Math Independent Study Final Report

Shuhan Yuan

Abstract

In recent years, data-driven methods have made several advances in approximating unknown partial differential equation systems (PDE). Fourier neural network (FNO) (Li et al., 2020 [1]) is an data-driven approach directly mapping from any functional parametric dependence to the solution with mesh free property. In this independent study, we conduct further experiments to evaluate its performance in 1-D heat equation and wave equation with non-zero Dirichlet and Neumann boundary conditions and justified its fast learning ability and super-resolution property.

1 Introduction

A large extent of partial differential equation (PDE) systems have contributed to our understanding and simulation of the fundamental processes of the natural world. Examples include the vibrations of solids, the flow of fluids, the diffusion of chemicals, the spread of heat, the structure of molecules, the interactions of photons and electrons, and the radiation of electromagnetic waves. Apart from that, PDE also plays an important role in modern mathematics, especially in geometry and analysis. Discretization is often being used when we want to capture the latent representation of a continuous and complicated PDE system. Considering the slow and inefficient limitation of the traditional analytical computation of solutions, the emphasis of PDE solvers has been gradually shifting to faster machine learning methods with the increasing availability of powerful computers.

In recent years, researches have been made to approximate those unknown PDE systems based on available data (Yeung et al., 2017 [2]; Wu et al., 2020 [3]; Grohs et al., 2020 [4]; Kovachki et al., 2021 [5]). These approaches are based on deep neural network frameworks, like Deep Residual Network (ResNet) (He et al.,2015 [6]) or Physics-informed Neural Networks (PINNs) (Raissi et al., 2019[7]), and are experimented on well-studied PDE system to evaluate their performance. Their performance achieved great advance on making spacial or temporal approximation of the PDE system despite of noise, corruptions as well as limitation on data.

Fourier neural operator method (FNO) (Li et al., 2020 [1]) was proposed to directly learn from the mapping between function spaces and instead of solving only one instance of the PDE, it can be generalized to an entire family of PDEs. Besides, FNO is resolution-invariant approach, i.e. it can achieve similar performance regardless of the scale of data points used for training. And as

a consequence, it can be trained on low resolution to save training time and tested on high resolution. In FNO's paper, experiments are made on Burgers' Equation, Darcy Flow and Navier Stokes to show its great performance. In this independent study, further experiments and researches are conducted on two most basic PDE systems: heat equation and wave equation to show how FNO approximated the exact solution. In addition, auxiliary conditions are more complicated than in FNO's paper, non-zero boundary condition will be added to the experiment settings. Noises are added to justify FNO's ability in front of corrupted data.

2 Related Work

2.1 Traditional Computation of Solution

Some PDE systems' solutions can be well-defined by formulas, however, other problems may not be as simple and have exact formula solutions. Even for those who have a formula solution, solving it with specified conditions may be rather complicated. The traditional computation of PDE solutions often relates to reduce the process of solving a PDE with its auxiliary conditions to a finite number of arithmetical calculations that can be carried out by computer, such as finite element methods (FEM) and finite difference methods (FDM) (Strauss, 2007 [8]). However, these traditional computation methods have their disadvantages. First, for different PDE system, the method needs to be carefully chosen, otherwise, the numerically computed results can be very different from the grand true value. Another disadvantage is that such computations can be very time-consuming. For an extremely complicated PDE system, it may take years to compute the result.

2.2 Data-driven Method

Data-driven methods are approaches allowing model to directly learn the trajectory of an entire family of PDEs from the data. In terms of PDE solver, the data-driven methods are targeted at approximating the solution at any point (spatially or temporally) based on limited given data of the unknown PDE. Data-driven PDE methods can be categorized as supervised learning if the emulation of mappings between function spaces is allowed. However, even if the governing equations, like conservation of energy, is local, PDE's solution operator are non-local. Such non-locality leads to neural network frameworks, which can greatly incorporate non-locality, to be often used in data-driven methods (Lu et al., 2019 [9]; Bhattacharya et al., 2020 [10]; Wu et al., 2020 [3]; Khoo et al., 2021 [11]), bringing another advantage which allows these approaches to be orders of magnitude faster than the traditional solvers. The evolution operator (Wu et al., 2020 [3]), for instance, is such a data-driven method which can temporally map data of unknown PDE systems: the solution at current time can be approximately evolved by model and estimate the result at future time.

