NEURAL OPERATORS AND OPERATOR NETWORKS VS PARAMETRIC APPROACH: A GENERAL COMPARISON*

Mingyuan Chi*, Dingran Feng*, Wenkai Xuan*
Department of Mathematics
ETHz
{minchi,difeng,wexuan}@student.ethz.ch

Yuke Cai*
Department of Mechanical and
Process Engineering
ETHz
yukcai@student.ethz.ch

ABSTRACT

We explore the use of different neural operator architectures for solving partial differential equations (PDEs). Specifically, it investigates the performance of the Feed-Forward Network (FFN), Deep Operator Network (DeepONet), Fourier Neural Operator (FNO), Convolutional Neural Operator (CNO), and Koopman Neural Operator (KNO) in solving the heat, wave, and Poisson equations. The models are trained and evaluated based on their ability to accurately predict the solutions of these PDEs under varying parameters.

1 INTRODUCTION

Partial differential equations (PDEs) play a crucial role in describing various physical phenomena and are widely used in many scientific and engineering domains. Solving PDEs accurately is essential for understanding and predicting the behavior of complex systems. Traditional numerical methods for solving PDEs often rely on discretization techniques and iterative algorithms, which can be computationally expensive and time-consuming. In recent years, there has been a growing interest in leveraging neural networks and deep learning techniques to solve PDEs more efficiently Kovachki et al. (2021).

Neural operators have emerged as a promising approach for solving PDEs using neural networks. These architectures aim to learn the underlying equations directly from data, enabling faster and more accurate predictions of PDE solutions. In this paper, we compare and evaluate several neural operator architectures, namely the Feed-Forward Network (FFN), Deep Operator Network (DeepONet) Lu et al. (2019), Fourier Neural Operator (FNO) Li et al. (2020), Convolutional Neural Operator (CNO) Raonic et al. (2023), and Koopman Neural Operator (KNO) Xiong et al. (2023), in the context of solving the heat, wave, and Poisson equation.

The heat equation describes the diffusion of heat in a medium, while the wave equation models the propagation of waves. On the other hand, the Poisson equation represents a static equation that does not depend on time. These equations have different characteristics and require different approaches for accurate solution approximation. By examining the performance of various neural operator architectures on these equations under varying parameters, we can gain insights into their strengths and weaknesses. Our code can be publicly accessed by https://github.com/walkerchi/DeepLearning-in-ScientificComputing-PartB

2 EQUATIONS

Heat Equation: The heat equation is a fundamental partial differential equation (PDE) that describes the behavior of heat distribution over time. It can be expressed as follows:

$$\frac{\partial u(t,\vec{x})}{\partial t} = \Delta u \tag{1}$$

*These authors contributed equally to this work

Here, $\vec{x}$ represents the position vector, which lies within the domain $\mathcal{D}^2_{\text{heat},\vec{x}} = [-1, 1]^2$, indicating that both elements x_1 and x_2 reside in the interval $[-1, 1]^2$. The variable t spans the domain $\mathcal{D}_{\text{heat},t} = [0, T]$, where T is the final time. In Appendix A.2.1, we provide a solution to the heat equation under a specific condition.

Wave Equation: The wave equation is another important PDE that describes the propagation of waves, such as sound waves or electromagnetic waves. It can be represented as follows:

$$\frac{\partial^2 u(t, \vec{x})}{\partial t^2} - c^2 \Delta_{\vec{x}} u = u_{tt} - c^2(u_{xx} + u_{yy}) \tag{2}$$

In this equation, $\vec{x}$ is the position vector, confined within the domain $\mathcal{D}^2_{\text{wave},\vec{x}} = [0, 1]^2$, indicating that the elements x_1 and x_2 lie within the interval $[0, 1]^2$. The variable t is restricted to the domain $\mathcal{D}_{\text{wave},t} = [0, T]$. The symbol c represents the propagation speed. In Appendix A.2.2, we solve the wave equation under a specific condition.

Poisson Equation: The Poisson equation is a static PDE that describes the distribution of scalar fields in various physical phenomena, such as electrostatics or fluid flow. It can be defined as follows:

$$-\Delta_{\vec{x}} u(\vec{x}) = f \tag{3}$$

Unlike the heat equation and wave equation, the Poisson equation does not depend on time. In Equation 3, f represents the source function. In Appendix A.2.3, we provide a solution to the Poisson equation under a specific condition.

