Motivation

- Traditional structural analysis methods, such as finite element analysis (FEA) and numerical methods, have been widely used in engineering disciplines for decades.

Method	Pros	Cons
Finite Element Method (FEM)	Accurate, well-established, and can handle complex geometries.	Can be computationally expensive, requires fine mesh for accurate results, may not be suitable for certain types of problems.
Numerical Methods	Can handle a wide range of problems, relatively easy to implement and computationally efficient.	May not be as accurate as FEM, may require a lot of computational resources for complex problems.

Problem Definition

- However, these methods have certain limitations that motivate the exploration of alternative approaches like Physics Informed Neural Networks (PINN) in structural applications.

- PINN addresses these motivations by leveraging the strengths of neural networks and physics-based modeling, providing a promising solution for accurate, efficient, and versatile structural analysis, design, and optimization.

- Physics-Informed Neural Networks (PINNs) are called "physics-informed" because they incorporate and leverage the underlying physics and governing Differential equations of the problem being solved. Unlike traditional neural networks that are purely data-driven, PINNs combine the power of neural networks with the knowledge of physical laws and constraints.

Introduction to Artificial Neural Network

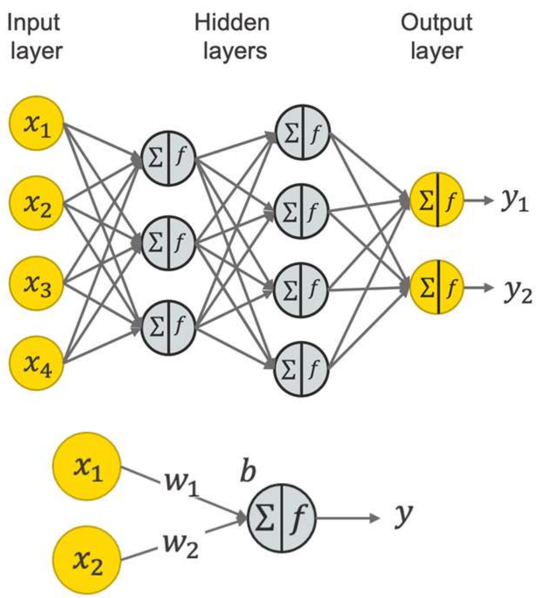

Fig 1. **NN Architecture**

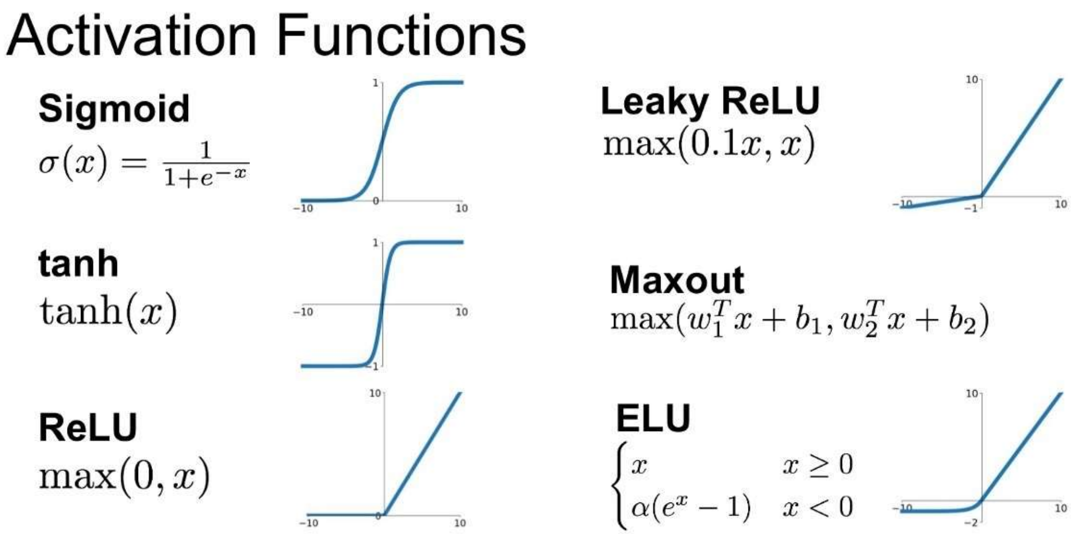

Fig 2. **Different Activation Functions used in ANN**

Fig 1 : https://www.knime.com/blog/a-friendly-introduction-to-deep-neural-networks
Fig 2: https://www.quora.com/What-is-the-difference-between-an-activation-function