3 Fourier Neural Operator

Fourier Neural Operator (FNO) (Li et al., 2020 [1]) is a novel and state-to-the-art data-driven method which can learn mappings between infinite-dimensional spaces of functions. FNO is a resolution-invariant approaches and the model parameters can be shared across different discretizations and thus training on low resolution data for saving time and using on high resolution data is applicable.

Consider with a given bounded open domain D and Banach spaces of functions $\mathcal{A}$ and $\mathcal{U}$, the non-linear mapping between function spaces $G : \mathcal{A} \to \mathcal{U}$ is the solution operator of parametric PDE systems. Suppose $\{a_j, u_j = G(a_j)\}_{j=1}^N$ are i.i.d function pairs. The goal is to build a neural network $\hat{G}_\theta : \mathcal{A} \to \mathcal{U}$, parameterized with θ to approximate G. The objective is defined as follows where $\mathcal{L} : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ is the loss function:

$$\min_\theta \mathbb{E}_{a \in \mathcal{A}} \left[\mathcal{L}(\hat{G}_\theta(a), G(a)) \right] \tag{1}$$

To make functions a_j and u_j numerically applicable, we sample $a(D_j), u(D_j)$ to be our point-wise input-output evaluations, where $D_j = \{x_1, ..., x_n\} \subset D$ is the discretization of D with resolution n.

For a particular a, the crucial part of the approximation model $\hat{G}_\theta$ is the T iterations: $v_1 \mapsto ... \mapsto v_T$, each consists of a non-local kernel integral operator $\mathcal{K}$, which is parameterized by ϕ, and a local nonlinear activation function σ. The iteration $v_t \mapsto v_{t+1}$ is formally defined as follows, where W is a linear map:

$$v_{t+1}(x) = \sigma(W v_t(x) + (\mathcal{K}_\phi(v_t))(x)) \tag{2}$$

And the kernel integral operator $\mathcal{K}_\phi$ is defined as follow, where κ_ϕ is a periodic trainable neural network parameterized by ϕ:

$$(\mathcal{K}_\phi(v_t))(x) = \int_D \kappa_\phi(x, y) v_t(y)\, dy \tag{3}$$

Since κ_ϕ is a trainable neural network, by imposing $\kappa_\phi(x, y) = \kappa_\phi(x - y)$, we can have equation 3 to now become a convolution and can support the following modification of the kernel integral operator.

Consider a general function f and its Fourier transform $\mathcal{F}$ as well as the inverse $\mathcal{F}^{-1}$:

$$\mathcal{F}(f, k) = \int_D f(x) e^{-2i\pi <x,k>}\, dx, \quad \mathcal{F}^{-1}(f, k) = \int_D f(x) e^{2i\pi <x,k>}\, dx \tag{4}$$

By imposing κ_ϕ to be convolutional, we can use the convolution theorem and have:

$$\mathcal{F}(\mathcal{K}_\phi(v_t)) = \mathcal{F}(\int_D \kappa_\phi(x, y) v_t(y)\, dy) = \mathcal{F}(\kappa_\phi) \cdot \mathcal{F}(v_t) \tag{5}$$

By taking the inverse Fourier transform on both sides, we have:

$$\mathcal{K}_\phi(v_t) = \mathcal{F}^{-1}(\mathcal{F}(\kappa_\phi) \cdot \mathcal{F}(v_t)) \tag{6}$$

Since κ_ϕ is a trainable neural network, we denote $R_\phi = \mathcal{F}(\kappa_\phi)$ to be the trainable neural network.