3 METHODOLOGY

In the following illustration, we sample $\mathcal{S}$ times for different $\vec{\mu}/\mathbf{a}$. Therefore, for each ith sample, $\vec{\mu}^i \in \mathcal{D}^d_{\text{heat},\vec{\mu}}$ and $\mathbf{a}^j \in \mathcal{D}^{K\times K}_{\text{wave},\mathbf{a}}$

Parameter Approach:Feed Forward Network(FFN): The objective of the parameter methodology is to establish a transformative function g, which is capable of constructing a bridge between the position vector $\vec{x}$ and parameter $\vec{\mu}/\mathbf{a}$ for the equation to the resultant value u of equation 4.

$$g(\vec{x}, \{\vec{\mu}, \mathbf{a}\}) = u(T, \vec{x}, \{\vec{\mu}, \mathbf{a}\}) \tag{4}$$

With the aid of the universal approximation theorem of Multilayer Perceptron (MLP) Hornik et al. (1989), we have chosen to implement a powerful MLP to closely approximate the function g.

DeepONet: In contrast to the parameter approach, DeepONet Lu et al. (2019) adopts a different strategy. It learns a mapping function G, which gives rise to a continuous function that maps the position vector $\vec{x}$ to the value u of the equation, as illustrated in equation 5:

$$G(u_0)(\vec{x}) = u(T, \vec{x}) \tag{5}$$

The u_0 is a short notation of $u(0, \vec{x})$. To execute this, DeepONet employs a matrix to approximate the continuous function. It mainly consists of two networks: the branch network $\mathcal{B}$ and the trunk network $\mathcal{T}$. In Equation 6, the jth sampled position $\vec{y}^j$, pertaining to different μ^j or $\mathbf{a}^j$, is preserved for further inference within the DeepONet framework. $\mathcal{B}$ and $\mathcal{T}$ take the form of a Multilayer Perceptron (MLP).

$$\tilde{u}(T, \vec{x}, \{\vec{\mu}^j, \mathbf{a}^j\}) = \mathcal{B}\left(u(0, \vec{y}^j, \{\vec{\mu}^j, \mathbf{a}^j\})\right) \cdot \mathcal{T}(T, \vec{x})^\top \tag{6}$$

Fourier Neural Operator: In contrast to both the parameter approach and the DeepONet approach, the Fourier Neural Operator (FNO) Li et al. (2020) demands the input points to conform to a mesh structure. We refer to this type of neural operator as the Mesh Neural Operator. It takes the form as shown in equation 7:

$$G(u_0, \vec{x}) = u(T, \vec{x}) \tag{7}$$

When it comes to FNO, it adopts the form as illustrated in equation 8:

$$H^{l+1} = \sigma\left(Conv^{l}_{1\times1}(H^{l}) + \mathcal{F}^{-1}\left(W^{l}(\mathcal{F}H^{l}) + b^{l}\right)\right) \tag{8}$$

In this equation, H^l represents the hidden layer at the lth level. $Conv^l_{1\times1}$ refers to the convolution operation with a kernel size of 1×1 at lth layer. The symbols $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the Fourier transform and its inverse, respectively. W^l and b^l stand for the weights and bias of the FNO network at lth layer, respectively. Finally, σ signifies the activation function.

Convolutional Neural Operator: Inspired by the architectural principles of UNet Ronneberger et al. (2015) and the activation mechanisms of StyleGAN3 Karras et al. (2021), the Convolutional Neural Operator (CNO) Raonic et al. (2023) integrates the structural design of UNet with the Leaky Rectified Linear Unit (LReLU) filter from StyleGAN3, utilized as an activation layer.

The activation layer, as depicted in Figure 1, operates through a sequence of transformations on the feature map. Initially, the feature map undergoes an upsampling process, followed by the application of a Finite Impulse Response (FIR) filter. Subsequently, the LeakyReLU activation function Maas et al. (2013) is applied, introducing non-linearity to the feature map. A second FIR filter is then applied, followed by a downsampling operation.

