# Scaling PINNs to high-frequency and multiscale problems using domain decomposition


Ben Moseley

Postdoctoral fellow, ETH Zürich AI Center /
Computational Applied Mathematics Lab

In collaboration with:

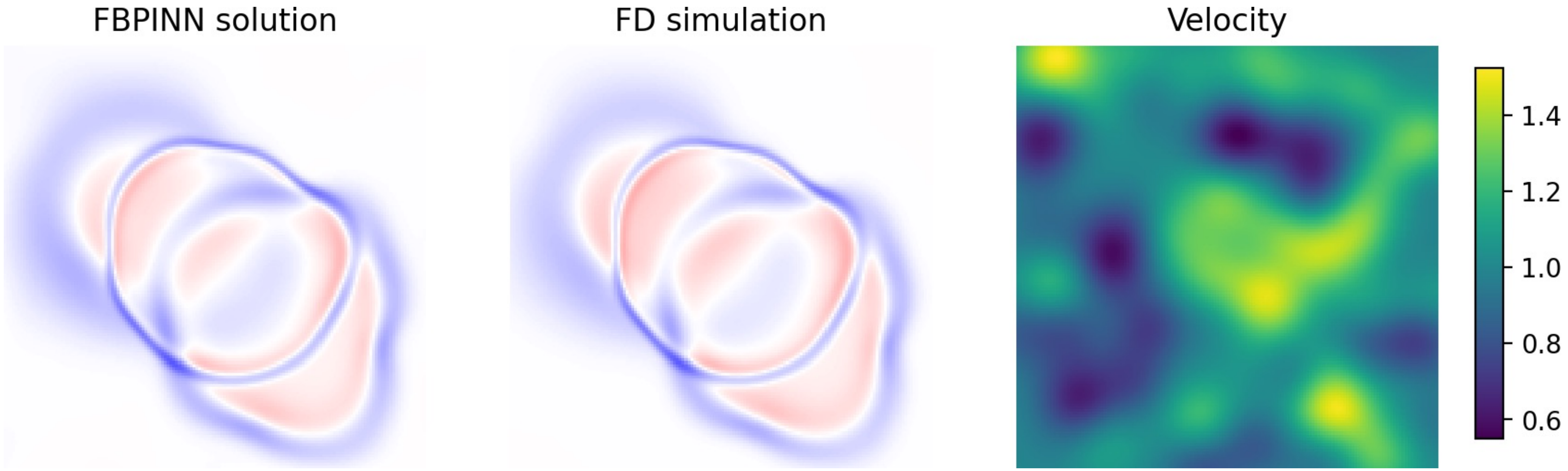
Prof. Siddhartha Mishra, ETH Zürich AI Center / CAMLab
Prof. Victorita Dolean Maini, University of Strathclyde
Prof. Alexander Heinlein, Delft University of Technology
Prof. Tarje Nissen-Meyer, University of Oxford
Prof. Andrew Markham, University of Oxford

# ETH Zürich Deep Learning in Scientific Computing Master's course 2023


[youtube.com/@**CAMLabETHZurich**](https://youtube.com/@CAMLabETHZurich)

**13 lectures, 20+ hours, 10k+ views**

# High frequency / multiscale simulation with PINNs




Solving the 2+1D acoustic wave equation:

$$\nabla^2 u(x,t) - \frac{1}{c(x)^2}\frac{\partial^2 u(x,t)}{\partial t^2} = 0$$

# Publications

**Finite basis physics-informed neural networks (FBPINNs): a scalable domain decomposition approach for solving differential equations**

Moseley, B., Markham, A., Nissen-Meyer, T., ACM (2023).

https://arxiv.org/abs/2107.07871

**Multilevel domain decomposition-based architectures for physics-informed neural networks**

Dolean, V., Heinlein, A., Mishra, S., Moseley, B. (2023) (under review).

https://arxiv.org/abs/2306.05486

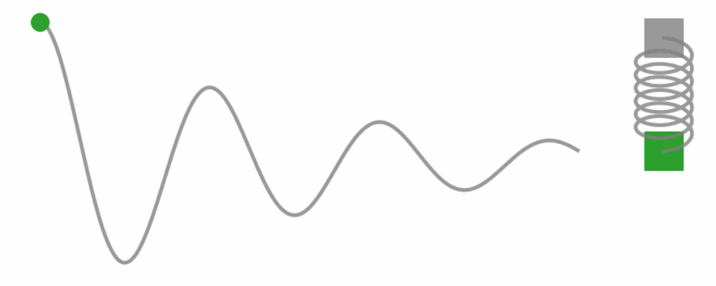
Code: github.com/**benmoseley/FBPINNs**

# Contents

- Why is it challenging to scale PINNs to high-frequency / multiscale problems?
- A potential solution: PINNs + domain decomposition
- Finite basis physics-informed neural networks (FBPINNs)
  - Improving scalability with multilevel modelling
  - Improving scalability with parallelisation / subdomain scheduling
- Future work

