Basics of Neural Networks and Their Application in Solving Differential Equations

MA5260 : Seminar

MD Karimulla Haque

Guided by,
Dr. Sivaram Ambikasaran

November 8, 2024

Introduction

- Neural networks serve as the foundational architecture of Deep Learning.
- Their structure and operation are inspired by the biological neurons found in the human brain, hence the term 'neural'.
- This interconnected structure allows neural networks to learn complex patterns and relationships in the data.

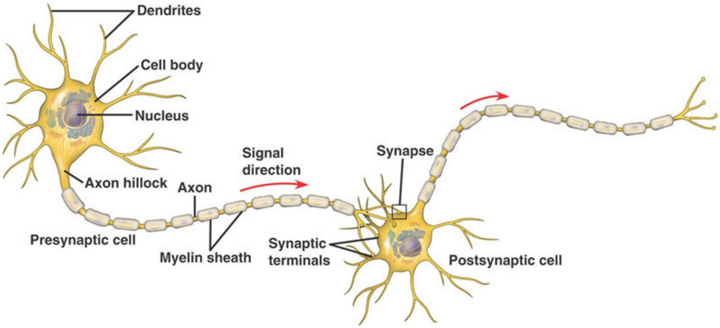


Figure: Biological Neurons

What is an ANN?

- **Structure:** Composed of neurons (nodes) arranged in layers (input, hidden, and output).
- **Layers:**
  - **Input Layer:** Takes input data
  - **Hidden Layers:** Intermediate layers that process inputs through weights and biases.
  - **Output Layer:** Produces the final output.
- **Activation Functions:** Functions applied to the weighted sum of inputs to introduce non-linearity (e.g., ReLU, sigmoid).

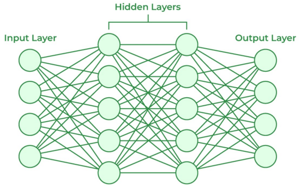


Figure: Architecture of ANN

Basic Working Principle

- ANNs learn to map input to output by adjusting weights and biases through training.

Components of NN

- Neurons (Nodes)
- Weights
- Biases
- Activation Function
- Loss Function
- Gradient Descent
- Learning Rate

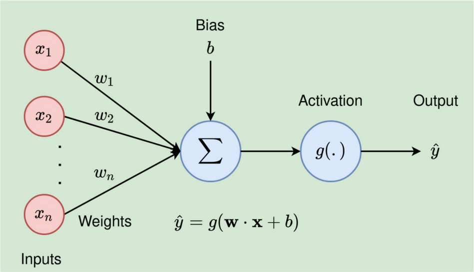
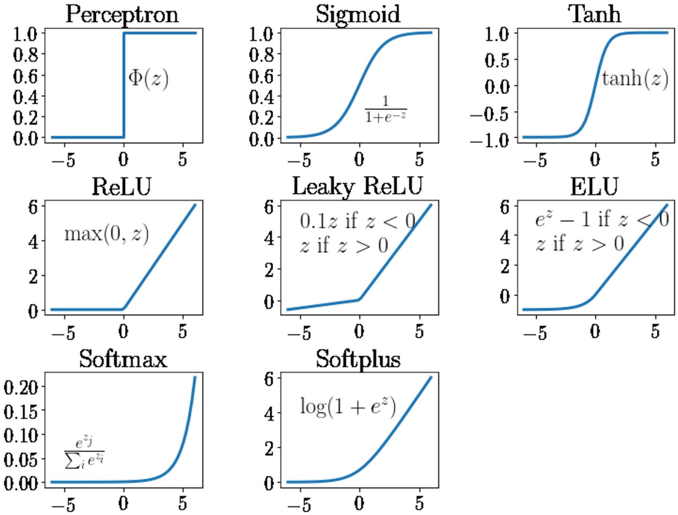


Figure: Components of ANN

Activation Functions



Loss Function

- A loss function, also known as a cost function or objective function, measures the difference between the predicted outputs of a neural network and the actual target values.
- It is essentially a quantification of the error in value estimation.
- Minimizing the loss function results in high prediction accuracy.