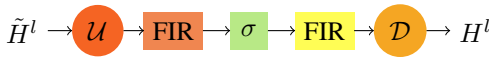


Figure 1: single activation layer of Convolutional Neural Operator

Koopman Neural Operator: In order to encapsulate complex long-term dynamics, the Koopman Neural Operator (KNO) Xiong et al. (2023) has been proposed. This operator is designed to learn the Koopman operator, an infinite-dimensional linear operator that governs all observations of a dynamic system. The KNO is applied to the evolution mapping of the dynamic system's solution, thereby capturing the system's behavior over time.

Contrasting with the Fourier Neural Operator, the KNO employs parameter sharing across layers for the Spectral Convolution, as shown in Equation 9.

$$H^{l+1} = F^{-1}\left(W(\mathcal{F}H^{l}) + b\right) \tag{9}$$

In this equation, H^{l+1} represents the output of the spectral convolution layer, $\mathcal{F}^{-1}$ is the inverse Fourier transformation, W and b are the weight and bias parameters respectively, and $\mathcal{F}H^l$ is the Fourier transformation of the input to the layer.

4 Experiment

4.1 Setup

CPU/GPU: We use the CPU/GPU of Intel i5-11300H/Nvidia MX450 for these experiments.

Equations: The parameter d/K for all equations was systematically varied across the set $[1, 2, 4, 8, 16]$. Therefore, we will train $3 \times 5 \times 6 = 90$ weights for the experiments. In the heat equation, where the attenuation of high-frequency signals over time is observed, the parameter T was assigned a value of 0.005. Conversely, in the wave equation, the propagation speed c was determined as 0.1, the radial decay parameter r was set to 0.85, and T was specified as 5. Notably, in the case of the Poisson equation, which exhibits time-independence, the radial decay parameter r was consistently assigned a value of 0.85

Sampling: The Mesh sampler is employed for all models. The samplers generated 4096 mesh spatial points (64×64) for the training dataset, validation dataset, with 64 samplings.

Model: For the FFN, a Multi-Layer Perceptron (MLP) with 4 layers was used, each layer having a hidden size of 64 and employing a ReLU activation function. For the DeepONet, both the branch and trunk networks were configured with the same settings as the FFN. The Fourier Neural Operator (FNO) and Koopman Neural Operator (KNO) also followed the same settings as the FFN. To maintain the complexity among models, for the Convolutional Neural Operator (CNO) and UNet, the depth was set to 3 with hidden channel range within $[8, 32, 128]$ and the residual block length was set to 2.

Training: The Adam optimizer was used for training, with a learning rate of 0.001 and 1000 epochs. Model validation was performed every 100 epochs using the validation dataset, and the best validation model was stored.

Acceleration: To expedite the activation process on the GPU, we utilized the implementation from StyleGAN3 Karras et al. (2021), which is implemented in CUDA.

4.2 PREDICTION

In this experiment, we set the parameter d to a fixed value of 4 for the heat equation. A single sample of u_0 and u_T was generated. We compare the predictions of each model with the ground truth u_T, as depicted in Figure 2. The error is quantified using the L_2 error metric. Notably, the Fourier Neural Operator and Koopman Neural Operator models exhibit sensitivity to the presence of high-frequency signals. The FFN and DeepONet models struggle to effectively capture the high-frequency components of the signal. Conversely, the Convolutional Neural Operator and Unet models demonstrate good fitting capability, accurately representing the signal. Additional comparative results can be found in Appendix A.3.

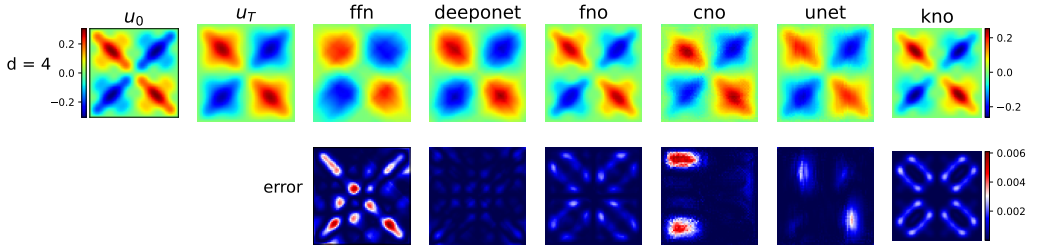
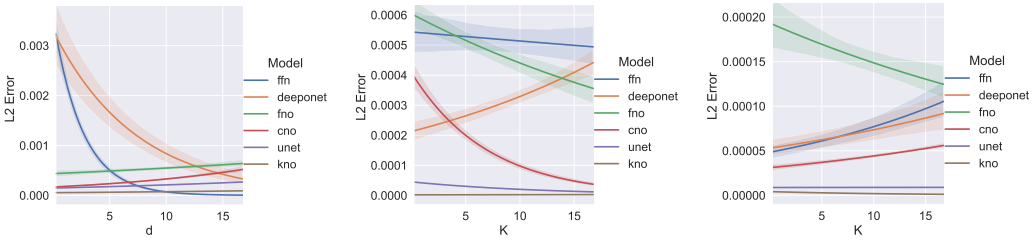