As we assumed before that κ_ϕ is periodic, so is R_ϕ. Consider the complex Fourier series expansion, we can parameterize R_ϕ and $\mathcal{F}(v_t)$ to be a collection of truncated Fourier modes with the highest mode k_{max}. Therefore we can use Fast Fourier Transform to replace the multiplication of functions on the right hand side of equation 6.

For the whole model architecture, input data $a(x)$ will first be projected to higher dimensional representation by the transformation layer P, followed by T iterative Fourier layer $\{v_i\}_{i=1,\dots,T}$, and finally projected back to the output dimension by local transformation Q. The whole framework is summarized in figure 1.

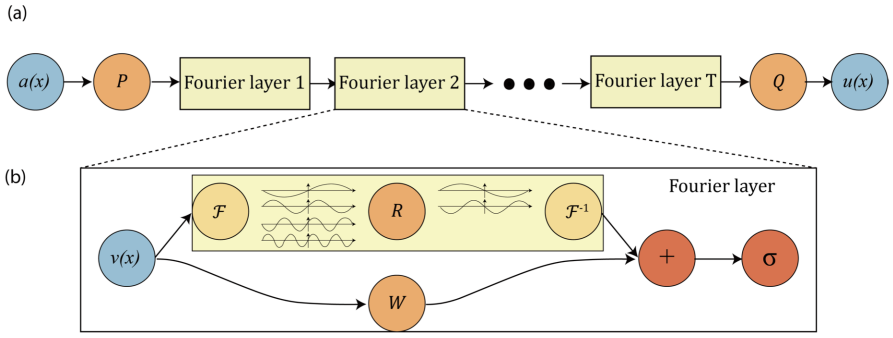


Figure 1: **top:** The whole framework of Fourier Neural Operator; **bottom:** Structure of a single Fourier Layer.

4 Results and Discussion

In this section, we will discuss the experimental results of Fourier Neural Operator evaluated in different PDE systems as well as different boundary conditions.

4.1 Experimentation Set-Up

In the FNO paper, its performance was tested on 1-D Burgers' Equation but only with zero Dirichlet boundary condition and zero Neumann boundary condition. Heat equation can be considered as a simplified case of Burgers' Equation, so our experiment will start with 1-D heat equation with non-zero Dirichlet boundary condition and non-zero Neumann boundary condition. A famous problem in physics is to understand the shapes of water waves and therefore wave equation is derived. 1-D wave equation, being one of the simplest second order PDE, will be

our second PDE system for experimentation, with non-zero Dirichlet boundary condition and non-zero Neumann boundary condition. Apart from that, we add random noise to each training input-output pair to test the robustness of FNO in front of the data corruption. More details of two PDE systems as well as their data generation processes are discussed in Appendices A.1.

In each experiment, our goal is to train an efficient PDE solution approximator that can solve the whole family of PDE systems instead of only one instance of them. We use the same Fourier neural operator framework as specified in FNO paper and in section 3 with $T = 4$ Fourier integral operator layers with the ReLU activation as well as batch normalization. For all experiments, we use $N = 1000$ training instances and 100 testing instances. We use Adam optimizer to train for 500 epochs with a learning rate of 0.001. Details of neural network architecture and other hyperparameters are listed in Appendix A.2.

4.2 Experimentation Results

Learning Speed In figure 2a, we plot the relative error changing, which is averaged over all training data, in each training epochs. The relative error for an input-output pair is defined to be the absolute difference between model's estimation and the grand true value of the PDE solution. As we can tell from the figure, the FNO can quickly learn the latent representation of the unknown PDE in all cases, as the relative error is smaller than 0.01 after first 100 epochs.