# What is a physics-informed neural network (PINN)?

Problem: damped harmonic oscillator

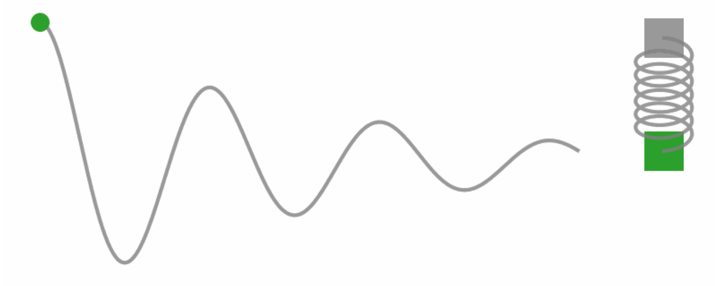


$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$

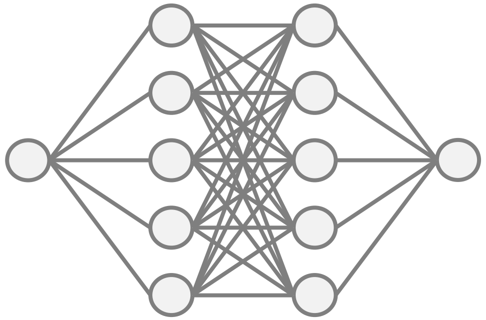
Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)
Lagaris et al, Artificial neural networks for solving ordinary and partial differential equations, IEEE (1998)

# What is a physics-informed neural network (PINN)?

Problem: damped harmonic oscillator



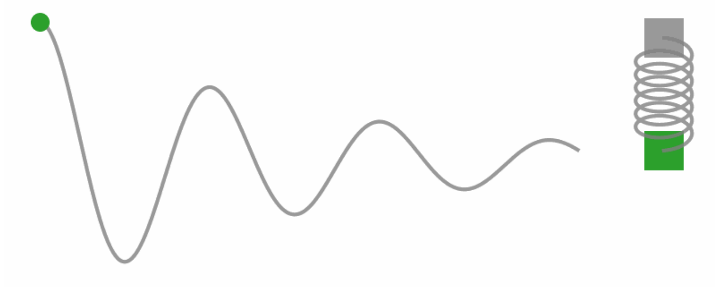
$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$



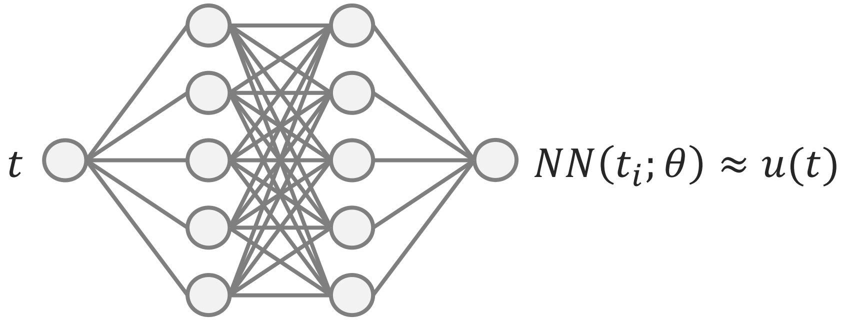

Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)
Lagaris et al, Artificial neural networks for solving ordinary and partial differential equations, IEEE (1998)

