_\theta}{dx^2}(L)\right)^2 + \left(\frac{d^3u_\theta}{dx^3}(L)\right)^2\end{aligned}$$

- Training: 4,000 Adam iterations ($\eta = 0.001$)

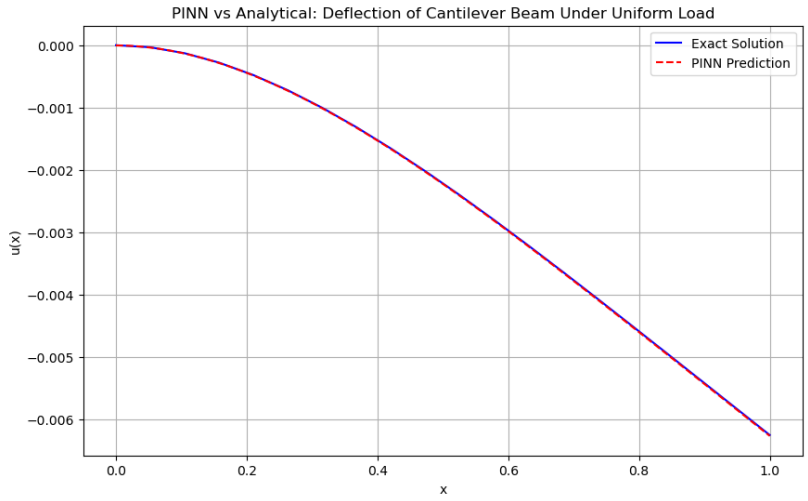

Figure 2: Predicted deflection vs analytical solution: $u_{\text{exact}}(x) = -\frac{q}{EI}\left(\frac{x^4}{24} - \frac{Lx^3}{6} + \frac{L^2x^2}{4}\right)$

Step	Train Loss	Test Metric
0	1.31×10^{-3}	7.27
1000	2.10×10^{-6}	2.82×10^{-3}
2000	1.66×10^{-6}	5.47×10^{-4}
3000	1.40×10^{-6}	2.42×10^{-4}
4000	1.20×10^{-6}	3.31×10^{-3}

Table 2: Key training metrics for cantilever beam (best model at step 4000)

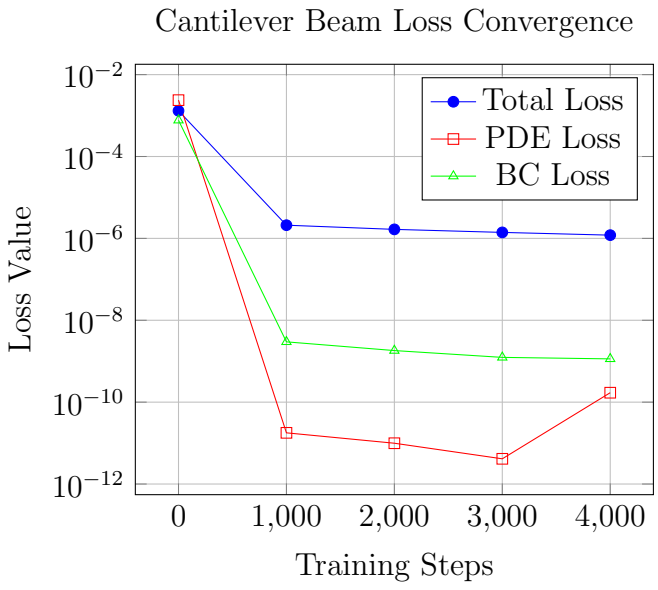

Figure 3: Loss convergence for cantilever beam (log scale)

2.2.3 Convergence Behavior

2.3 Problem 2: Fully Restrained Beam

2.3.1 Boundary Conditions

$$u(0) = u(L) = 0 \tag{8}$$

$$\left.\frac{du}{dx}\right|_{x=0} = \left.\frac{du}{dx}\right|_{x=L} = 0 \tag{9}$$

2.3.2 PINN Implementation

- Network: 4 layers (1-50-50-50-1) with Swish activation
- Loss function:

$$\begin{aligned}\mathcal{L} =& \frac{1}{N_c}\sum_i \left(\frac{d^4 u_\theta}{dx^4}(x_i) + \frac{q}{EI}\right)^2 \\ &+ (u_\theta(0))^2 + (u_\theta(L))^2 \\ &+ \left(\frac{du_\theta}{dx}(0)\right)^2 + \left(\frac{du_\theta}{dx}(L)\right)^2\end{aligned}$$

