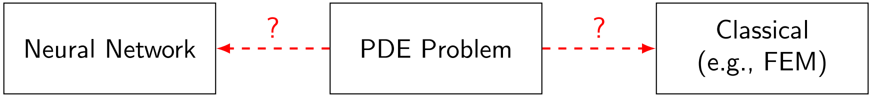
Comparison of Classical Numerical PDE Methods and Deep Learning-Based PDE Solvers

Vincent Scholz & Dr. Yaohua Zang

Professorship for Data-driven Materials Modeling

May 28, 2025

Which Solver Should We Use?



Observation

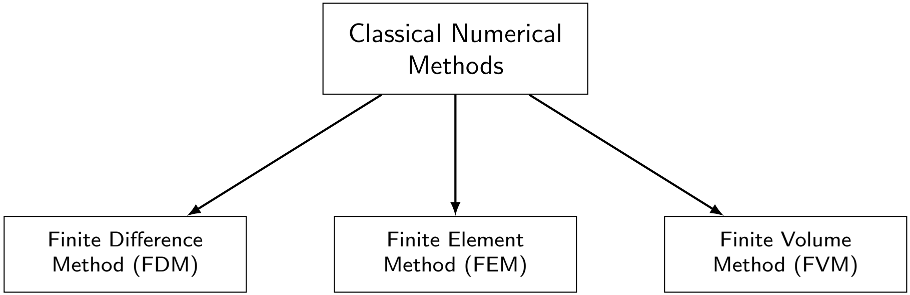
From what we have seen so far, it's still unclear when to use classical numerical methods or neural networks to solve a given PDE problem.

- Classical methods are reliable and well-understood — but expensive in high dimensions
- Neural networks can generalize — but may lack guarantees or interpretability
- Choosing between them is still an open research question

Table of Contents

Table of Contents

Classical Numerical Methods


Classical Numerical Methods
Finite Difference Method (FDM)
Finite Element Method (FEM)
Finite Volume Method (FVM)

Advantages of Classical Methods

Accuracy & Convergence Theory

- Well-established mathematical theories ensure convergence and stability
- Rigorous error estimates for FDM and FEM

Flexibility & Generality

- Handle arbitrary PDEs with proper formulation
- FEM particularly powerful for complex geometries and adaptive meshing

Interpretability & Reliability

- Solutions based on first-principles physics
- No need for large datasets

Computational Inefficiency

High-dimensional PDEs suffer from the curse of dimensionality:

- Boltzmann equation: $d = 7$
- Radiative Transfer: $d \geq 5$
- Black-Scholes: $d >> 1$
- Schrödinger: $d >> 1$

Fine grids required:

- Both spatial and temporal resolution
- High computational costs

Complexity of Implementation:

- Assembly of system matrices requires careful integration over elements.
- Boundary condition enforcement can be subtle and error-prone.
- Handling curved boundaries, adaptive refinement, and mesh conformity adds significant code complexity.

Many-Query Problems

Applications requiring multiple PDE solves:

- Design optimization
- Uncertainty quantification
- Inverse problems (EIT, Seismic Waves)

Example: Flow past airfoils

- Various Mach numbers
- Different angles of attack
- Lift & drag evaluation

Additional Challenges

Multiscale & Multiphysics

- Turbulence modeling
- Geophysics applications
- Material science
- Multiple spatial & temporal scales
- Mesh resolution constraints

Grid Generation Dependency

- Careful meshing required
- Complex for irregular geometries
- Poor meshing leads to numerical errors and instability

Table of Contents

Why Use Deep Learning for PDEs?

