# Deep Learning for Partial Differential Equations in Economics

Benjamin Fan, Eddie Qiao
Mentored by Prof. Lu Lu

October 15, 2022
MIT PRIMES Conference

# Table of Contents

# What does operating a bank involve?

- People depositing money
- People taking loans
- Amount of workers
- Fixed costs (property tax, insurance)
- Many more factors

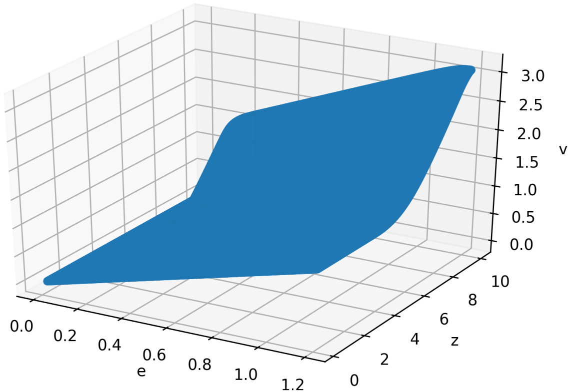
# Economic Modeling

- Make simplifications and assumptions
- Model these using *partial differential equations*

# Economic Modeling

- Make simplifications and assumptions
- Model these using *partial differential equations*
- For a bank, solve for the value function $v(e, z)$ ("how good" it is)
  - Equity $e$ (amount of shares sold)
  - Productivity $z$ (amount of deposits/loans made per worker)

# HJB Equation Solution




Graph of solution $v$ to HJB equation.

# Partial Differential Equations (PDEs) — Partial Derivatives

## Definition

A *partial derivative* is the derivative with respect to one variable for functions of several variables. We denote the derivative of $f$ with respect to $x$ to be $\frac{\partial f}{\partial x}$.

# Partial Differential Equations (PDEs) — Partial Derivatives

**Definition**

A *partial derivative* is the derivative with respect to one variable for functions of several variables. We denote the derivative of $f$ with respect to $x$ to be $\frac{\partial f}{\partial x}$.

**Example 1**

$\frac{\partial}{\partial x}(x^2 + 2xy + 3y) = 2x + 2y$

# Partial Differential Equations (PDEs) — Partial Derivatives

## Definition

A *partial derivative* is the derivative with respect to one variable for functions of several variables. We denote the derivative of $f$ with respect to $x$ to be $\frac{\partial f}{\partial x}$.

## Example 1

$$\frac{\partial}{\partial x}(x^2 + 2xy + 3y) = 2x + 2y$$

## Example 2

$$\frac{\partial^2}{\partial x \partial y}(x^4 + 2x^2y^2 + y) = \frac{\partial}{\partial x}(4x^2y + 1) = 8xy$$

# PDE Example — Heat Equation

## 1D Heat Equation

We have

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \quad x \in [0,1], \quad t \in [0,1]$$

where $\alpha = 0.4$ is the thermal diffusivity constant. Boundary condition and initial conditions:

$$u(0,t) = u(1,t) = 0, \quad u(x,0) = \sin(\pi x).$$

Models the diffusion of heat over time.

# PDEs — Formal Setup

Consider a PDE with solution $u(\mathbf{x}, t)$ for $\mathbf{x} = (x_1, \ldots, x_d)$ and parameters $\boldsymbol{\lambda}$ over the domain $\Omega \subset \mathbb{R}^d$:

$$f\left(\mathbf{x}; \frac{\partial u}{\partial x_1}, \ldots, \frac{\partial u}{\partial x_d}; \frac{\partial^2 u}{\partial x_1 \partial x_1}, \ldots, \frac{\partial^2 u}{\partial x_1 \partial x_d}; \ldots; \boldsymbol{\lambda}\right) = 0, \quad \mathbf{x} \in \Omega$$

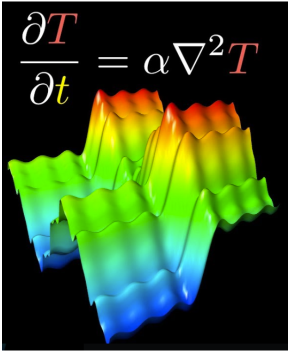
with boundary conditions

$$\mathcal{B}(u, \mathbf{x}) = 0 \quad \text{on} \quad \partial\Omega.$$

# Table of Contents

# Solving PDEs

PDEs are difficult to solve — think differentiating vs. integrating



# Deep Learning — FNN Formal Setup

**Definition**

An $L$-layer *feed-forward neural network* (FNN) is a function $\mathcal{N}^L(\mathbf{x})\colon \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$. For each layer, we define a weight matrix $\mathbf{W}^\ell$ and a bias vector $\mathbf{b}^\ell$. Then, letting $T^\ell(\mathbf{x}) = \mathbf{W}^\ell \mathbf{x} + \mathbf{b}^\ell$ and $\sigma$ be a non-linear *activation function*, we define

$$\mathcal{N}^L(\mathbf{x}) = T^L \circ \sigma \circ T^{L-1} \circ \ldots \circ \sigma \circ T^1(\mathbf{x}).$$