Introduction to Artificial Neural Network

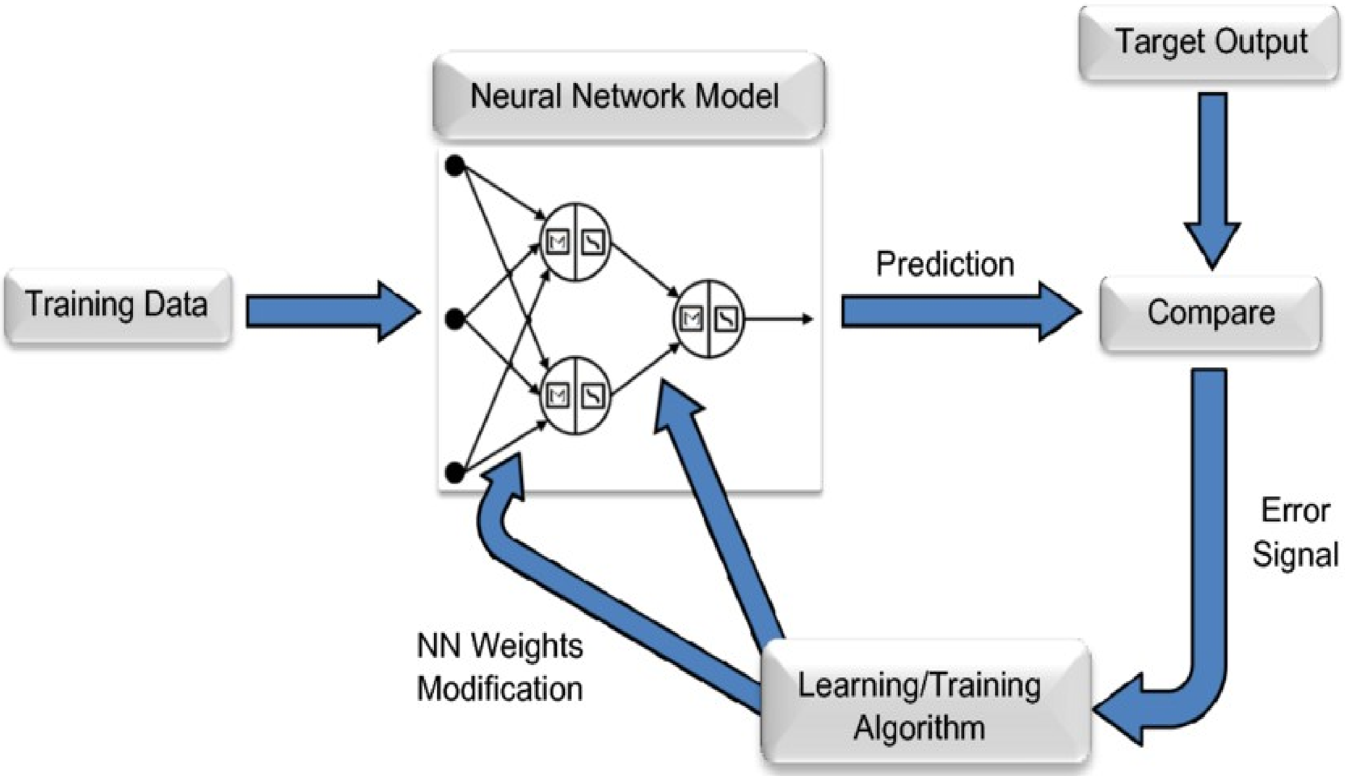

Fig 3. **ANN Training Procedure**

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

$$W = W - \alpha \frac{\partial L}{\partial W}$$

$$b = b - \alpha \frac{\partial L}{\partial b} \qquad (\alpha - \textit{Learning Rate})$$

Fig 4. **Loss function and weights updating**

Fig 3 : https://www.researchgate.net/figure/The-representation-of-neural-network-training-process_fig5_299390844
Fig 4 : https://studymachinelearning.com/setting-dynamic-learning-rate-while-training-the-neural-network/

Physics Informed Neural Network(PINN)

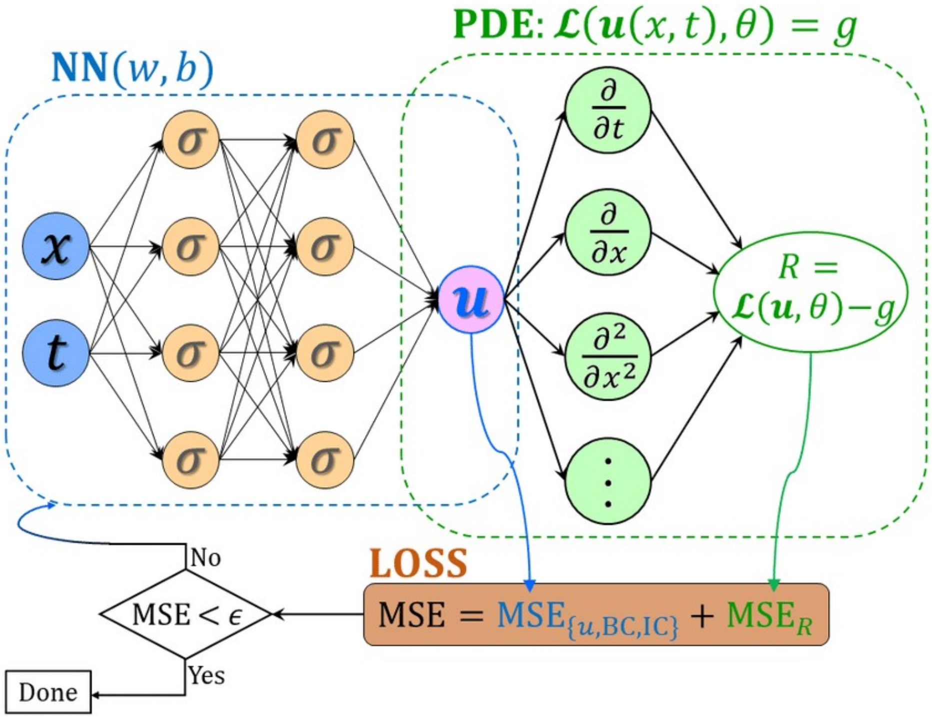

Fig 5: **PINN Architecture**

$$\vartheta \frac{\partial^2 u}{\partial x^2} = \frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x}$$