Deep learning-based PDE solvers address computational challenges of classical methods:

- Mesh-free approaches
- High-dimensional capability
- Fast inference
- Data-driven solutions

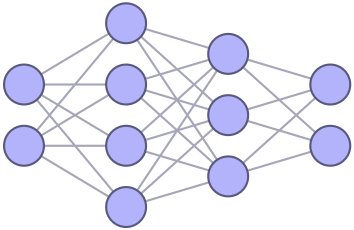


Figure: Deep Neural Network Architecture

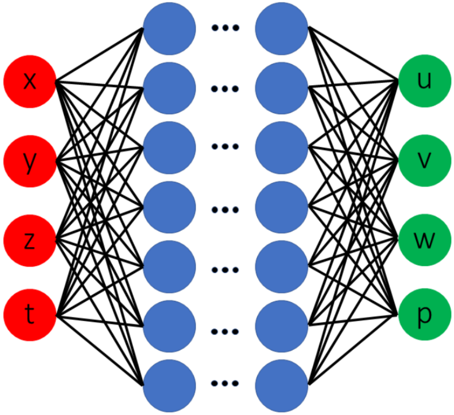
Mesh-Free Methods

Traditional methods:

- Require mesh generation
- Grid-dependent solutions
- Complex for irregular geometries

Deep learning methods:

- Physics-Informed Neural Networks (PINNs)
- Deep Ritz method (DeepRitz)
- No mesh generation required



Mesh-free approaches

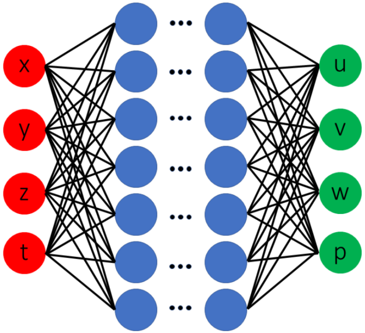
High-Dimensional PDE Handling

Classical methods:

- Struggle with curse of dimensionality
- Exponential growth in computational cost
- Limited to low-dimensional problems

Neural networks:

- Approximate high-dimensional operators efficiently
- Scale better with dimension
- Universal approximation properties



Scaling with dimension

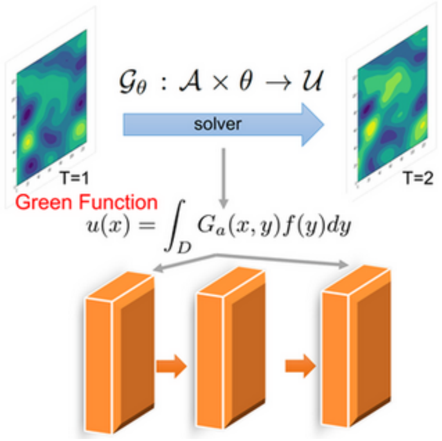
Computational Efficiency

Training Phase:

- One-time computational cost
- Learn PDE solution patterns
- Neural operator learning (FNO)

Inference Phase:

- Instantaneous solutions
- Ideal for many-query problems
- Optimization, inverse problems, UQ



$\mathcal{G}_\theta : \mathcal{A} \times \theta \to \mathcal{U}$
solver
T=1
T=2
Green Function
$u(x) = \int_D G_a(x,y) f(y) dy$

Neural Operator

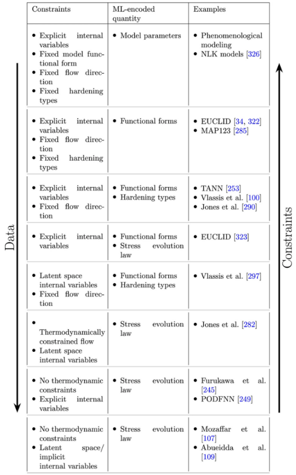
Data-Driven PDE Solving

Real-world challenges:

- Unknown or incomplete physics
- Turbulence modeling
- Climate modeling
- Complex material behavior

Deep learning advantages:

- Learn from experimental data
- Observational data integration
- Solutions where first-principles fail