# Deep Learning — FNN Formal Setup

## Definition

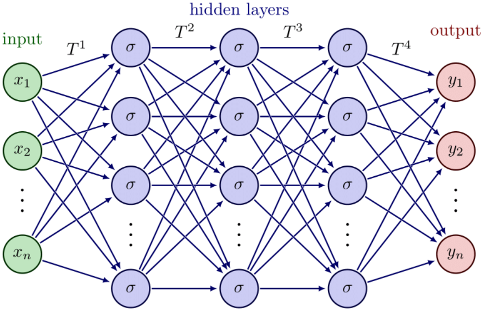
An $L$-layer *feed-forward neural network* (FNN) is a function $\mathcal{N}^L(\mathbf{x})\colon \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$. For each layer, we define a weight matrix $\mathbf{W}^\ell$ and a bias vector $\mathbf{b}^\ell$. Then, letting $T^\ell(\mathbf{x}) = \mathbf{W}^\ell \mathbf{x} + \mathbf{b}^\ell$ and $\sigma$ be a non-linear *activation function*, we define

$$\mathcal{N}^L(\mathbf{x}) = T^L \circ \sigma \circ T^{L-1} \circ \ldots \circ \sigma \circ T^1(\mathbf{x}).$$

Note: We can formalize this by specifying the dimensions of $\mathbf{W}^\ell$ and $\mathbf{b}^\ell$.

# Deep Learning — Unpacking the Definition

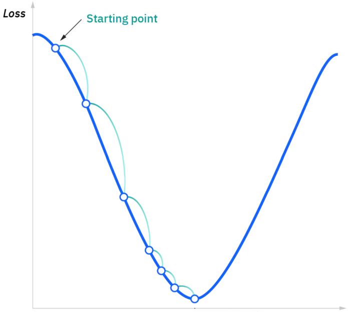
$$\mathcal{N}^L(\mathbf{x}) = T^L \circ \sigma \circ T^{L-1} \circ \ldots \circ \sigma \circ T^1(\mathbf{x}).$$




Visualization with $L = 4$

# Deep Learning — Training

The *loss function* represents how good the solution is.




When training, we aim to minimize the loss function. We use Adam and L-BFGS.

# Physics-Informed Neural Networks (PINNs)

Physics-Informed Neural Networks get their name from using information from physics to aid the model.

- Equations such as conservation laws
- Basic idea: we embed a PDE into the loss function
- As the loss function gets closer to zero, the model increases in accuracy

# PINN Details

First, we define the training set: we have $\mathcal{T}_f$ points inside the domain and $\mathcal{T}_b$ points on the boundary. Furthermore, we let:

$$\mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f) = \frac{1}{|\mathcal{T}_f|} \sum_{\mathbf{x} \in \mathcal{T}_f} \left\| f\left(\mathbf{x}; \frac{\partial \hat{u}}{\partial x_1}, \ldots, \frac{\partial \hat{u}}{\partial x_d}; \frac{\partial^2 \hat{u}}{\partial x_1 \partial x_1}, \ldots, \frac{\partial^2 \hat{u}}{\partial x_1 \partial x_d}; \ldots; \boldsymbol{\lambda}\right) \right\|_2^2,$$

$$\mathcal{L}_b(\boldsymbol{\theta}, \mathcal{T}_b) = \frac{1}{|\mathcal{T}_b|} \sum_{\mathbf{x} \in \mathcal{T}_b} \|\mathcal{B}(\hat{u}, \mathbf{x})\|_2^2.$$

$\mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f)$ is the $L^2$ mean of the PDE residuals and $\mathcal{L}_b(\boldsymbol{\theta}, \mathcal{T}_b)$ is the $L^2$ mean of the errors for boundary points with the boundary condition.

# PINN Details

First, we define the training set: we have $\mathcal{T}_f$ points inside the domain and $\mathcal{T}_b$ points on the boundary. Furthermore, we let:

$$\mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f) = \frac{1}{|\mathcal{T}_f|} \sum_{\mathbf{x} \in \mathcal{T}_f} \left\| f\left(\mathbf{x}; \frac{\partial \hat{u}}{\partial x_1}, \dots, \frac{\partial \hat{u}}{\partial x_d}; \frac{\partial^2 \hat{u}}{\partial x_1 \partial x_1}, \dots, \frac{\partial^2 \hat{u}}{\partial x_1 \partial x_d}; \dots; \boldsymbol{\lambda}\right) \right\|_2^2,$$

$$\mathcal{L}_b(\boldsymbol{\theta}, \mathcal{T}_b) = \frac{1}{|\mathcal{T}_b|} \sum_{\mathbf{x} \in \mathcal{T}_b} \|\mathcal{B}(\hat{u}, \mathbf{x})\|_2^2.$$

$\mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f)$ is the $L^2$ mean of the PDE residuals and $\mathcal{L}_b(\boldsymbol{\theta}, \mathcal{T}_b)$ is the $L^2$ mean of the errors for boundary points with the boundary condition.