Figure 2: prediction of heat equation from different models when $d = 4$

4.3 VARYING PARAMETER

In this experiment, we generated 64 samples for each equation parameter d/K within the range $[1, 2, 4, 8, 16]$. It is important to note that separate models were trained for each parameter of the equation. The results of the experiment are illustrated in Figure 3. The shaded bands in the figure represent the 2σ uncertainty, while the curve is fitted using logistic regression. Analyzing the figure, it becomes evident that the Koopman neural operator consistently achieved the lowest error across all variations of the parameter d/K. However, it is important to note that there is no discernible ascending or descending pattern observed for certain models as the parameter d/K varies. This behavior indicates that the models respond differently to different equations, and the relationship between the parameter and the model's performance is not strictly linear.



(a) heat equation (b) wave equation (c) poisson equation

Figure 3: L2 error for different equation parameter (d/K)

5 CONCLUSION

In conclusion, neural operator architectures show great promise in solving PDEs efficiently and accurately. This study contributes to the understanding of different neural operator architectures and their performance in solving the heat, wave, and Poisson equations. Further research can explore the application of neural operators to more complex and diverse PDEs, as well as the development of hybrid approaches that combine the strengths of different architectures.

REFERENCES

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *Proc. NeurIPS*, 2021.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces. *arXiv preprint arXiv:2108.08481*, 2021.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, pp. 3. Atlanta, Georgia, USA, 2013.

Bogdan Raonic, Roberto Molinaro, Tobias Rohner, Siddhartha Mishra, and Emmanuel de Bezenac. Convolutional neural operators. In *ICLR 2023 Workshop on Physics for Machine Learning*, 2023.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241. Springer, 2015.

Wei Xiong, Xiaomeng Huang, Ziyang Zhang, Ruixuan Deng, Pei Sun, and Yang Tian. Koopman neural operator as a mesh-free solver of non-linear partial differential equations. *arXiv preprint arXiv:2301.10022*, 2023.

A APPENDIX

A.1 CONTRIBUTION

Mingyuan Chi: work on main structure of the code, all the models, training and plotting infrastructure, especially for heat equation. And work on main structure of the report.

Yuke Cai: work on the details of the code, the training of the models and the setup of all models for wave equation.

Wenkai Xuan: work on the details of the codes, models of the poisson equation part and the details of the report.

Dingran Feng: work on the coding of configuration for parameters such as sampling density and models, debugging of programs, writing of repository README and report.

A.2 EQUATIONS

A.2.1 HEAT EQUATION

The initial condition is constructed as equation 10:

$$u(0, \vec{x}, \vec{\mu}) = -\frac{1}{d} \sum_{m=1}^{d} \mu_m sin(\pi m x_1) sin(\pi m x_2) / \sqrt{m} \tag{10}$$

The parameter vector $\vec{\mu}$ occupies the domain $\mathcal{D}^d_{\text{heat},\vec{\mu}} = [0, 1]^d$. We also apply a zero Dirichlet boundary condition 11:

$$u(t, \{0,1\}, \{0,1\}) = 0 \tag{11}$$

In light of the governing equation 1, initial condition 10, and boundary condition, we can deduce the exact solution as demonstrated in equation 12:

$$u(t, \vec{x}, \vec{\mu}) = -\frac{1}{d} \sum_{m=1}^{d} \frac{\mu_m}{\sqrt{m}} e^{-2m^2\pi^2 t} sin(\pi m x_1) sin(\pi m x_2) \tag{12}$$

In this expression, μ_m denotes the mth element of the vector $\vec{\mu}$, while x_1 and x_2 represent the first and second element of the position vector $\vec{x}$, respectively.

