DLSC Project B: FBPINNs

Zhenyu Zhang

15.07.2023

1 Introduction

Physics-informed neural networks (PINNs) [1] show great potential in solving differential equations by neural networks, with the abundant research reference in the area of deep learning and traditional numerical methods. However, on large problem domains, PINNs are tuned to neural networks with large size, which requires an increasing number of sampling points and thus consumes large computational costs. Meanwhile, the spectral bias [3] shows the neural networks learn slowly on tasks with higher frequencies than those with low frequencies. These issues limit the applicable areas for PINNs.

Based on the domain decomposition methods, Finite Basis PINNs (FBPINNs)[2] are proposed to solve problems with multi-scale and large domains. The decomposition of the domains lowers the frequency of the problems in the subdomains, while the continuity of the approximation solutions is guaranteed by the coupled training on the overlapped subdomains. In this Project, I implement FBPINNs methods based on the framework in DLSC tutorials, and test the performance of FBPINNs on the multi-scale problems (the baseline requirement), multis-scale problems with high frequency problems (the extension a), and the 2D sinusoidal problems. FBPINNs show better fitting accuracy than PINNs.

2 FBPINNs method

In the workflow of FBPINNs shown in Figure 2, the problem domain is decomposed into a group of overlapped subdomains. This project deploys uniform decomposition. After the uniform division, the overlapped intervals are constructed to be centered on the boundaries of each uniform area. The number of points in each center area and the number of points in each overlapped area should be as equal as possible to avoid local overfitting.

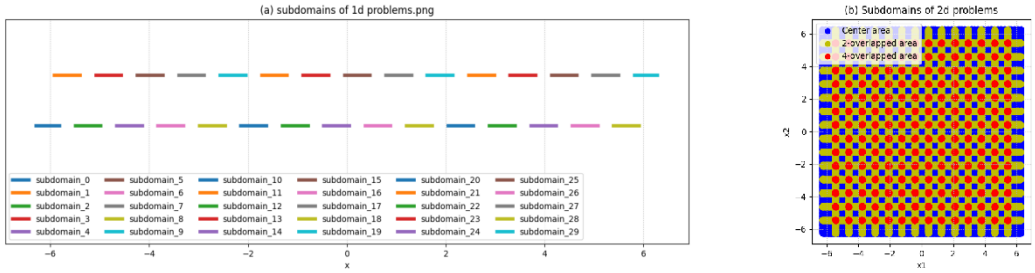

Figure 1 The decomposition of the domains for 1D and 2D problems

The points are initially sampled over the whole domain by generating from the Sobol sequence. Then the points are classified into different lists for the center area or the overlapped area by the order of their corresponding intervals. On each subdomain, there is one neural network approximating the solution. However, the representations differ on the center and overlapped area. The input training points x are first normalized from the subdomain into [-1,1], then forward propagated through the neural network and unnormalized with a certain scale. On overlapped subdomains, the identical training points are in marginal area in different subdomains, and thus each output is weighted

summed up with multiplying a window function, which is derived from the sigmoid function and aims at restricting the output inside the corresponding subdomains.

$$\overline{NN}(x;\theta) = \sum_{i}^{n} w_i(x) \cdot unnorm \circ NN_i \circ norm_i(x) \tag{1}$$

$$w_i(x) = \prod_{j}^{d} \phi\left(\left(x^j - a_i^j\right)/\sigma_i^j\right) \phi\left(\left(b_i^j - x^j\right)/\sigma_i^j\right) \tag{2}$$

Actually in my training policy, due to the initial division the points are definitely located in the corresponding area, the window function plays a more important role in regularizing the marginal training points by providing gradients that vary with the positions.