The loss function is

$$\mathcal{L}(\boldsymbol{\theta}, \mathcal{T}) = w_f \mathcal{L}_f(\boldsymbol{\theta}; \mathcal{T}_f) + w_b \mathcal{L}_b(\boldsymbol{\theta}; \mathcal{T}_b).$$

$w_f$ and $w_b$ are loss weights.

# PINNs in Action: Solving the Heat Equation

We use the python library DeepXDE to implement PINNs for solving PDEs. Recall the 1D Heat Equation

## 1D Heat Equation

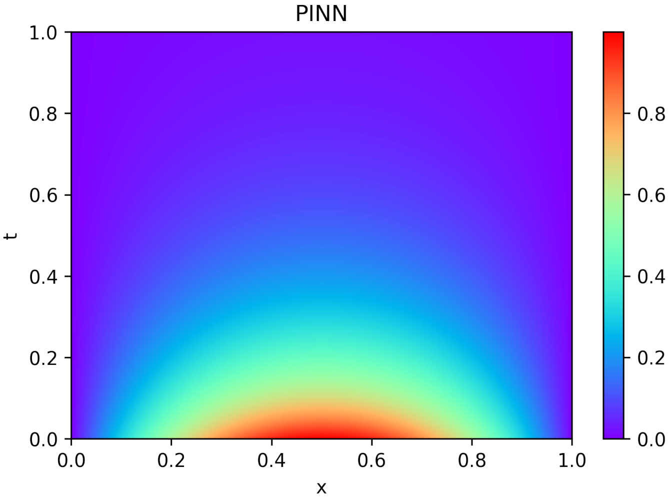
We have

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}, \quad x \in [0,1], \quad t \in [0,1]$$

where $\alpha = 0.4$ is the thermal diffusivity constant. Boundary condition and initial conditions:

$$u(0,t) = u(1,t) = 0, \quad u(x,0) = \sin(\pi x).$$

# PINNs in Action: 1D Heat Equation Results



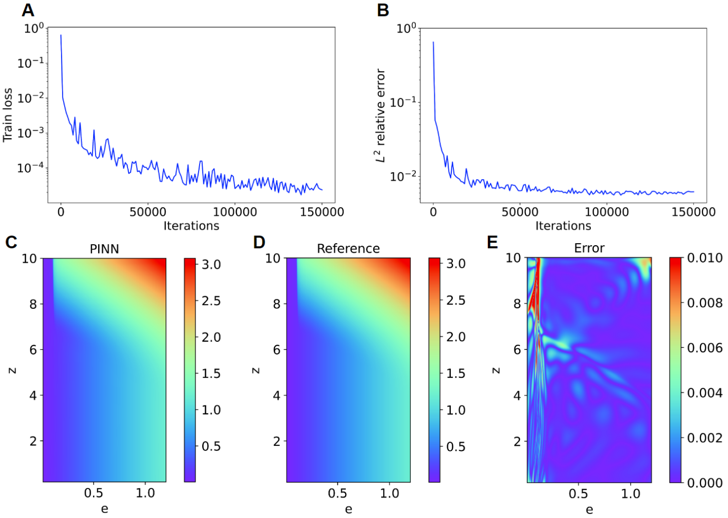
# Table of Contents

# Applications to Other Fields

PINNs can be applied to many other fields:

- Physics
- Systems biology
- Biochemistry
- Optics
- Economics

# PINNs for Economic Modeling — HJB Equation Results



## Acknowledgements

Many thanks to

- Our mentor, Prof. Lu Lu
- Prof. Wenhao Li, Zhouzhou Gu, Anran Jiao
- MIT PRIMES: Dr. Tanya Khovanova, Prof. Pavel Etingof, Dr. Slava Gerovitch
- Our families

# References


[1] *But what is a partial differential equation? — DE2*. YouTube, Apr. 2019. URL: https://www.youtube.com/watch?v=ly4S0oi3Yz8&ab_channel=3Blue1Brown.

[2] IBM Cloud Education. *What is gradient descent?* Oct. 2020. URL: https://www.ibm.com/cloud/learn/gradient-descent.

[3] L. Lu et al. "DeepXDE: A deep learning library for solving differential equations". In: *SIAM Review* 63.1 (2021), pp. 208–228.