$$u(x,0) = \sin(x)\,, x \in [-1,1]$$

$$u(\pm L,t) = 0, L \in [0,1]$$

$$R = \vartheta \frac{\partial^2 u}{\partial x^2} - \frac{\partial u}{\partial t} - u \frac{\partial u}{\partial x}$$

$$MSE = \left(\vartheta \frac{\partial^2 u}{\partial x^2} - \frac{\partial u}{\partial t} - u \frac{\partial u}{\partial x}\right)^2 + \left(u(x,0) - \sin(x)\right)^2 + \left(u(\pm L,t) - 0\right)^2$$

$$MSE = MSE_R + MSE_{\{BC,IC\}}$$

Fig 5 : https://www.researchgate.net/figure/Schematic-of-a-physics-informed-neural-network-PINN-where-the-loss-function-of-PINN

Results : 1.Rod with uniform Axial Load

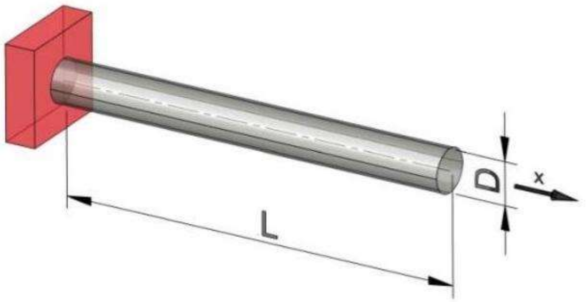

Fig.6 **Rod with UAL**

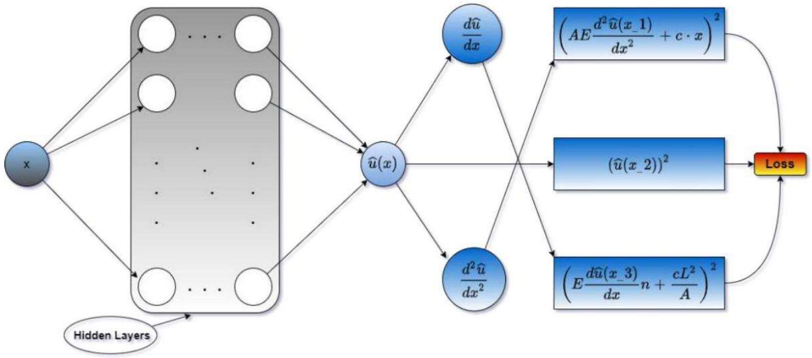

Fig.7 **PINN**

$$Governing\ Differential\ Equation:\ AE\frac{d^2u}{dx^2} = -\mathrm{cx}\ , \mathrm{x} \in (0, L)$$

$$Boundary\ Conditions:\ u(0) = 0, \frac{du}{dx}|_{\mathrm{x=L}} = 0$$

$$Loss\ Function: MSE = (AE\frac{d^2u}{dx^2} + \mathrm{cx})^2 + (u(0) - 0)^2 + (\frac{du}{dx} - 0)^2$$

Continued…

No	Layers	epochs	Loss (MSE)	Time (sec)
1	[5, 5]	600	7.1E-5	12
2	[10, 10]	500	6.67E-5	11
3	[5, 5, 5]	500	3.997E-5	12
4	[15, 15]	500	2.941E-5	13