A.2.2 WAVE EQUATION

The initial condition is formulated as per equation 13:

$$u(0, x, y, \mathbf{a}) = \frac{\pi}{K^2} \sum_{i,j=1}^{K} a_{ij} \cdot (i^2 + j^2)^{-r} sin(\pi i x) sin(\pi j y) \tag{13}$$

The parameter matrix $\mathbf{a}$ is defined within the domain $\mathcal{D}_{\text{wave},\mathbf{a}}^{K\times K} = [0,1]^{K\times K}$. r denotes the radial decay factor for the source.

We employ a zero Dirichlet boundary condition, as expressed in equation 14:

$$u(t, \{0,1\}, \{0,1\}, \mathbf{a}) = 0 \tag{14}$$

In consideration of the governing equation 2, the initial condition 13, and the boundary condition 14, the solution can be derived as showcased in equation 15:

$$u(t, \vec{x}, \mathbf{a}) = \frac{\pi}{K^2} \sum_{i,j=1}^{K} a_{ij} \cdot (i^2 + j^2)^{-r} sin(\pi i x_1) sin(\pi j x_2) cos(c\pi t \sqrt{i^2 + j^2}) \tag{15}$$

In this equation, a_{ij} corresponds to the element situated at the ith row and jth column of the matrix $\mathbf{a}$, while x_1 and x_2 represent the first and second element of the position vector $\vec{x}$, respectively.

A.2.3 POISSON EQUATION

For simplicity, we define the source function f as shown in Equation 16:

$$f(\vec{x}, \mathbf{a}) = \frac{\pi}{K^2} \sum_{i,j=1}^{K} a_{ij} \cdot (i^2 + j^2)^{r} \sin(\pi i x_1) \sin(\pi j x_2) \tag{16}$$

Here, $\vec{x}$ lies within the same domain as the wave equation, $\vec{x} \in \mathcal{D}_{\text{poisson},\vec{x}}^{2} = [0,1]^2$, and r is defined as the radial decaying factor. The domain of a is shared with the wave equation, $a \in \mathcal{D}_{\text{poisson},\mathbf{a}}^{K\times K} = [-1,1]^{K\times K}$. We also applied zero Dirichlet boundary conditions.

Using the source function defined in Equation 16 and the boundary condition, we can solve the Poisson partial differential equation (Equation 3) to obtain the solution described by Equation 17:

$$u(\vec{x}) = \frac{1}{\pi K^2} \sum_{i,j=1}^{K} a_{ij} \cdot (i^2 + j^2)^{r-1} \sin(\pi i x_1) \sin(\pi j x_2) \tag{17}$$