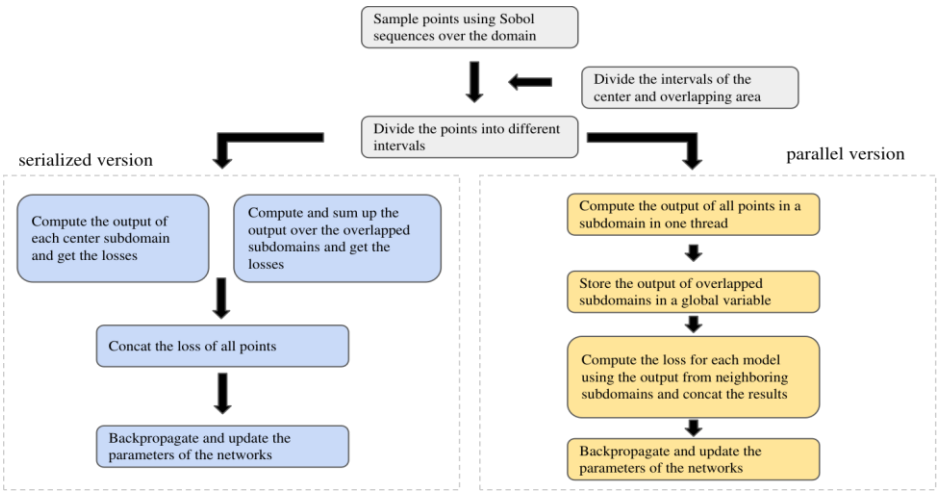

Figure 2 Worflow of FBPINNs in this project

Both the output of the center area and the overlapped area are then processed by the Operator C, which adds a strong limit to the output for satisfying the spatial boundary conditions.

$$\hat{u}(x;\theta) = C\left[\overline{NN}(x;\theta)\right] \tag{3}$$

Consequently the residuals for the spatial boundary conditions are no longer needed in FBPINNs workflow. Finally the loss functions are optimized by ADAM optimizer and the parameters of the models are updated.

2D problems are confronted with the high computational costs from the large number of neural networks due to the complex subdomain division in high dimensions. For high performance, the parallel version of FBPINNs processes one network by one thread. In this method, all points in one subdomain are first forward computed and the results are stored in the global variables. Then the neighbour outputs are collected by each thread to compute the loss over the overlapped area. Finally the losses are concatenated together for optimization.

3 Multi-scale problem (baseline)

This case contains sine formatted solution functions,

$$\frac{du}{dx} = \sum_{i=1}^{n} \omega_i \cos(\omega_i x), u(0) = 0 \tag{4}$$

The constraining operator is $\tanh(\omega_{\text{highest}} x)$ .The overlapping area has a length of 0.3, and 9000 points are sampled over the whole domain. The neural networks identically have 2 layers and 16 neurons. As a comparison. PINNs with 2 layers 16 neurons, 4 layers 64 neurons and 5 layers 128 neurons are trained.