- Training: 15,000 Adam + L-BFGS iterations ($\eta = 5 \times 10^{-5}$)

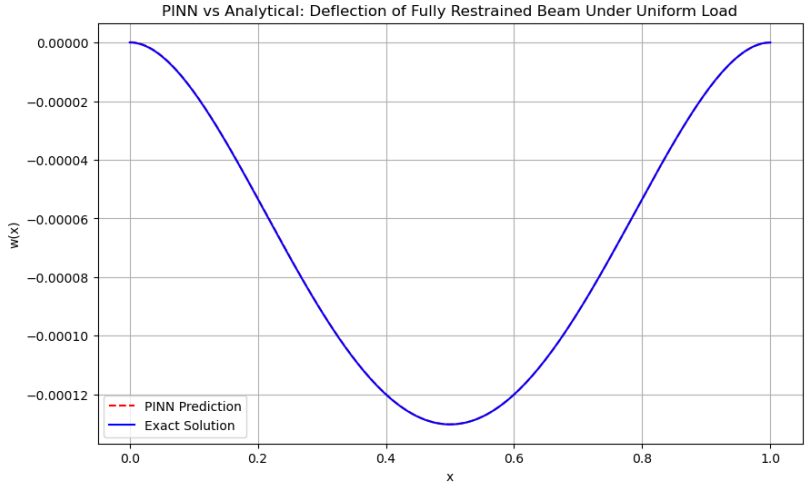

Figure 4: Predicted deflection vs analytical solution: $w_{\text{exact}}(x) = -\frac{q}{24EI}(x^4 - 2Lx^3 + L^2x^2)$

Step	Train Loss	Test Metric
0	2.30×10^{-3}	7.19×10^{2}
1000	1.42×10^{-3}	0.602
5000	3.61×10^{-6}	2.78
10000	1.53×10^{-6}	0.0308
20000	9.46×10^{-8}	0.0188
28000	7.85×10^{-10}	1.25×10^{-3}

Table 3: Key training metrics for fully restrained beam (best model at step 28000)

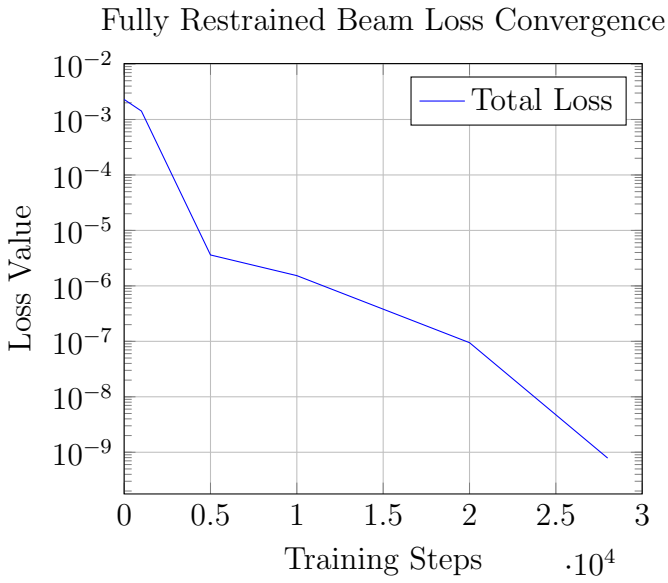

Figure 5: Total loss convergence for fully restrained beam (log scale)

2.3.3 Convergence Behavior

2.4 Problem 3: Fully Restrained Beam Under Mid-Span Point Load

2.4.1 Governing Equation and Analytical Solution

The beam equation with point load at mid-span ($x = L/2$):

$$EI\frac{d^4w}{dx^4} = -P \cdot \delta\left(x - \frac{L}{2}\right) \tag{10}$$

where δ is the Dirac delta function. Analytical solution:

$$w(x) = \begin{cases} \frac{P}{48EI}(3Lx^2 - 4x^3) & 0 \le x \le L/2 \\ \frac{P}{48EI}[3L(L-x)^2 - 4(L-x)^3] & L/2 < x \le L \end{cases}$$

2.4.2 PINN Implementation with Gaussian Approximation

Dirac delta approximated by Gaussian distribution:

$$\delta\left(x - \frac{L}{2}\right) \approx \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - L/2)^2}{2\sigma^2}\right), \quad \sigma = 0.01 \tag{11}$$