A.3 PREDICTION

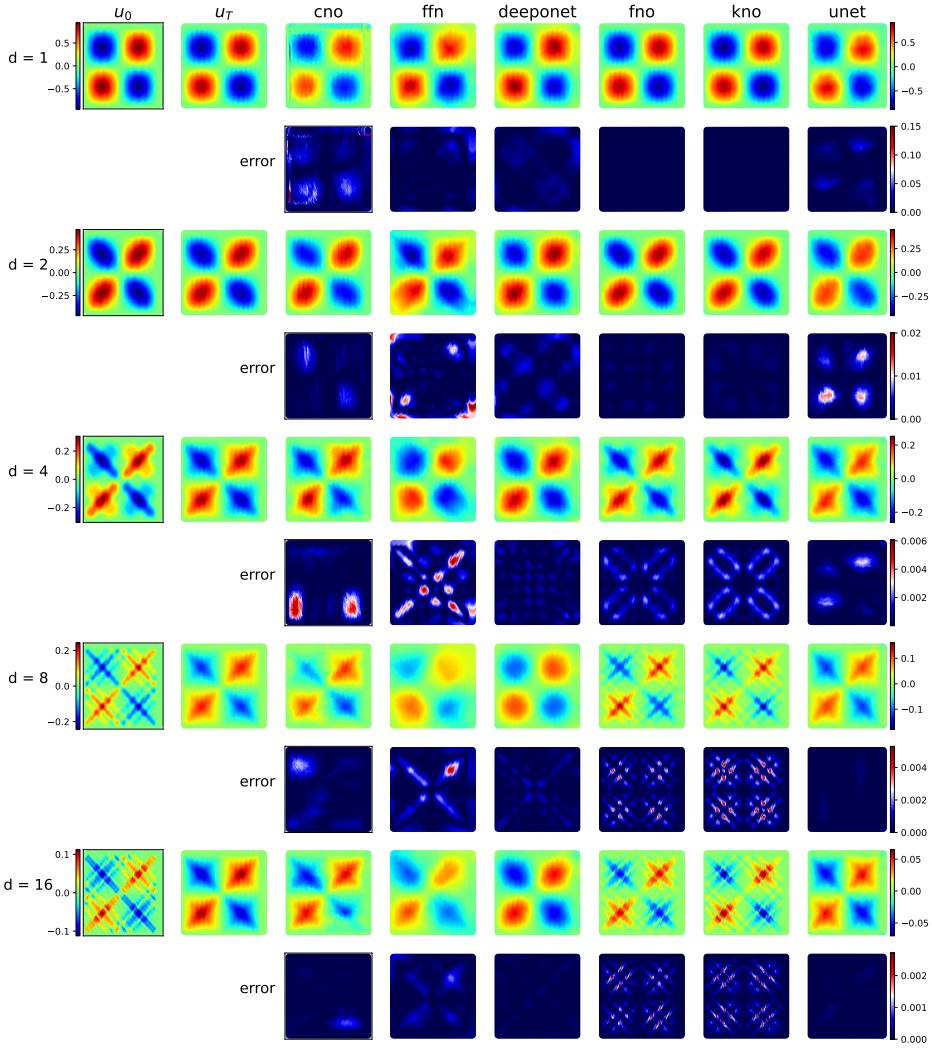


Figure 4: prediction of heat equation from different models with different equation parameters