Table 1:**Optimized Parameters for Rod with UAL**

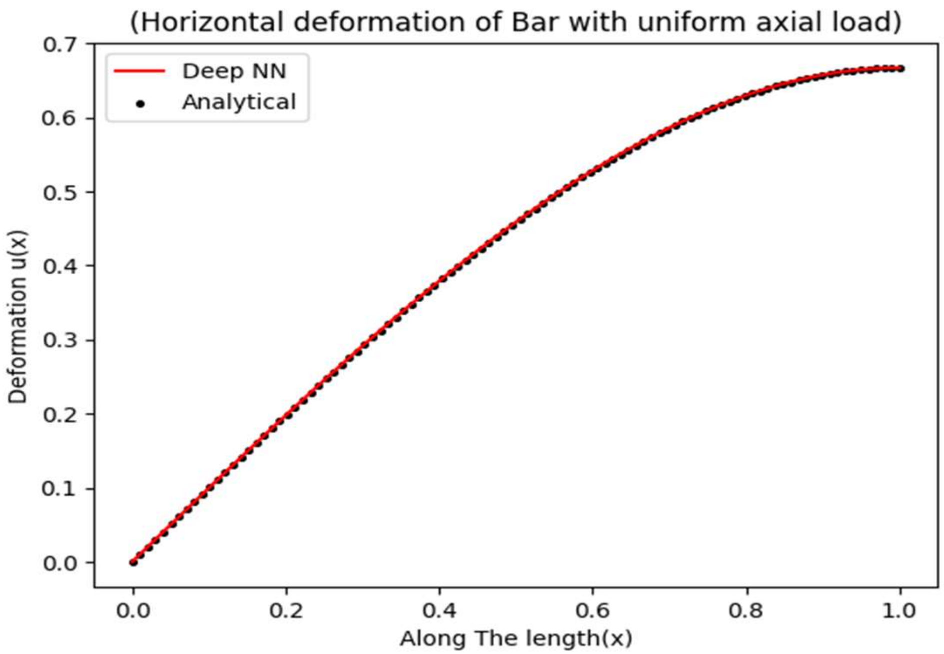

Fig 8: **PINN vs True Solution**

2.Simply Supported Beam with UDL

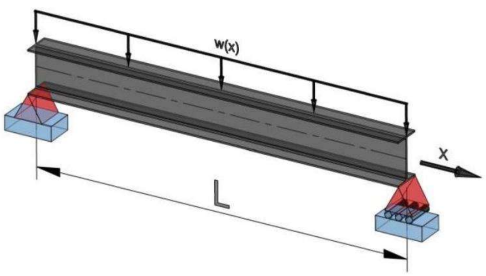

Fig.9 **SSB with UDL**

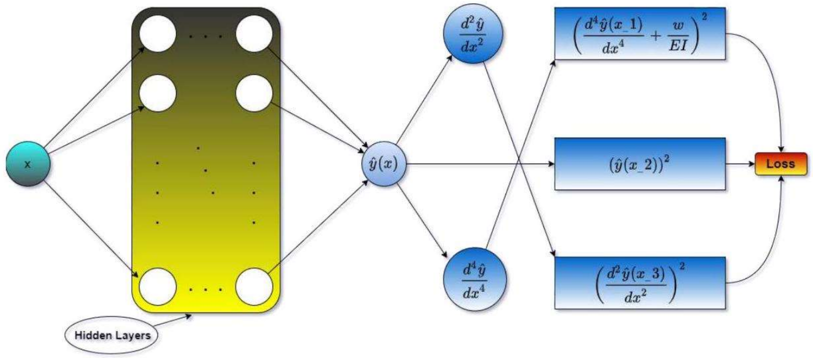

Fig 10: **PINN**

$$Governing\ differential\ eq: \frac{d^4u}{dx^4} = -\frac{w(x)}{EI}, x \in (0, L)$$

$$B.Cs: u(0) = 0, u(L) = 0, \frac{d^2u(0)}{dx^2} = 0, \frac{d^2u(L)}{dx^2} = 0$$

$$Loss\ Function = \left(\frac{d^4u}{dx^4} + \frac{w(x)}{EI}\right)^2 + MSE_{BC}$$

Continued...

No	Layers	epochs	Loss (MSE)	Time (sec)
1	[35, 35]	1000	3.13E − 06	21
2	[50, 20]	1500	4.28E − 07	28
3	[50, 20]	2000	3.4E − 07	31
4	[20, 20, 20]	1000	3.87E − 06	29
5	[30, 20, 20]	1000	2.53E −06	30
6	[40, 30, 30]	1500	2.2E − 07	33
7	40, 30, 30]	2000	8.7E −08	44