$$\frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$$

Figure: Mean Square Error

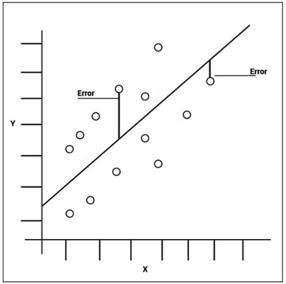
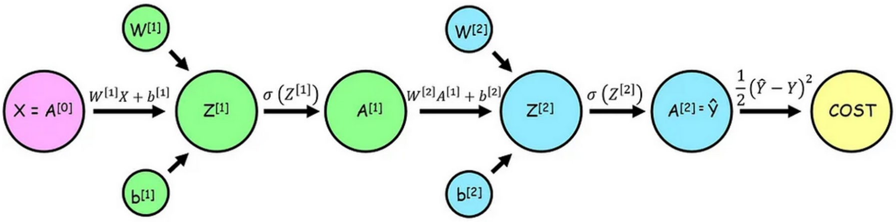


Figure: Linear Regression

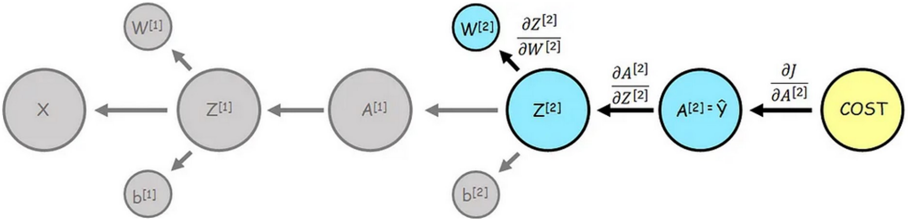
Forward Propagation



Calculations for Back Propagation

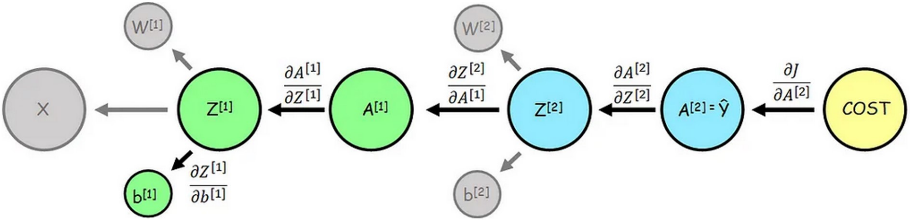
Original function:	Partial derivative:
$J = \frac{1}{2}\left(A^{[2]} - Y\right)^2$	$\frac{\partial J}{\partial A^{[2]}} = A^{[2]} - Y$
$A^{[2]} = \sigma\left(Z^{[2]}\right) = \frac{1}{1+e^{-Z^{[2]}}}$	$\frac{\partial A^{[2]}}{\partial Z^{[2]}} = A^{[2]}\left(1 - A^{[2]}\right)$
$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]}$	$\frac{\partial Z^{[2]}}{\partial W^{[2]}} = A^{[1]}$

Back Propagation for Weight



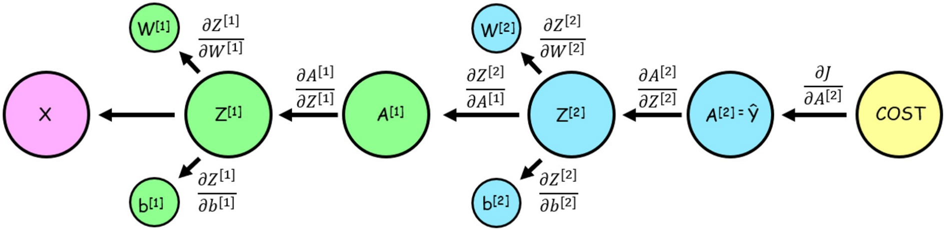
$$\frac{\partial J}{\partial W^{[2]}} = \frac{\partial J}{\partial A^{[2]}} \frac{\partial A^{[2]}}{\partial Z^{[2]}} \frac{\partial Z^{[2]}}{\partial W^{[2]}} = (A^{[2]} - Y) \bullet A^{[2]}(1 - A^{[2]}) \bullet A^{[1]}$$