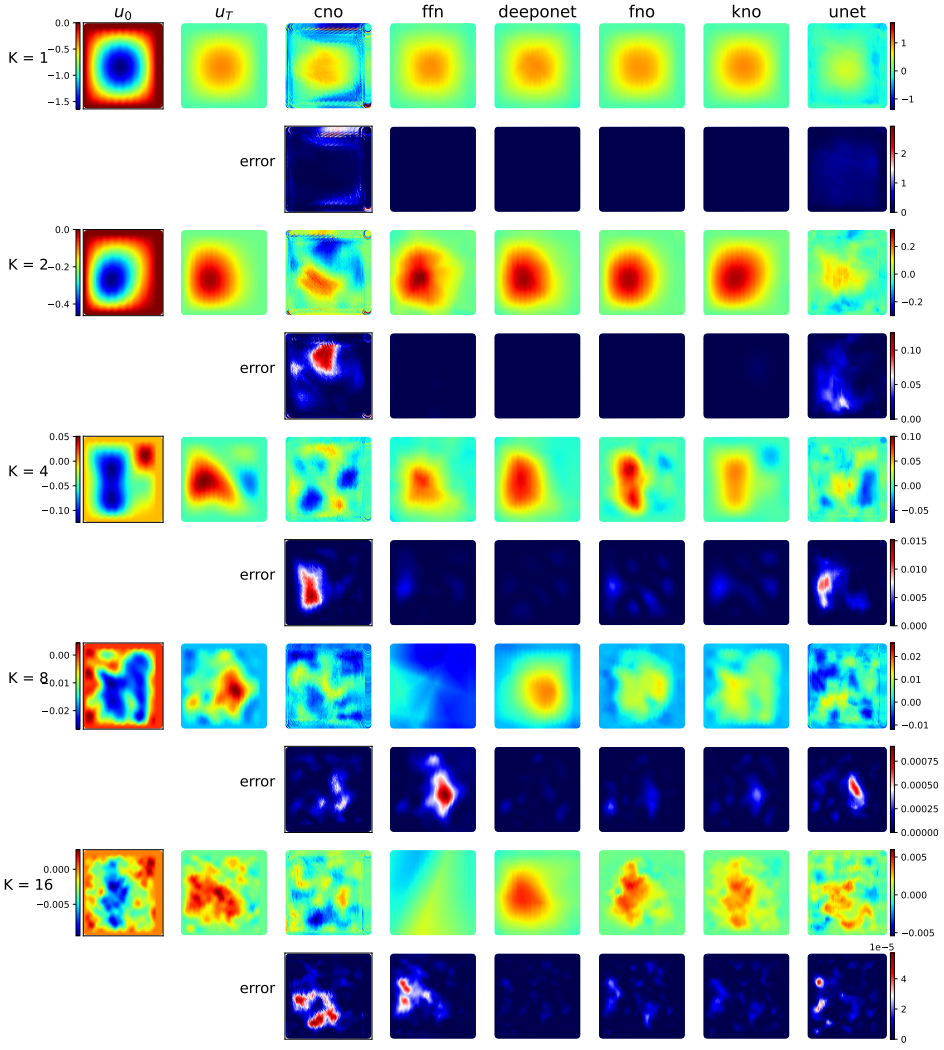


Figure 5: prediction of wave equation from different models with different equation parameters

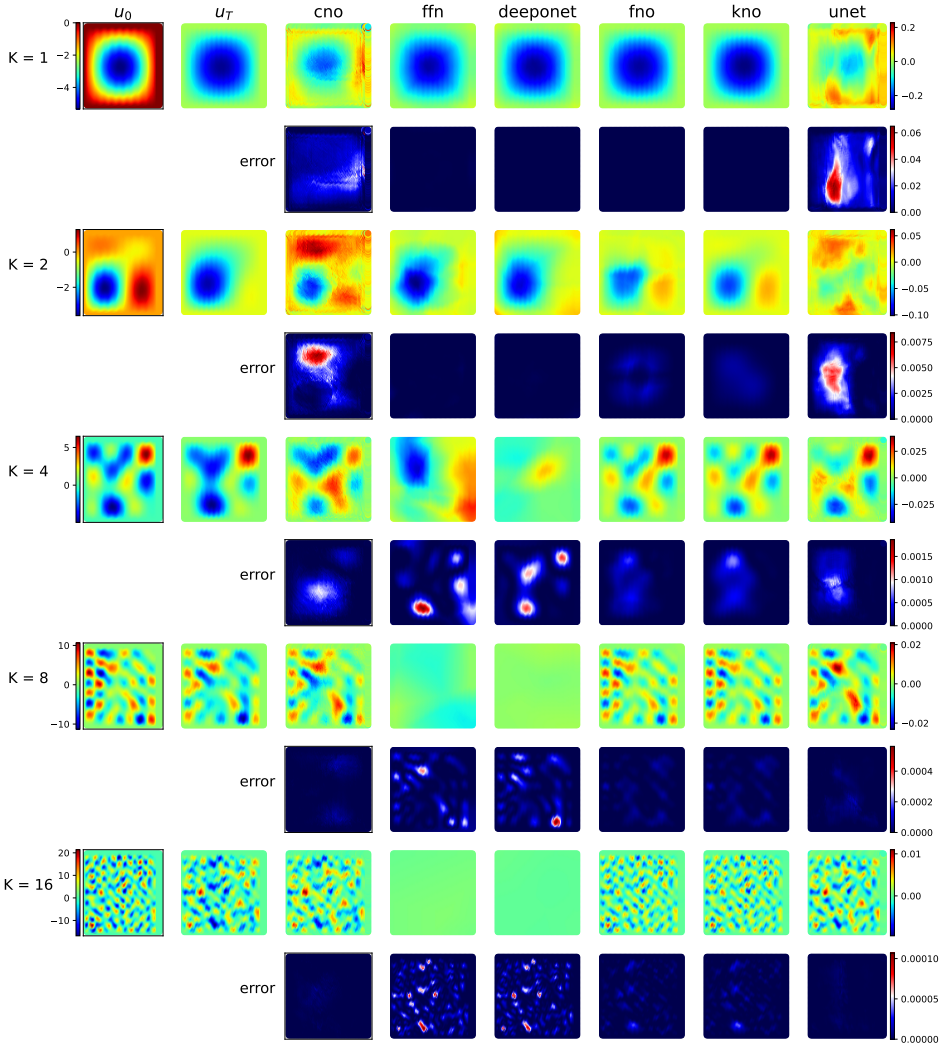


Figure 6: prediction of poisson equation from different models with different equation parameters