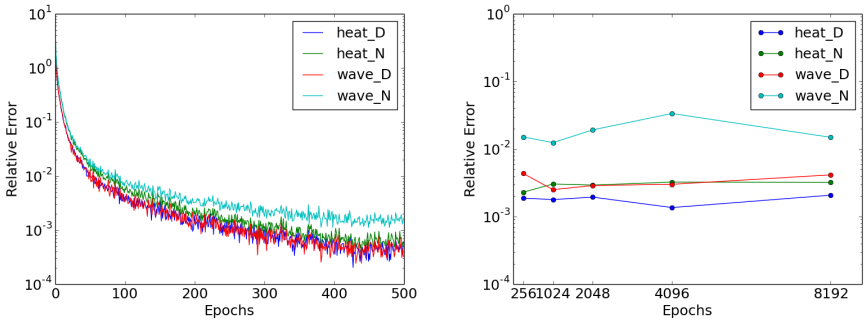
Resolution Consistency Apart from the fast training speed, FNO also shows great resolution consistency in all four cases. For each case, we all sample input-output pairs with highest resolution $n = 8192$ and than sub-sample to lower resolution to make sure the comparability. In figure 2b, we plot the averaged testing data relative error, result is consistent with the final training loss and the testing performance remains in the same level when training data is $32\times$ less, allowing FNO to be resolution-free and thus supporting zero-shot super-resolution: low-resolution trained and high-resolution evaluated.

4.3 Experimentation Analysis

We made experiments to evaluate FNO's performance in heat equation and wave equation with non-zero Dirichlet and Neumann boundary condition. FNO's fast speed of learning the latent representation of PDE and zero-shot super-resolution ability is justified in our experiment results.

However, in terms of some unexpected relative error increasing when resolution increases, one possible explanation could be the over-fitting to the data noise. As we add noise to all training data-points, such noise may accumulates when resolution increase – more corrupted data are used for finding the latent representation of PDE. Since FNO is capable of solving much more complicated PDE solutions, it may over-fit to these noises and influence the final testing performance.

So in the future, further study related to FNO's performance v.s. the scale of noise may be conducted to examine whether FNO is over-sensitive and easy



(a) Relative Err v.s. Epoch (b) Relative Err v.s. Resolution

Figure 2: Experiment Results: (a) Relative error over all training data are averaged and plotted v.s. the number of training epochs; FNO achieves smaller than 0.01 relative error in the first 100 epochs for all experiment tasks. (b)Relative error over all testing data are averaged and plotted v.s. resolution; FNO shows performance robustness when training data is 32× less, justifying its zero-shot super-resolution ability

to over-fit especially in relative simple PDE systems.

5 Conclusion

Partial Differential Equations (PDE) is an important tool for us to explore and exploit some fundamental process of our natural world. However, how to find or approximate the latent representation of unknown PDE systems remains an open question. Fourier neural network (FNO) is a data-driven method which can simulate the hidden PDE solution with limited data in hand. In this independent study, experiments are made in two most basic PDE equations: heat equation and wave equation with non-zero Dirichlet and Neumann boundary condition to evaluate FNO's performance. In the experiment, FNO shows its fast learning speed as well as mesh-free training ability, but may have potential over-fitting problems.

References

[1] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. M. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," *CoRR*, vol. abs/2010.08895, 2020. [Online]. Available: https://arxiv.org/abs/2010.08895

[2] E. Yeung, S. Kundu, and N. O. Hodas, "Learning deep neural network representations for koopman operators of nonlinear dynamical systems," *CoRR*, vol. abs/1708.06850, 2017. [Online]. Available: http://arxiv.org/abs/1708.06850

[3] K. Wu and D. Xiu, "Data-driven deep learning of partial differential equations in modal space," *Journal of Computational Physics*, vol. 408, p. 109307, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999120300814

[4] P. Grohs and L. Herrmann, "Deep neural network approximation for high-dimensional elliptic pdes with boundary conditions," 2020. [Online]. Available: https://arxiv.org/abs/2007.05384

[5] N. B. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. M. Stuart, and A. Anandkumar, "Neural operator: Learning maps between function spaces," *CoRR*, vol. abs/2108.08481, 2021. [Online]. Available: https://arxiv.org/abs/2108.08481

[6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[7] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999118307125

[8] W. Strauss, *Partial Differential Equations: An Introduction*. Wiley, 2007. [Online]. Available: https://books.google.com/books?id=PihAPwAACAAJ