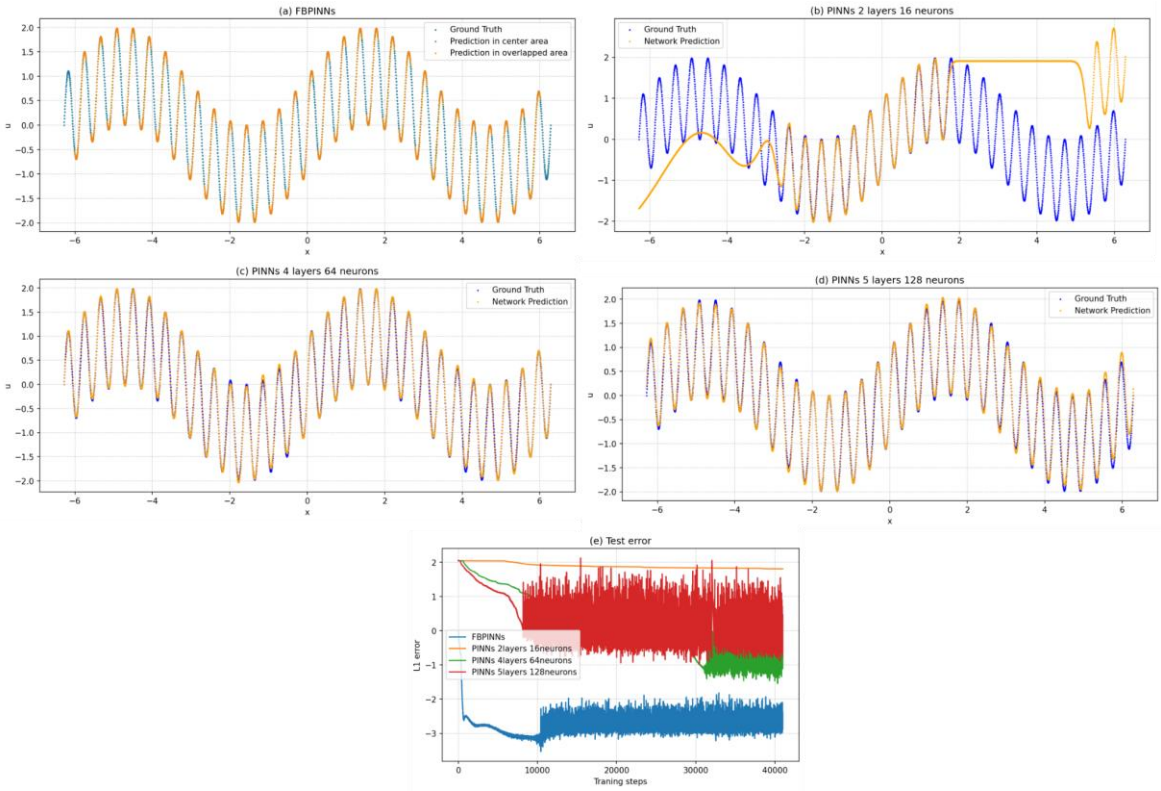

Figure 3 Fitting results of FBPINNs and PINNs

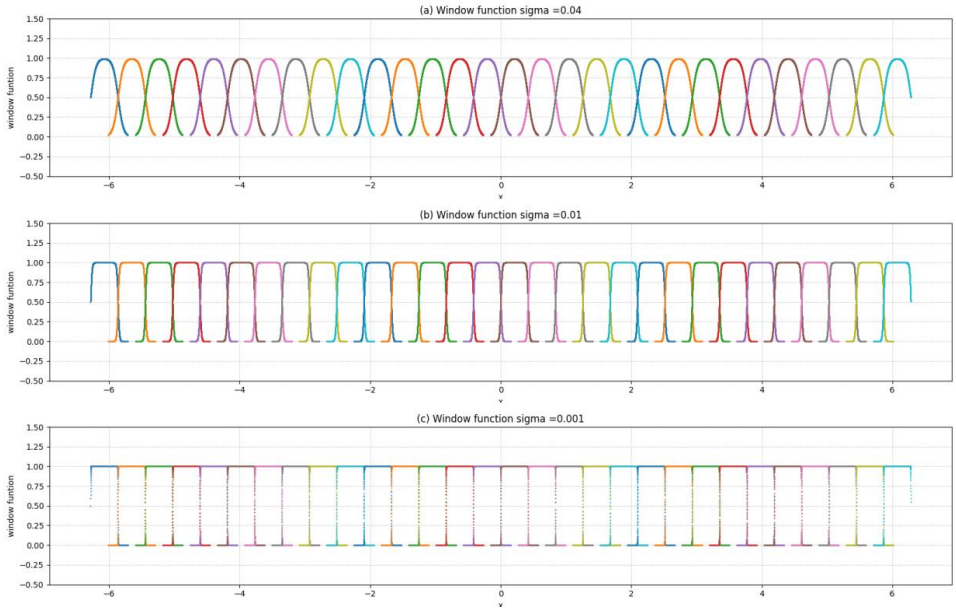

Figure 4 Window functions with different sigma values

According to Fig. 3, the results show FBPINNs have fast convergence and more accurate fitting capabilities, and PINNs show underfitting in the marginal area. The window function parameters also affect the fitting result (shown in Fig. 4). Excessively low sigma value leads to extremely high gradients to the overlapped training points, while high sigma could not regularize the subdomain.

4 Complex multi-scale problem (Extension a)

This case has the similar format with the previous set, but set $n = 5$, $\omega^i = 2^i$. Since the highest frequency of the components determines the whole frequency of the solution, the domains are decomposed into 64 subdomains. The width of the overlapped area is adjusted to 0.12 and 19200 training points are sampled. In comparison, FBPINNs achieve convergence within 10000 epochs of optimization, while PINNs take 4 times longer. Similarly, FBPINNs could fit the solutions with high accuracy while PINNs.