Table 2: **Optimized parameters for SSB with UDL**

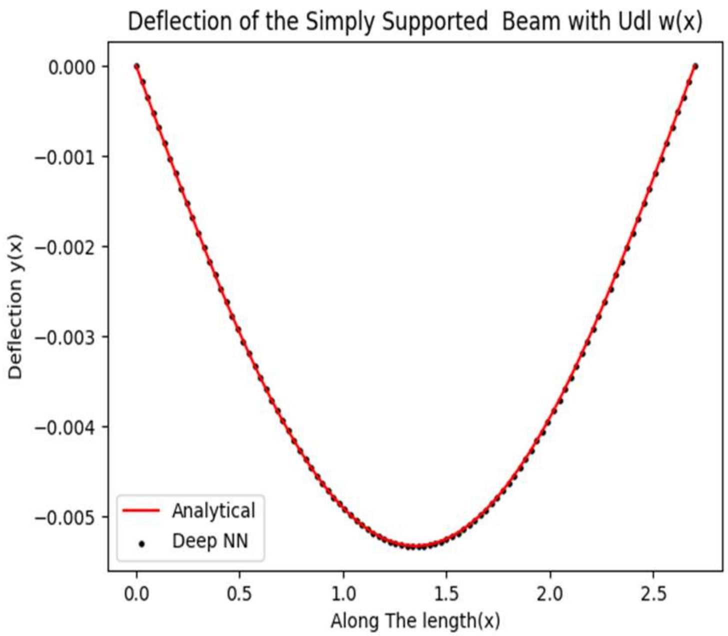

Fig 11: **PINN vs True Solution**

3.Cantilever Beam with UDL

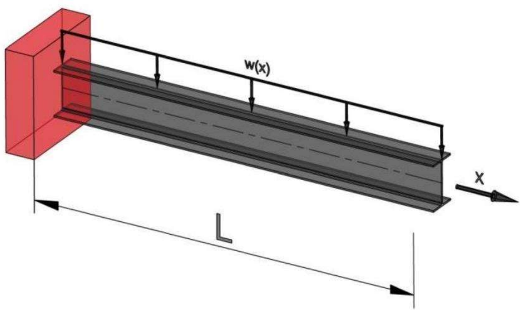

Fig.12 **Cantilever with UDL**

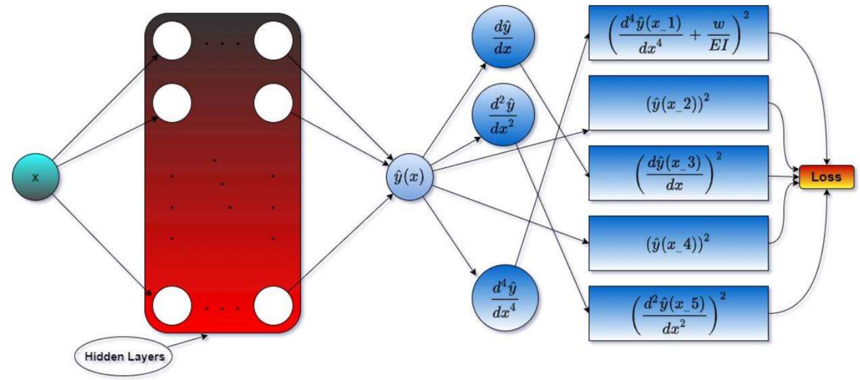

Fig 13**: PINN**

$$Governing\ differentail\ equ: \frac{d^4u}{dx^4} = -\frac{w(x)}{EI}\ , x \in (0, L)$$

$$B.Cs:\ u(0) = 0, \frac{du(0)}{dx} = 0, \frac{d^2u(L)}{dx^2} = 0, \frac{d^3u(L)}{dx^3} = 0$$

$$Loss\ Function = \left(\frac{d^4u}{dx^4} + \frac{w(x)}{EI}\right)^2 + MSE_{BC}$$

Continued...

No	*Layers*	*epochs*	*Loss (MSE)*	*Time (sec)*
1	[25,25]	1500	2.15E-07	*24*
2	[30,30]	2500	3.53E-07	*17*
3	[40,30]	2500	2.04E-08	*17*
4	[40,30,40]	2000	9.50E-08	*22*
5	[40,30,40]	2500	6.40E-08	*32*
6	[24,24,12]	2000	2.70E-08	*28*