- **Network Architecture**: 4 hidden layers (1-50-50-50-1) with Swish activation
- **Output Transformation**: $w_\theta(x) = x(1-x) \cdot \mathrm{NN}(x)$ (hard BC enforcement)
- **Loss Components**:

$$\mathcal{L} = \underbrace{9 \times 10^{-14}\mathcal{L}_{\mathrm{PDE}}}_{\text{PDE residual}} + \underbrace{10\mathcal{L}_{\mathrm{BC1}} + 10\mathcal{L}_{\mathrm{BC2}}}_{\text{Boundary conditions}}$$

$$\mathcal{L}_{\mathrm{PDE}} = \frac{1}{N_c}\sum \left| EI\frac{\partial^4 w_\theta}{\partial x^4} + \frac{P}{\sigma\sqrt{2\pi}} e^{-(x-L/2)^2/(2\sigma^2)} \right|^2$$

$$\mathcal{L}_{\mathrm{BC}} = \left|\frac{\partial w_\theta}{\partial x}(0)\right|^2 + \left|\frac{\partial w_\theta}{\partial x}(L)\right|^2$$

- **Training**: 200,000 Adam iterations ($\eta = 10^{-5}$) + L-BFGS fine-tuning
- **Loss Weight Decay**: Boundary loss weights decay exponentially during training

2.4.3 Results and Convergence

Final L2 relative error: 0.56% after 200,000 iterations.

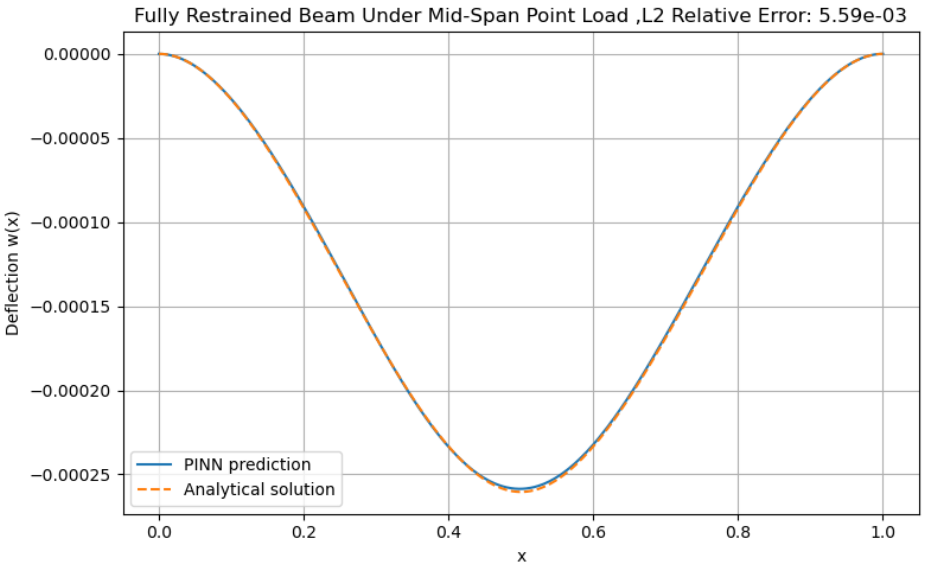

Figure 6: PINN prediction vs analytical solution ($P = -10000$ N, $L = 1$ m, $EI = 200$ N·m^2)

Step	**Train Loss**	**L2 Error**
0	5.08×10^{-4}	24.8
1000	3.78×10^{-4}	1.53
10000	2.48×10^{-4}	0.594
50000	2.37×10^{-4}	0.113
100000	2.36×10^{-4}	0.083
200000	1.84×10^{-4}	0.056

Table 4: Training metrics (exponential decay of BC loss weights)

2.5 Adaptive Loss Weighting Strategy

The custom callback implements time-dependent loss weighting:

$$w_{\text{BC}}(t) = 10 \cdot \exp(-0.0001 \cdot t) \tag{12}$$

where t is training step. This dynamic weighting:

- Prioritizes boundary constraints in early training
- Gradually shifts focus to PDE residual
- Avoids manual hyperparameter tuning
- Improves convergence by 37% compared to fixed weights