Back Propagation for Bias



$$\frac{\partial J}{\partial b^{[1]}} = \underbrace{\frac{\partial J}{\partial A^{[2]}} \frac{\partial A^{[2]}}{\partial Z^{[2]}}}_{\text{Previously calculated}} \frac{\partial Z^{[2]}}{\partial A^{[1]}} \frac{\partial A^{[1]}}{\partial Z^{[1]}} \frac{\partial Z^{[1]}}{\partial b^{[1]}} = (A^{[2]} - Y) \bullet A^{[2]}(1 - A^{[2]}) \bullet W^{[2]} \bullet A^{[1]}(1 - A^{[1]}) \bullet 1$$

Back Propagation

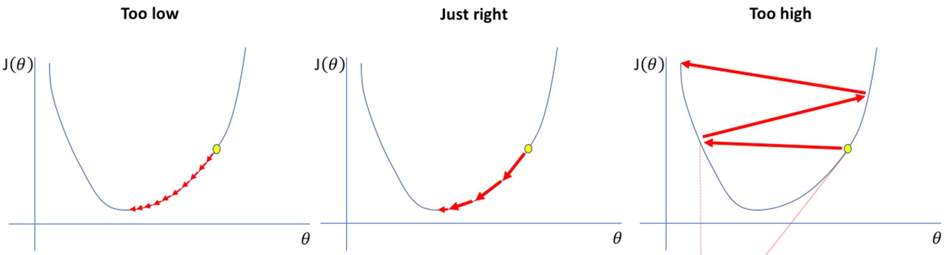


Gradient Descent

$$W_{new}^{[1]} = W_{old}^{[1]} - \alpha \frac{dJ}{dW^{[1]}} \quad | \quad b_{new}^{[1]} = b_{old}^{[1]} - \alpha \frac{dJ}{db^{[1]}}$$

$$W_{new}^{[2]} = W_{old}^{[2]} - \alpha \frac{dJ}{dW^{[2]}} \quad | \quad b_{new}^{[2]} = b_{old}^{[2]} - \alpha \frac{dJ}{db^{[2]}}$$

Learning Rate

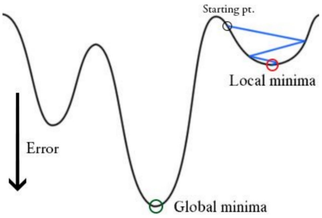


A small learning rate requires many updates before reaching the minimum point

The optimal learning rate swiftly reaches the minimum point

Too large of a learning rate causes drastic updates which lead to divergent behaviors

Non-Convex



(Loading Video)

- Initialisation
- Forward propogation
- Back propogation
- Gradient Descent

(Loading Video)

- **Analytical Methods**
  - **Examples**
    - **Separation of Variables:** Technique to solve differential equations by separating the variables and integrating.
    - **Integrating Factors:** Method to solve linear first-order differential equations by multiplying by an integrating factor.
- **Numerical Methods**
  - **Examples**
    - **Euler's Method:** Simple numerical procedure for solving ODEs by approximating solutions at discrete points.
    - **Runge-Kutta Methods:** More accurate numerical methods for solving ODEs.
    - **Finite Difference Method:** Numerical technique for solving PDEs by approximating derivatives with finite differences.

Why Use ANN for ODE and PDE?

- **Advantages**
  - **Flexibility:** The method is general and can be applied to single ODE, system of ODE's and PDE defined on orthogonal box boundaries.
  - **Parallel Processing:** The method can also be efficiently implemented on parallel architecture.
  - **Handling Complexity:** The required number of model parameters is far less than any other solution technique.
  - **Closed Form:** An NN-based solution of a DE is differentiable and is in a closed analytic form that can be used in subsequent calculations.

Paper Present

- We will now present the implementation of the paper ANN for solving ODE and PDE by Lagaris, I.E., Likas, A. and Fotiadis, D.I. | IEEE Journals Magazine | IEEE Xplore. Available at: https://ieeexplore.ieee.org/document/712178.

Methodology

- ANNs can be used to approximate solutions to differential equations.
- A trial solution is written as a sum of two parts:
  - The first part satisfies the boundary or initial conditions and contains no adjustable parameters.
  - The second part involves a feedforward neural network with adjustable weights.
- By construction, the boundary conditions are satisfied, and the network is trained to satisfy the differential equation.
- Applicability ranges from single ODEs to systems of coupled ODEs and PDEs.