Data
Constraints

Constraints	ML-encoded quantity	Examples
• Explicit internal variables • Fixed model functional form • Fixed flow direction • Fixed hardening types	• Model parameters	• Phenomenological modeling • NLK models [326]
• Explicit internal variables • Fixed flow direction • Fixed hardening types	• Functional forms	• EUCLID [34, 322] • MAP123 [285]
• Explicit internal variables • Fixed flow direction	• Functional forms • Hardening types	• TANN [253] • Vlassis et al. [100] • Jones et al. [290]
• Explicit internal variables	• Functional forms • Stress evolution law	• EUCLID [323]
• Latent space internal variables • Fixed flow direction	• Functional forms • Hardening types	• Vlassis et al. [297]
• Thermodynamically constrained flow • Latent space internal variables	• Stress evolution law	• Jones et al. [282]
• No thermodynamic constraints • Explicit internal variables	• Stress evolution law	• Furukawa et al. [245] • PODFNN [249]
• No thermodynamic constraints • Latent space/ implicit internal variables	• Stress evolution law	• Mozaffar et al. [107] • Abueidda et al. [109]

Data-driven modeling of constitutive laws

Table of Contents

Classical vs. Deep Learning Methods

Feature	Classical Methods	Deep Learning
Accuracy	High for well-resolved grids	Approximate, improves with training
Interpretability	High, first-principles physics	Lower, but improving
Computational Cost	Expensive for high-dim & many-query	Fast inference after training
High-Dimensional PDEs	Struggles beyond 3D	Efficient for high-dim
Inverse & UQ Problems	Many PDE solves, costly	Fast trained solutions
Data Dependency	No data needed	Can learn from data

When to Use Each Method?

Use Classical Methods when:

- High accuracy required
- Well-understood physics
- Low-dimensional problems
- Single or few PDE solves
- Interpretability crucial

Use Deep Learning when:

- High-dimensional PDEs
- Many-query problems
- Unknown/incomplete physics
- Real-time applications
- Large datasets available

Table of Contents

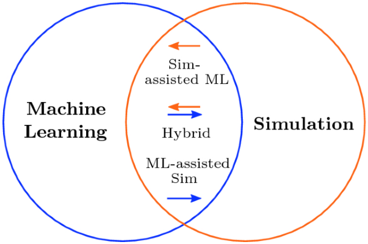
Hybrid Approaches

Combining the best of both worlds:

- Physics-informed deep learning
- Neural-enhanced classical solvers
- Multi-fidelity approaches
- Adaptive mesh refinement with ML

Emerging trends:

- Scientific machine learning
- Differentiable programming
- Neural operators



Hybrid classical-ML approach

Mini Game: Classical vs Neural Network?

Can you guess the right solver for each case?

1. **Parametric FEM is too slow**
   A company simulates heat transfer in engine blocks for different materials. Each FEM run takes 2 hours. They want quick temperature field predictions during design.
2. **No PDE, just data**
   A hospital collects data from patients with a rare lung disease. They have stress-strain curves but no clear mechanical model of the tissue.
3. **Safety-critical simulation**
   An aerospace firm needs accurate stress predictions on a turbine blade. These simulations inform certification and safety margins.

Mini Game: Classical vs Neural Network?

Can you guess the right solver for each case?

1. **Parametric FEM is too slow**
   A company simulates heat transfer in engine blocks for different materials. Each FEM run takes 2 hours. They want quick temperature field predictions during design.
2. **No PDE, just data**
   A hospital collects data from patients with a rare lung disease. They have stress-strain curves but no clear mechanical model of the tissue.
3. **Safety-critical simulation**
   An aerospace firm needs accurate stress predictions on a turbine blade. These simulations inform certification and safety margins.

Discussion:

- (1) $\rightarrow$ **Neural Network Surrogate** trained on offline FEM runs
- (2) $\rightarrow$ **Neural Network Model** fit to the data, no physics known
- (3) $\rightarrow$ **Classical FEM Solver** for reliability and interpretability

Conclusion

- Classical methods remain gold standard for accuracy and interpretability
- Deep learning excels in high-dimensional and many-query scenarios
- Method choice depends on specific problem requirements
- Future lies in hybrid approaches combining both strengths
- Scientific machine learning is rapidly evolving field

Thank you for your attention!