[9] L. Lu, P. Jin, and G. E. Karniadakis, "Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators," *CoRR*, vol. abs/1910.03193, 2019. [Online]. Available: http://arxiv.org/abs/1910.03193

[10] K. Bhattacharya, B. Hosseini, N. B. Kovachki, and A. M. Stuart, "Model reduction and neural networks for parametric pdes," 2020. [Online]. Available: https://arxiv.org/abs/2005.03180

[11] Y. KHOO, J. LU, and L. YING, "Solving parametric PDE problems with artificial neural networks," *European Journal of Applied Mathematics*, vol. 32, no. 3, pp. 421–435, jul 2020. [Online]. Available: https://doi.org/10.1017%2Fs0956792520000182

A Appendix

A.1 Data Generation

In this section, we will provide more details about the PDEs used in the experimentation as well as their data generation process.

A.1.1 Heat Equation

Recall the 1-D heat equation limited on $x \in [0, 1]$:

$$\begin{aligned} u_t &= \alpha u_{xx}, \quad x \in (0,1), t > 0 \\ u(x,0) &= \phi(x), \quad x \in (0,1) \end{aligned} \tag{7}$$

For simplicity, we set coefficient $\alpha = 1$ and initial condition $\phi(x)$ to be the most typical periodic function which is a linear combination of sine and cosine function:

$$\phi(x) = Asin(k_1 \pi x) + Bcos(k_2 \pi x) \tag{8}$$

where A,B,k_1, k_2 are random variables in (0,10) with uniform distribution. Besides, we experiment with non-zero constant Dirichlet boundary condition and Neumann boundary condition, they are also set to be random variable in (0,10). Grand true solution is computed using exact formula in Strauss's book (Strauss, 2007 [8]).

For data generation, we first use highest resolution $n = 8192$ then subsampled to other resolutions. We sampled $N = 1000$ training data-points and 100 testing data-points. For each training data-points, noise ϵ is added according to normal distribution $\epsilon \sim \mu = \mathcal{N}(0, 10^{-3})$.

A.1.2 Wave Equation

Recall the 1-D heat equation limited on $x \in [0, 1]$:

$$\begin{aligned} u_{tt} &= c^2 u_{xx}, \quad x \in (0,1), t > 0 \\ u(x,0) &= \phi(x), \quad x \in (0,1) \\ u_t(x,0) &= \psi(x), \quad x \in (0,1) \end{aligned} \tag{9}$$

For simplicity, we set coefficient $c = 1$ and initial condition $\phi(x)$ and $\psi(x)$ to be the most typical periodic functions which are linear combinations of sine and cosine function:

$$\phi(x) = Asin(k_1 \pi x) + Bcos(k_2 \pi x) \tag{10}$$

$$\psi(x) = Csin(k_3 \pi x) + Dcos(k_4 \pi x) \tag{11}$$

where A,B,C,D,k_1, k_2, k_3, k_4 are random variables in (0,10) with uniform distribution. Besides, we experiment with non-zero constant Dirichlet boundary condition and Neumann boundary condition, they are also set to be random variable in (0,10). Grand true solution is computed using exact formula in Strauss's book (Strauss, 2007 [8]).

For data generation, we first use highest resolution $n = 8192$ then subsampled to other resolutions. We sampled $N = 1000$ training datapoints and 100 testing datapoints. For each training datapoints, noise ϵ is added according to normal distribution $\epsilon \sim \mu = \mathcal{N}(0, 10^{-3})$.

A.2 Experimentation Model Hyperparameters

For the model structure used in experimentation, it consist of one local transformation P which projects the input data to dimension 128, then stack four Fourier layers with ReLU activation as well as batch normalization, and finally a local transformation Q project to the output dimension 1.

For each experiment, we set number of training data $N_{train} = 1000$ and number of testing data $N_{test} = 100$, batch size = 20, in total 500 training epochs, learning rate $\gamma = 0.001$, highest Fourier mode $k_{max} = 16$.