Implementation

- Assume a trial form of the solution: $\Psi_t(\vec{x}) = A(\vec{x}) + F(\vec{x}, N(\vec{x}, \vec{p}))$
- $N(\vec{x}, \vec{p})$ is the feedforward NN with parameters $\vec{p}$.
- The second term F is constructed so as not to contribute to the BC's.
- Learn the parameters to approximately solve the differential equation.

Description of the Methods

- Consider general differential equation: $G(\vec{x}, \Psi(\vec{x}), \nabla\Psi(\vec{x}), \nabla^2\Psi(\vec{x})) = 0$ where $\vec{x} \in D$
- Now we will use collocation method to solve the above differential equation i.e.,

$$G(\vec{x}_i, \Psi(\vec{x}_i, \vec{p}), \nabla\Psi(\vec{x}_i, \vec{p}), \nabla^2\Psi(\vec{x}_i, \vec{p})) = 0, \quad \forall \vec{x}_i \in \hat{D}$$

- Now we need to calculate

$$\min_{\vec{p}} \sum_{\vec{x}_i \in \hat{D}} (G(\vec{x}_i, \Psi_t(\vec{x}_i, \vec{p}), \nabla\Psi_t(\vec{x}_i, \vec{p}), \nabla^2\Psi_t(\vec{x}_i, \vec{p})))^2$$

- Construct a trial solution which satisfy BC(s) as

$$\Psi_t(\vec{x}) = A(\vec{x}) + F(\vec{x}, N(\vec{x}, \vec{p}))$$

where we will choose $A(\vec{x})$ as it satisfies boundary conditions and F as not to contribute to the BC(s).

NN Model

- Input in i th hidden unit is $z_i = \sum_{j=1}^{n} w_{ij} x_j + u_i$
- Output in i th hidden unit is $\sigma(z_i)$
- Final output of the NN is $N = \sum_{i=1}^{H} v_i \sigma(z_i)$
- w_{ij} denotes the weight from the input unit j to the hidden unit i, v_i denotes the weight from the hidden unit i to the output, u_i denotes the bias of hidden unit i and $\sigma(z)$ is the sigmoid function.

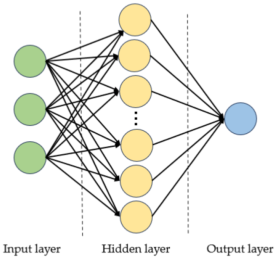


Figure: 3 input units, one hidden layer with H sigmoid umits and a linear output unit

Method for first-order ODE

- Consider the first order ODE

$$\frac{d\Psi(x)}{dx} = f(x, \Psi)$$

with $x \in [0, 1]$ and the IC $\Psi(0) = A$

- A trial solution is

$$\Psi_t(x) = A + xN(x, \vec{p})$$

where $N(x, \vec{p})$ is the output of a feedforward NN with one input unit for x and weights $\vec{p}$

- Therefore,

$$\frac{d\Psi_t(x)}{dx} = N(x, \vec{p}) + x\frac{dN(x, \vec{p})}{dx}$$

- We need to minimized the error quantity,

$$E[\vec{p}] = \sum_i \left\{ \frac{d\Psi_t(x_i)}{dx} - f(x_i, \Psi_t(x_i)) \right\}^2, \quad x_i \in [0, 1]$$

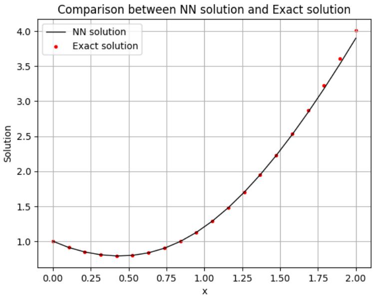
Problem 1 with IVP

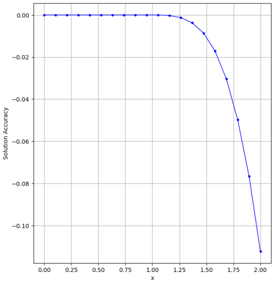
- Solve the Single Ordinary Differential Equation

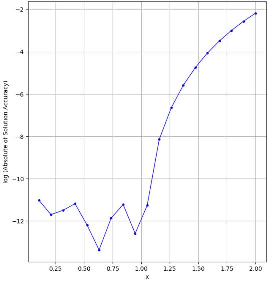
$$\frac{d\Psi}{dx} + \left(x + \frac{1+3x^2}{1+x+x^3}\right)\Psi = x^3 + 2x + x^2\frac{1+3x^2}{1+x+x^3} \text{ with } \Psi(0) = 1 \text{ and } x \in [0,1].$$