Table 3: **Optimized parameters for Cantilever Beam with UDL**

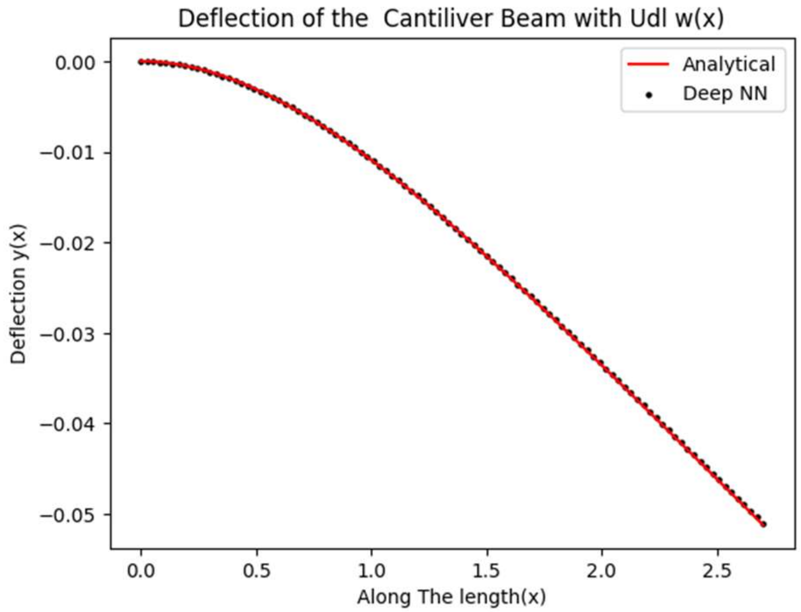

Fig 14: **PINN vs True Solution**

4.Propped Cantilever Beam with UDL

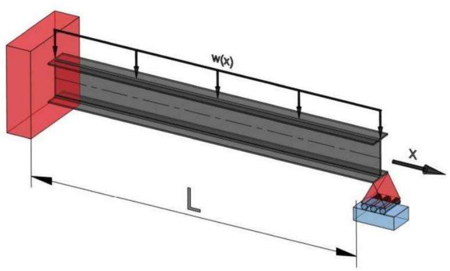

Fig.15 **Cantilever with UDL**

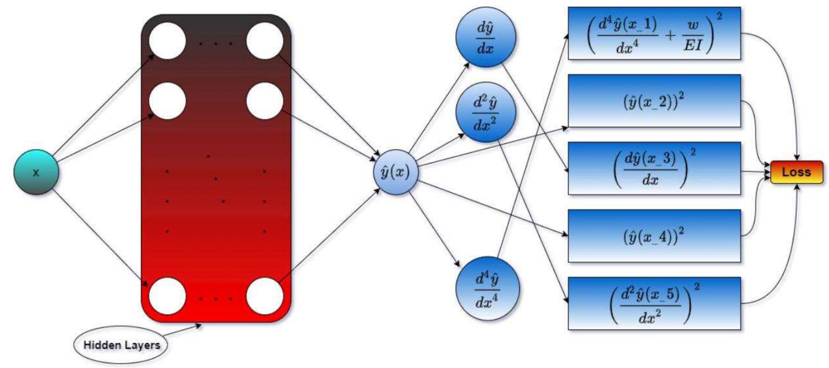

Fig 16: **PINN**

$$Governing\ dif\ equ{:}\ \frac{d^4u}{dx^4} = -\frac{w(x)}{EI}\ , x \in (0, L)$$

$$BCs{:}\ u(0) = 0, \frac{du(0)}{dx} = 0, \frac{d^2u(L)}{dx^2} = 0, \frac{d^3u(L)}{dx^3} = 0$$

$$Loss\ Function = \left(\frac{d^4u}{dx^4} + \frac{w(x)}{EI}\right)^2 + MSE_{BC}$$

Continued...

No	Layers	epochs	Loss (MSE)	Time (sec)
1	[2,20]	1000	1.64E-06	27
2	[25,25]	2000	7.34E-07	54
3	[25,25,25]	2000	1.06E-06	26
4	[30,25,25]	2000	8.39E-07	26
5	[30,30,25]	2500	6.17E-07	34
6	[40,30,20]	3000	3.40E-08	35
7	[50,40,30]	3000	3.27E-08	41

Table 4:**Optimized parameters for PCB with UDL**

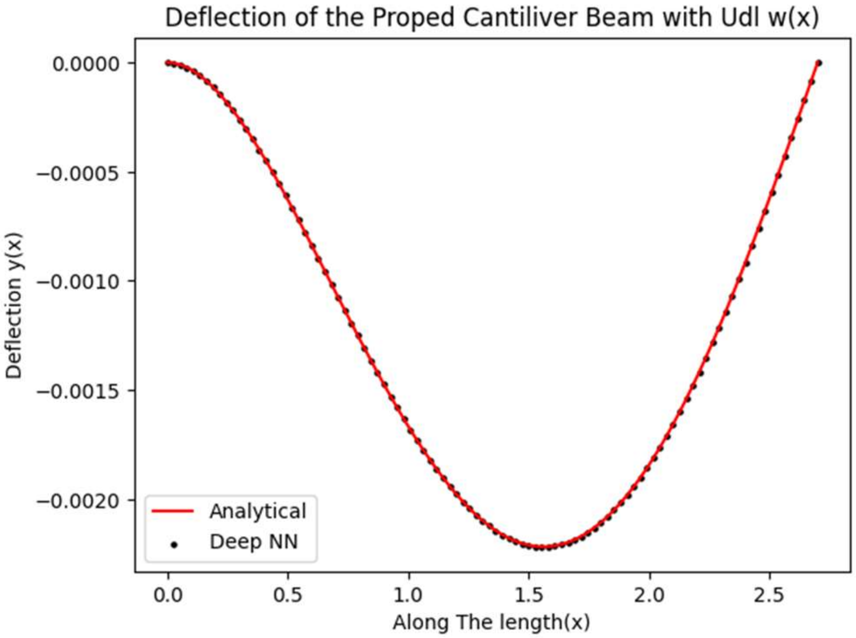

Fig 17: **PINN vs True Sol**

5.Cantilever Beam with UVL

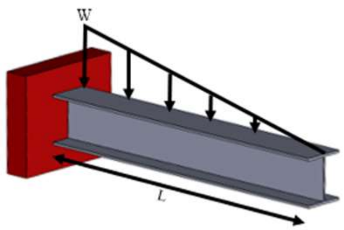

Fig.18 **Cantilever with UDL**

$$Governing\ diff.equ: \frac{d^4u}{dx^4} = -\frac{w(x)}{EI}\left(1-\frac{x}{L}\right), x \in (0, L)$$

$$BCs: u(0) = 0, \frac{du(0)}{dx} = 0, \frac{d^2u(L)}{dx^2} = 0, \frac{d^3u(L)}{dx^3} = 0$$

$$MSE = \left(\frac{d^4u}{dx^4} - f(x)\right)^2 + (u(0) - 0)^2 + \left(\frac{du(0)}{dx} - 0\right)^2 + \left(\frac{d^2u(L)}{dx^2} - 0\right)^2 + \left(\frac{d^3u(L)}{dx^3} - 0\right)^2$$

Continued...