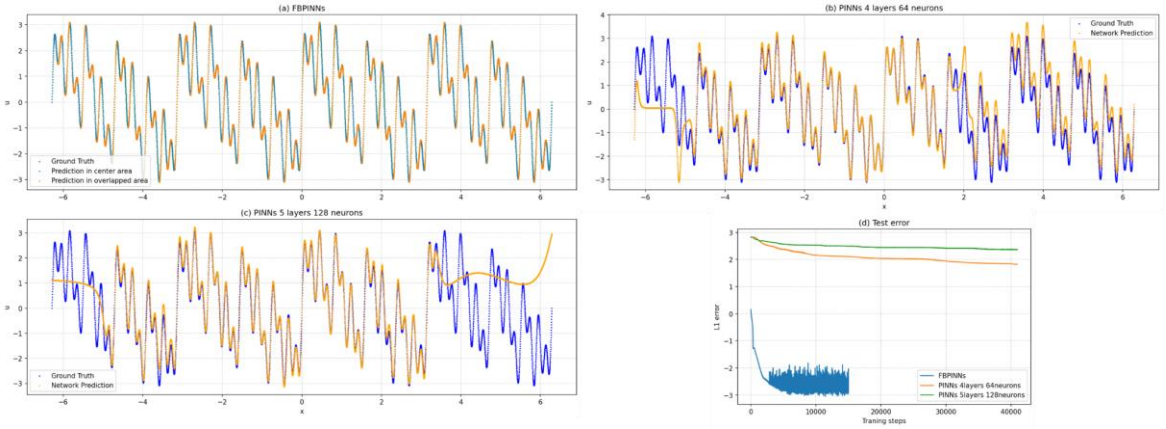

Figure 5 Fitting results of FBPINNs and PINNs

5 2D Sinusoidal problem

The problem set is

$$\frac{\partial u}{\partial x_1} + \frac{\partial u}{\partial x_2} = \cos(15x_1) + \cos(15x_2), u(0, x_2) = \frac{1}{\omega}\sin(15x_2) \tag{5}$$

The operator is

$$\hat{u}(x;\theta) = \frac{1}{15}\sin(15x_2) + \tanh(15x_1) \cdot \overline{NN}(x_1; x_2; \theta)$$

The domain is divided into 225 subdomains, with an overlapped width of 0.6. $900 \times 900 = 810000$ training points are sampled over the whole domain. In the parallel implementation, Python multithreading module is deployed, but the speed is not increased, which might result from the global interpreter lock (GIL) of Python. The fitting results show relatively nice capability on this case.

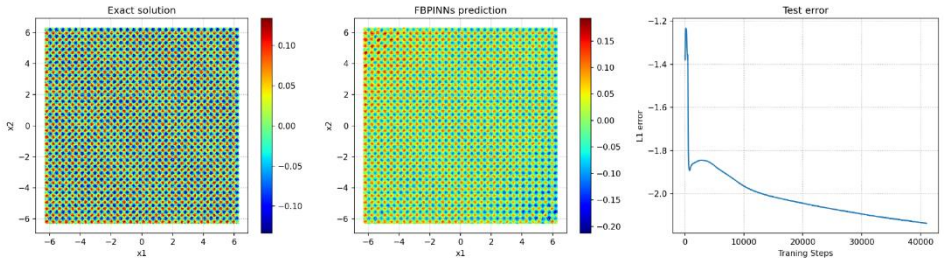

Figure 6 Fitting results of FBPINNs on 2D problem

6 Reference

[1] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378:686–707, 2019. ISSN 10902716

[2] Ben Moseley, Andrew Markham, and Tarje Nissen-Meyer. Finite Basis Physics-Informed Neu ral Networks (FBPINNs): a scalable domain decomposition approach for solving differential equations. arXiv, jul 2021.

[3] Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. Towards Understanding the Spectral Bias of Deep Learning, dec 2019.