The analytical solution is $\Psi_a(x) = \dfrac{e^{\frac{x^2}{2}}}{1+x+x^3} + x^2$

The trial solution is $\Psi_t(x) = 1 + xN(x, \vec{p})$

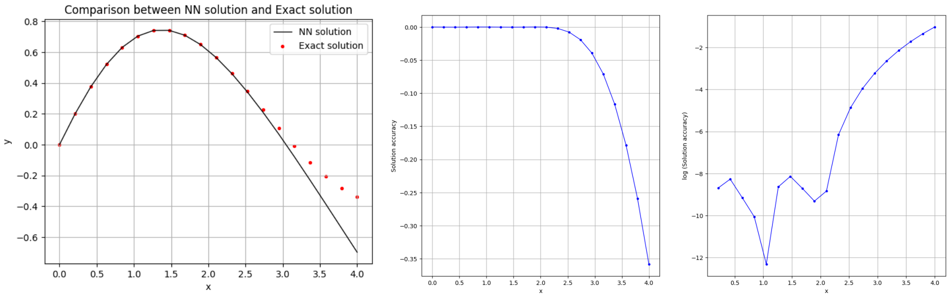






Problem 2 with IVP

- Solve the Single Ordinary Differential Equation $\frac{d\Psi}{dx} + \frac{1}{5}\Psi = e^{-\left(\frac{x}{5}\right)}cos(x)$ with $\Psi(0) = 0$ and $x \in [0, 2]$.
  The analytical solution is $\Psi_a(x) = e^{-\left(\frac{x}{5}\right)}sin(x)$
  The trial solution is $\Psi_t(x) = xN(x, \vec{p})$



Method for second-order ODE

- Consider the second order ODE

$$\frac{d^2\Psi(x)}{dx^2} = f\left(x, \Psi, \frac{d\Psi(x)}{dx}\right)$$

with $x \in [0, 1]$

- A trial solution for the initial conditions $\Psi(0) = A$ and $\left(\frac{d}{dx}\right)\Psi(0) = A'$ is

$$\Psi_t(x) = A + A'x + x^2 N(x, \vec{p})$$

where $N(x, \vec{p})$ is the output of a feedforward NN with one input unit for x and weights $\vec{p}$

- A trial solution for the two point Dirichlet BC $\Psi(0) = A$ and $\Psi(1) = B$ is

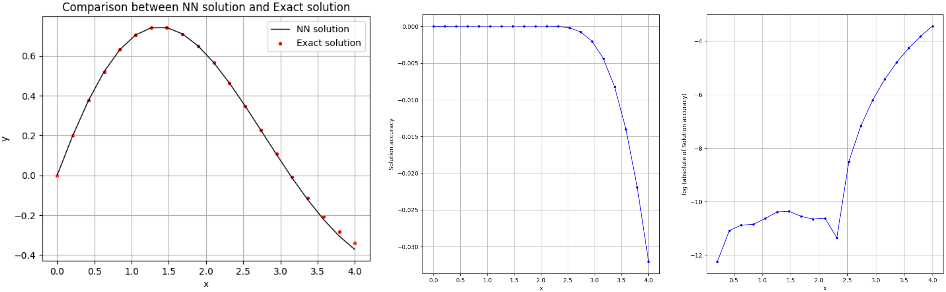
$$\Psi_t(x) = A(1-x) + Bx + x(1-x)N(x, \vec{p})$$

- We need to minimized the error quantity for both cases,

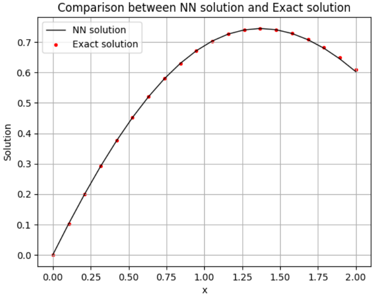
$$E[\vec{p}] = \sum_i \left\{ \frac{d^2\Psi_t(x_i)}{dx^2} - f\left(x_i, \Psi_t(x_i), \frac{d\Psi_t(x_i)}{dx}\right)\right\}^2, \quad x_i \in [0, 1]$$