No	Layers	epochs	Loss (MSE)	Time (sec)
1	[15,15]	1500	1.16E-09	9
2	[15,15]	2000	6.83E-10	12
3	[15,10,5]	2000	4.98E-09	18
4	[15,10,5]	3000	2.12E-09	27
5	[15,15,5]	3000	1.79E-09	28
6	[16,10,4]	3000	6.00E-08	25
7	[50,50]	3000	2.00E-10	23
8	[50,50]	2000	2.70E-11	14

Table 5: **Optimized parameters of Cantilever UVL**

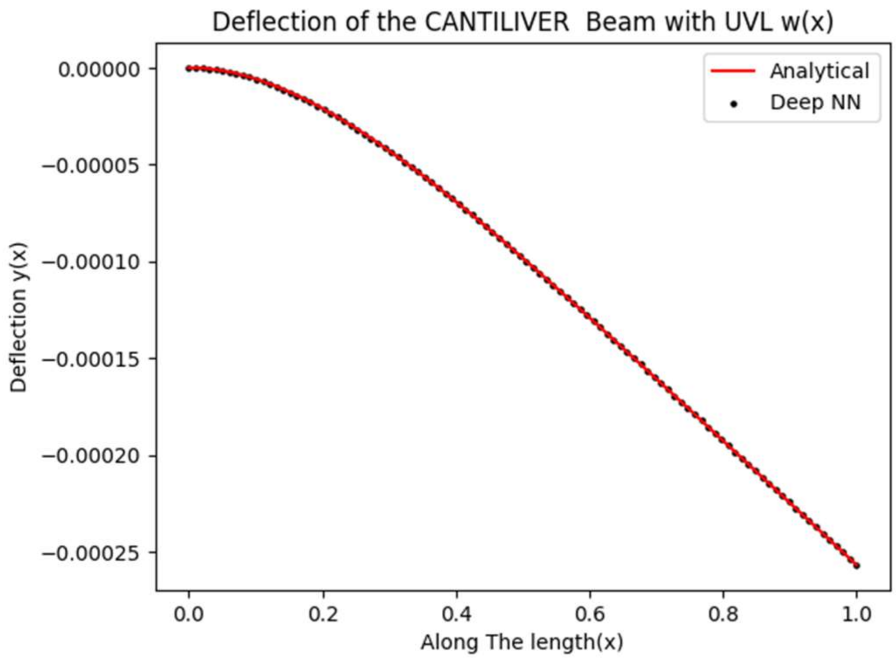

Fig 19: **PINN vs True Sol**

6. SSB with Multipoint loading

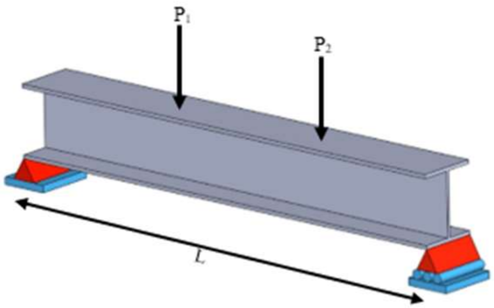

Fig.20 **Cantilever with UDL**

```
def stepFunction(x, a, b):
    M1=(60000*x)/(E*I)
    M2=(60000*x-48000*(x-1))/(E*I)
    M3=(60000*x-48000*(x-1)-40000*(x-3))/(E*I)
    # Initialize output tensor with zeros
    condition1 =  x < a
    condition2 = (x >= a) & (x < b)
    condition3 = (x >= b)
    region1 = torch.ones_like(x)*M1
    region2 = torch.ones_like(x)*M2
    region3 = torch.ones_like(x)*M3
    output = torch.where(condition1, region1,
    torch.where(condition2, region2, region3))
    return output
```

Fig 21: **Pseudo code of step Function**

$$\frac{d^2u}{dx^2} = stepFunction(x, a, b), x \in (0, L)$$

$$u(0) = 0, u(L) = 0$$

$$MSE = (\frac{d^2u}{dx^2} - stepFunction(x, a, b))^2 + (u(0) - 0)^2 + (u(L) - 0)^2$$

Continued...

No	Layers	epochs	Loss (MSE)	Time (sec)
1	[150,100,50,25]	3000	1.00E-08	67
2	[150,100,50,25]	3000	1.80E-08	60
3	[60,40,50,25]	3000	1.48E-08	42

