Introduction to Physics-Informed Neural Networks (PINNs)

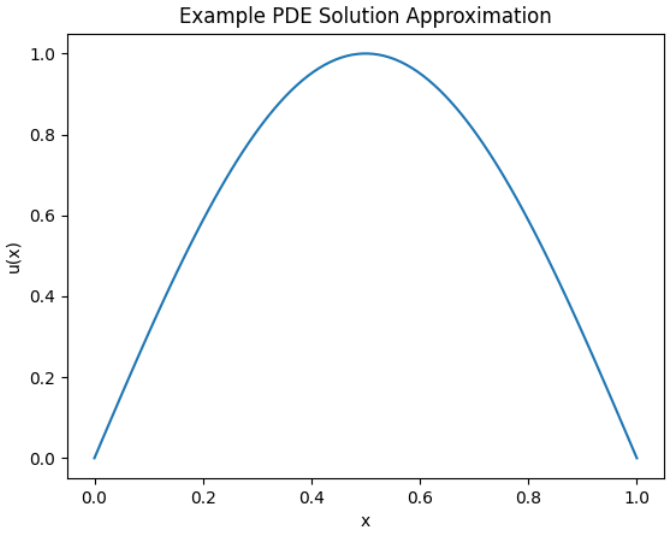
1. What are PINNs?

Physics-Informed Neural Networks (PINNs) are deep learning models that incorporate physical laws described by partial differential equations (PDEs) into the training process. Instead of relying only on data, PINNs enforce the governing equations as soft constraints in the loss function.

2. General PINN Formulation

Given a PDE: F(u, x, t) = 0 A neural network approximates u(x,t). Automatic differentiation is used to compute derivatives, and the loss function combines: - Data loss - PDE residual loss - Boundary/initial condition loss

Figure 1: Example Neural Network Approximation of a PDE Solution



3. Example: Solving 1D Poisson Equation

Consider the PDE: $d^2u/dx^2 = -\pi^2 \sin(\pi x)$, with boundary conditions u(0)=u(1)=0. Exact solution: u(x) = sin(πx).

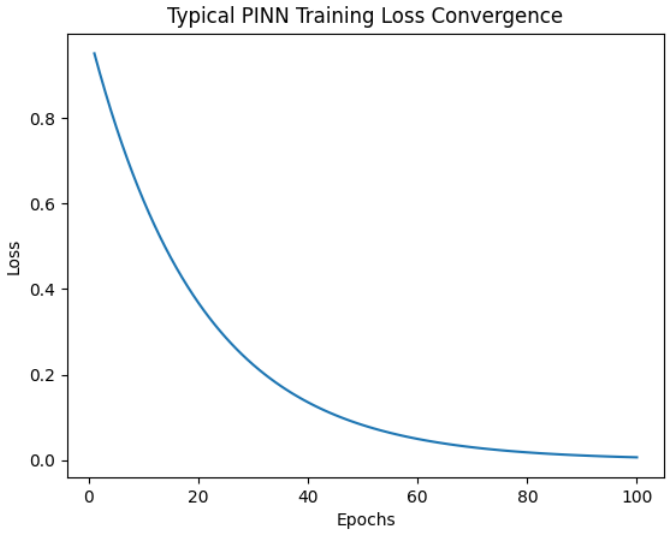
Example PyTorch Code:

```
import torch
import torch.nn as nn

class PINN(nn.Module):
    def __init__(self):
        super(PINN, self).__init__()
        self.net = nn.Sequential(
            nn.Linear(1, 20),
            nn.Tanh(),
            nn.Linear(20, 20),
            nn.Tanh(),
            nn.Linear(20, 1)
        )

    def forward(self, x):
        return self.net(x)

def pde_residual(model, x):
    x.requires_grad = True
    u = model(x)
    u_x = torch.autograd.grad(u, x, torch.ones_like(u), create_graph=True)[0]
    u_xx = torch.autograd.grad(u_x, x, torch.ones_like(u_x), create_graph=True)[0]
    return u_xx + (np.pi**2)*torch.sin(np.pi*x)
```

4. Loss Function Structure

Total Loss = L_data + L_PDE + L_BC Where: L_PDE = Mean squared PDE residual L_BC = Boundary condition loss L_data = Supervised data loss (if available)

Figure 2: Typical Loss Convergence in PINNs



5. Advantages and Applications

Advantages: - Works with small datasets - Incorporates physical laws directly - Solves forward and inverse problems Applications: - Fluid mechanics - Heat transfer - Structural mechanics - Electromagnetics