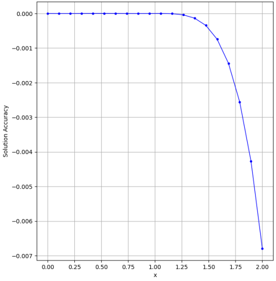
Problem 3 with IVP

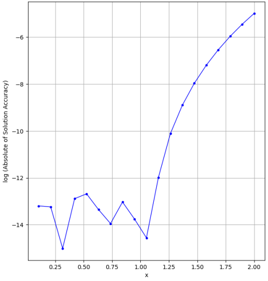
- Given the differential equation $\frac{d^2\Psi}{dx^2} + \frac{1}{5}\frac{d\Psi}{dx} + \Psi = -\frac{1}{5}e^{\frac{-x}{5}}cos(x)$ with $\Psi(0) = 0, \ \left(\frac{d}{dx}\right)\Psi(0) = 1$ and $x \in [0, 2]$.
  The analytical solution is $\Psi(x) = e^{-\left(\frac{x}{5}\right)}sin(x)$
  The trial solution is $\Psi_t(x) = x + x^2 N(x, \vec{p})$



- Given the differential equation $\frac{d^2\Psi}{dx^2} + \frac{1}{5}\frac{d\Psi}{dx} + \Psi = -\frac{1}{5}e^{\frac{-x}{5}}cos(x)$ with $\Psi(0) = 0,\ \ \Psi(1) = sin(1)e^{-(\frac{1}{5})}$ and $x \in [0, 1]$.
  The analytical solution is $\Psi(x) = e^{-(\frac{x}{5})}sin(x)$
  The trial solution is: $\Psi_t(x) = xsin(1)e^{-(\frac{1}{5})} + x(1-x)N(x, \vec{p})$







Method for systems of K first-order ODEs

- Consider the systems of first order ODEs

$$\frac{d\Psi_i}{dx} = f_i(x, \Psi_1, \Psi_2, \cdots, \Psi_K)$$

with $i = 1, \cdots, K$ and the ICs $\Psi_i(0) = A_i$

- A trial solution is

$$\Psi_{t_i}(x) = A_i + x N_i(x, \vec{p}_i)$$

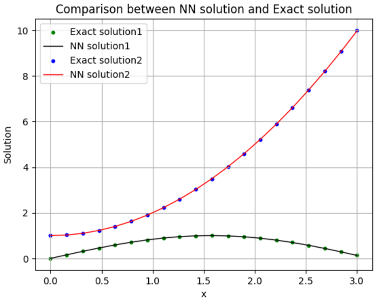
where $N_i(x, \vec{p}_i)$ is the output of a feedforward NN with one input unit for x and weights $\vec{p}_i$ for $i = 1, \cdots, K$

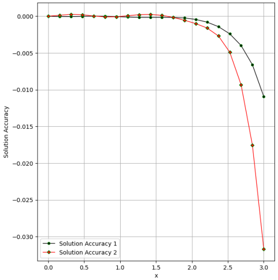
- We need to minimized the error quantity,

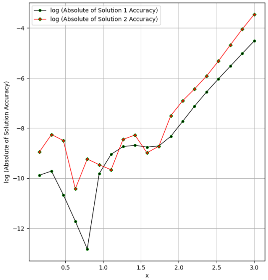
$$E[\vec{p}] = \sum_{k=1}^{K} \sum_{i} \left\{ \frac{d\Psi_{t_k}(x_i)}{dx} - f_k(x_i, \Psi_{t_1}, \Psi_{t_2}, \cdots, \Psi_{t_K}) \right\}^2, \qquad x_i \in [0, a]$$

Problem 4 with IVP

- Consider the system of two coupled first order Ordinary Differential Equations $\frac{d\Psi_1}{dx} = cos(x) + \Psi_1^2 + \Psi_2 - (1 + x^2 + sin^2(x))$
  $\frac{d\Psi_2}{dx} = 2x - (1 + x^2)sin(x) + \Psi_1\Psi_2$ with $\Psi_1(0) = 0, \;\; \Psi_2(0) = 1$ and $x \in [0, 3]$.
  The analytical solutions are $\Psi_{a_1}(x) = sin(x), \;\; \Psi_{a_2}(x) = 1 + x^2$
  The trial solutions are $\Psi_{t_1}(x) = xN_1(x, \vec{p_1}), \;\; \Psi_{t_2}(x) = 1 + xN_2(x, \vec{p_2})$