Table 6: **Optimized parameters of SSB with MultiPoint Loading**

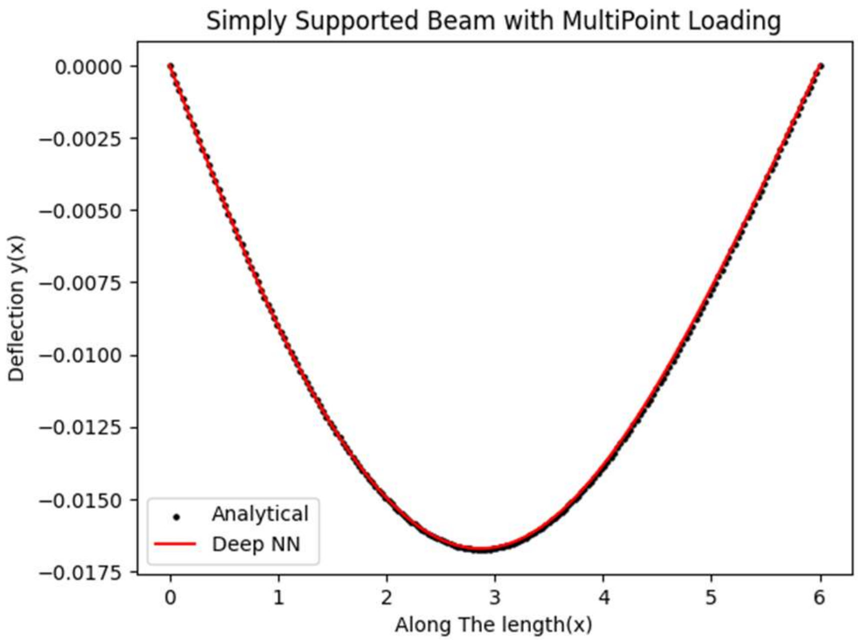

Fig 22: **PINN vs True Sol**

7. Cantilever Beam with Multi UDL loading

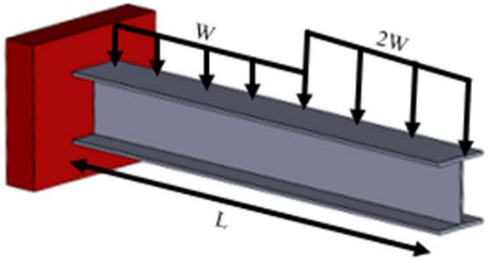

Fig.23 **Cantilever with UDL**

$$\frac{d^4u}{dx^4} = -stepFunction(x, a, b), x \in (0, L)$$

$$u(0) = 0, \frac{du(0)}{dx} = 0, \frac{d^2u(L)}{dx^2} = 0, \frac{d^3u(L)}{dx^3} = 0$$

$$MSE = \left(\frac{d^4u}{dx^4} - stepFunction(x, a, b)\right)^2 + (u(0) - 0)^2 + \left(\frac{du(0)}{dx} - 0\right)^2 + \left(\frac{d^2u(L)}{dx^2} - 0\right)^2 + \left(\frac{d^3u(L)}{dx^3} - 0\right)^2$$

Continued...

No	Layers	epochs	Loss (MSE)	Time (sec)
1	[80]	4000	2.13E-07	26
2	[60,50]	4000	2.00E-07	36
3	[60,50,50]	3000	1.80E-07	45
4	[50,30,20]	3000	4.30E-07	50
5	[40,30,20,10]	4000	3.00E-07	67
6	[100,75,50,25]	4000	2.80E-07	90

Table 7: **Optimized parameters for Cantilever Beam with Multi UDL**

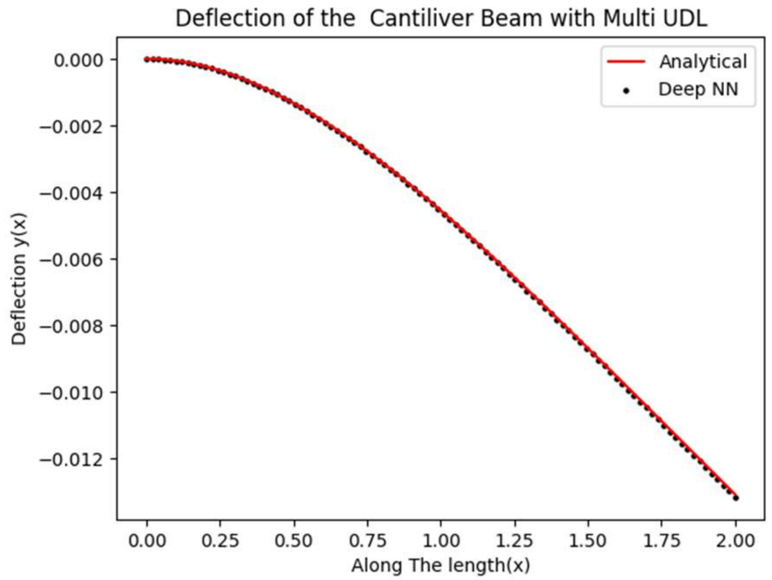

Fig 24: **PINN vs True Sol**

Future Plan

❖ Although this current work has made significant strides in using PINN to solve structural problems, there are still a number of areas that can be explored and improved. Here, we highlight potential research directions for the future:

- Enhancing accuracy and convergence
- Handling complex geometries
- Uncertainty quantification
- Integration with experimental data
- Parallelization and scalability

References

- Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2017). Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations.
- Hanke, M., & Obersteiner, M. (1994). Solving Ordinary Differential Equations with Neural Networks.
- Long, Z., Lu, Y., \& Dong, B. (2020). DeepXDE: A Deep Learning Library for Solving Differential Equations.
- Zhang, L., Wang, D., \& Zhao, L. (2018). A Physics-Informed Neural Network for Multiphase Flow in Porous Media.

Continued...

- Katsikis, D., Muradova, A., & Stavroulakis, G. (2022). A Gentle Introduction to Physics-Informed Neural Networks, with Applications in Static Rod and Beam Problems. Journal of Advances in Applied & Computational Mathematics,9,103-128. doi:10.15377/2409-5761.2022.09.8

- Wong, T.-K. L., Xu, J., & Yang, X. (2019). Solving High-Dimensional Partial Differential Equations Using Deep Learning.

- Zhang, D., Yu, B., & Shu, C.-W. (2020). A Machine Learning Framework for Solving High-Dimensional Partial Differential Equations.

- Karniadakis, G. E. (2020). Physics-Informed Deep Learning for Fluid Dynamics: A Review.

Thank you.