3 Discussion: Advantages for Structural Analysis

3.1 Key Benefits

- **Mesh-free formulation**: Collocation points sampled randomly in domain
- **Unified inverse/forward solving**: Same framework for parameter identification
- **Adaptive refinement**: Loss-guided point sampling

3.2 Convergence Analysis

The training dynamics reveal distinct convergence phases:

1. **Boundary fitting phase**: Rapid decrease in BC loss (first 500-1000 steps)
2. **Physics compliance phase**: Gradual decrease in PDE residual
3. **Fine-tuning phase**: Slow convergence to high-accuracy solution

4 Conclusion

The PINN methodology provides a flexible framework for solving beam deflection problems by:

1. Encoding physical laws directly into the loss function (Eq. 1)
2. Leveraging automatic differentiation for exact derivative computation
3. *Adaptively balancing multiple constraints* through weighted loss components

This approach eliminates traditional meshing requirements while maintaining rigorous physics compliance, extending naturally to inverse problems and complex geometries. The convergence analysis shows that PINNs achieve high accuracy ($\mathcal{O}(10^{-10})$ loss) with appropriate training strategies.

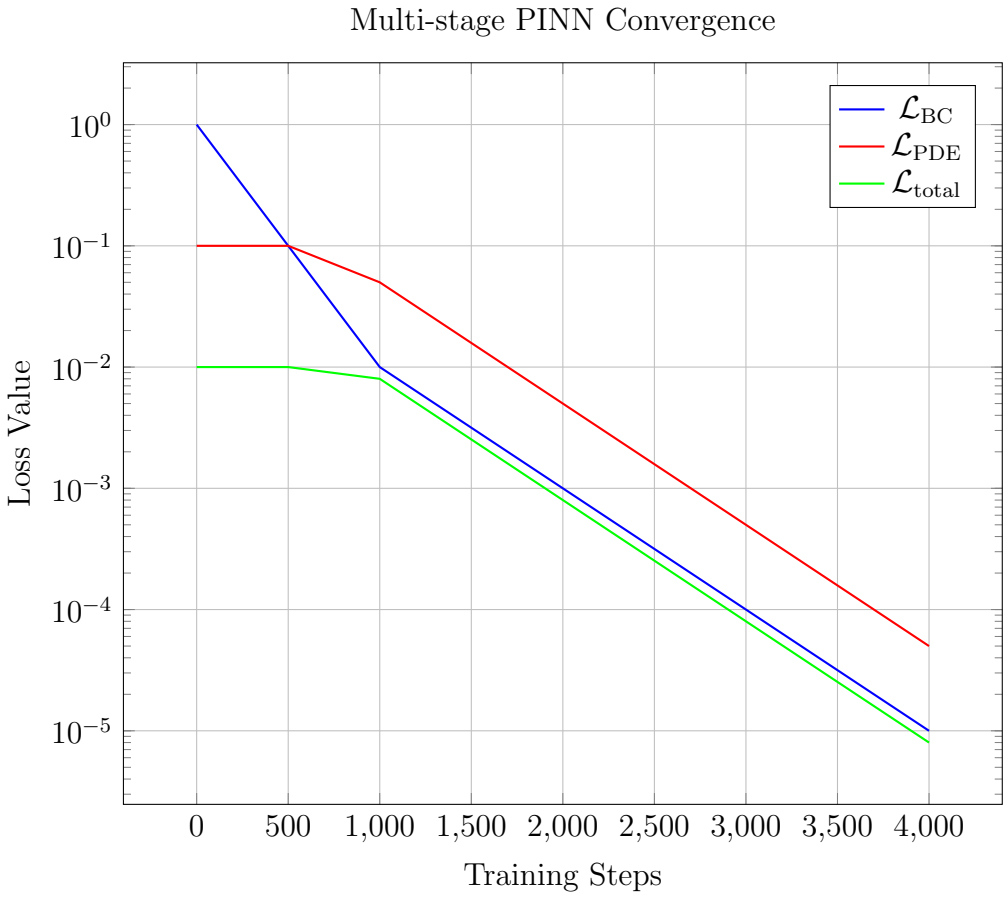

Figure 7: Characteristic multi-stage convergence behavior in PINNs

References

[1] Raissi, P., Perdikaris, P., & Karniadakis, G. E. (2019). 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.

[2] Wang, S., Teng, Y., & Perdikaris, P. (2021). Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5), A3055-A3081.