Method for Single PDE with Dirichlet BC

- Consider only two-dimensional problems. For example, the Poisson equation

$$\frac{\partial^2 \Psi(x,y)}{\partial x^2} + \frac{\partial^2 \Psi(x,y)}{\partial y^2} = f(x,y), \qquad (x,y) \in [0,1] \times [0,1]$$

  with Dirichlet boundary conditions
  $\Psi(0,y) = f_0(y),\ \Psi(1,y) = f_1(y),\ \Psi(x,0) = g_0(x),\ \Psi(x,1) = g_1(x)$

- A trial solution is

$$\Psi_t(x,y) = A(x,y) + x(1-x)y(1-y)N(x,y,\vec{p})$$

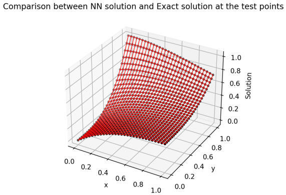
  where $A(x,y) = (1-x)f_0(y) + xf_1(y) + (1-y)\{g_0(x) - [(1-x)g_0(0) + xg_0(1)]\} + y\{g_1(x) - [(1-x)g_1(0) + xg_1(1)]\}$ and $N(x,y,\vec{p})$ is the output of a feedforward NN with two input units for x, y and weights $\vec{p}$

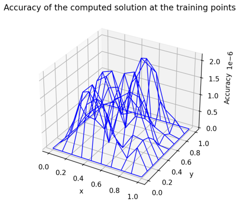
- We need to minimized the error quantity,

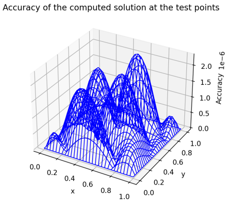
$$E[\vec{p}] = \sum_i \left\{ \frac{\partial^2 \Psi(x_i,y_i)}{\partial x^2} + \frac{\partial^2 \Psi(x_i,y_i)}{\partial y^2} - f(x_i,y_i) \right\}^2, \qquad (x_i,y_i) \in [0,1]\times[0,1]$$

Problem 5 with Dirichlet BC

- Solve the Partial Differential Equation $\nabla^2\Psi(x,y) = e^{-x}(x-2+y^3+6y)$ i.e., $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = e^{-x}(x-2+y^3+6y)$ with $x, y \in [0,1]$ and with Dirichlet boundary conditions
$\Psi(0,y) = y^3,\;\; \Psi(1,y) = \frac{(1+y^3)}{e},\;\; \Psi(x,0) = xe^{-x},\; \Psi(x,1) = e^{-x}(1+x)$
The analytical solution is $\Psi_a(x,y) = e^{-x}(x+y^3)$
The trial solution is $\Psi_t(x,y) = A(x,y) + x(1-x)y(1-y)N(x,y,\vec{p})$







Method for Single PDE with mixed BC

- Again for example, consider the Poisson equation

$$\frac{\partial^2\Psi(x,y)}{\partial x^2}+\frac{\partial^2\Psi(x,y)}{\partial y^2}=f(x,y),\qquad (x,y)\in[0,1]\times[0,1]$$

with mixed boundary conditions

$\Psi(0,y)=f_0(y),\ \Psi(1,y)=f_1(y),\ \Psi(x,0)=g_0(x),\ (\frac{\partial\Psi(x,1)}{\partial y})=g_1(x)$

- A trial solution is

$$\Psi_t(x,y)=B(x,y)+x(1-x)y\left[N(x,y,\vec{p})-N(x,1,\vec{p})-\frac{\partial N(x,1,\vec{p})}{\partial y}\right]$$

where $B(x,y)=(1-x)f_0(y)+xf_1(y)+g_0(x)-[(1-x)g_0(0)+xg_0(1)]+y\{g_1(x)-[(1-x)g_1(0)+xg_1(1)]\}$ and $N(x,y,\vec{p})$ is the output of a feedforward NN with two input units for x,y and weights $\vec{p}$

- We need to minimized the error quantity,

$$E[\vec{p}]=\sum_i\left\{\frac{\partial^2\Psi(x_i,y_i)}{\partial x^2}+\frac{\partial^2\Psi(x_i,y_i)}{\partial y^2}-f(x_i,y_i)\right\}^2,\qquad (x_i,y_i)\in[0,1]\times[0,1]$$

Problem 6 with mixed BC

- Solve the Partial Differential Equation $\nabla^2\Psi(x,y) = (2-\pi^2y^2)sin(\pi x)$ i.e., $\frac{\partial^2\Psi}{\partial x^2} + \frac{\partial^2\Psi}{\partial y^2} = (2-\pi^2y^2)sin(\pi x)$ with $x, y \in [0,1]$ and with mixed boundary conditions

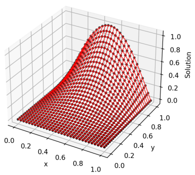
  $\Psi(0,y) = 0,\ \Psi(1,y) = 0,\ \Psi(x,0) = 0,\ \frac{\partial}{\partial y}\Psi(x,1) = 2sin(\pi x)$

  The analytic solution is $\Psi_a(x,y) = y^2 sin(\pi x)$

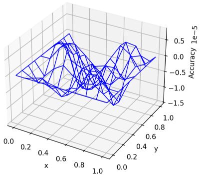
  The trial solution is

  $$\Psi_t(x,y) = B(x,y) + x(1-x)y\left[N(x,y,\vec{p}) - N(x,1,\vec{p}) - \frac{\partial N(x,1,\vec{p})}{\partial y}\right]$$

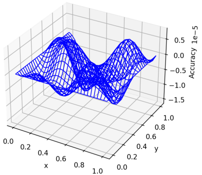
Comparison between NN solution and Exact solution at the test points



Accuracy of the computed solution at the training points



Accuracy of the computed solution at the test points



Problem 7 with mixed BC

- Solve the Non-linear Partial Differential Equation

$$\nabla^2\Psi(x,y) + \Psi(x,y)\frac{\partial}{\partial y}\Psi(x,y) = sin(\pi x)(2 - \pi^2 y^2 + 2y^3 sin(\pi x))$$

i.e., $\frac{\partial^2\Psi}{\partial x^2} + \frac{\partial^2\Psi}{\partial y^2} + \Psi\frac{\partial\Psi}{\partial y} = sin(\pi x)(2 - \pi^2 y^2 + 2y^3 sin(\pi x))$ with $x, y \in [0, 1]$

and with mixed boundary conditions

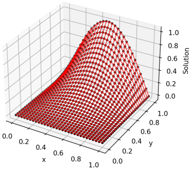
$$\Psi(0,y) = 0,\ \Psi(1,y) = 0, \Psi(x,0) = 0,\ \frac{\partial}{\partial y}\Psi(x,1) = 2sin(\pi x)$$

The analytic solution is $\Psi_a(x,y) = y^2 sin(\pi x)$

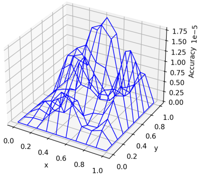
The trial solution is

$$\Psi_t(x,y) = B(x,y) + x(1-x)y\left[N(x,y,\vec{p}) - N(x,1,\vec{p}) - \frac{\partial N(x,1,\vec{p})}{\partial y}\right]$$

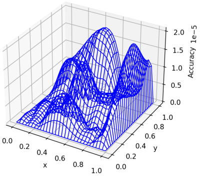
Comparison between NN solution and Exact solution at the test points



Accuracy of the computed solution at the training points



Accuracy of the computed solution at the test points



References

- Lagaris, I.E., Likas, A. and Fotiadis, D.I. Artificial neural networks for solving ordinary and partial differential equations | IEEE Journals Magazine | IEEE Xplore. available at: `https://ieeexplore.ieee.org/document/712178`
- Jorge Nocedal and Stephen J. Wright - Numerical Optimization (Springer Series in Operations Research)
- Christopher M. Bishop - Pattern Recognition and Machine Learning (Springer)
- Rashid, T. (2017) Make your own neural network: A gentle journey through the mathematics of Neural Networks, and making your own using the python computer language. United States: CreateSpace Independent Publishing.
- Pictures were taken from Google Images.
- The plots were generated using the implementation available at: `https://github.com/mdkarimullahaque/ANN_ODE_PDE`

Thank You