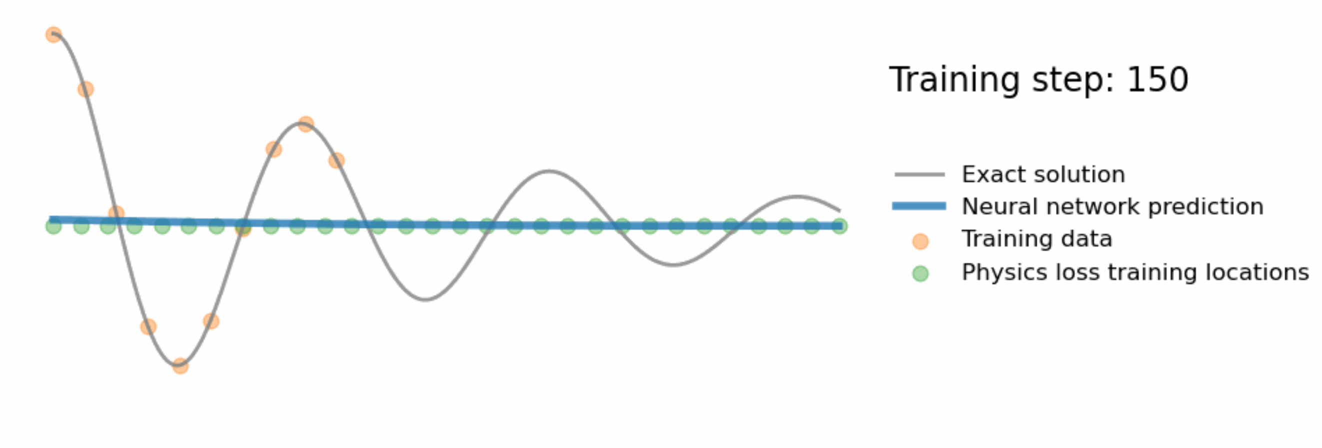
# What is a physics-informed neural network (PINN)?

Problem: damped harmonic oscillator



$$m\frac{d^2u}{dt^2} + \mu\frac{du}{dt} + ku = 0$$




$$L(\theta) = \frac{1}{N}\sum_{i}^{N}(NN(t_i;\theta) - u_i)^2$$ Boundary loss

$$+ \frac{\lambda}{M}\sum_{j}^{M}\left(\left[m\frac{d^2}{dt^2} + \mu\frac{d}{dt} + k\right]NN(t_j;\theta)\right)^2$$ Physics loss


Raissi et al, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, JCP (2018)
Lagaris et al, Artificial neural networks for solving ordinary and partial differential equations, IEEE (1998)


ETH zürich

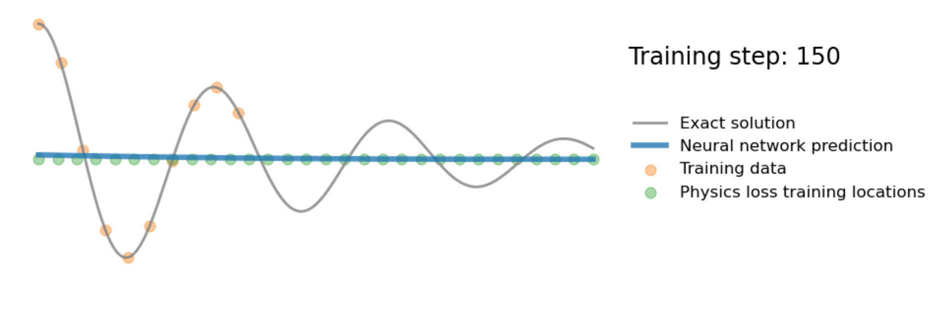
# What is a physics-informed neural network (PINN)?

**Advantages of PINNs**

- **Mesh-free**
- Can jointly solve forward and inverse problems
- Often performs well on “**messy**” problems (where some observational data is available)
- Mostly **unsupervised**
- Can perform well for high-dimensional PDEs

**Limitations of PINNs**

- **Computational cost** often high (especially for forward-only problems)
- Can be hard to **optimise**
- Challenging to **scale** to high-frequency, multi-scale problems



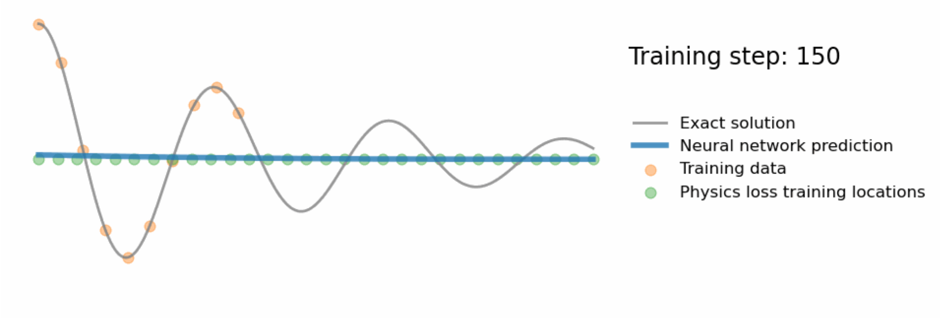
# What is a physics-informed neural network (PINN)?

## Advantages of PINNs

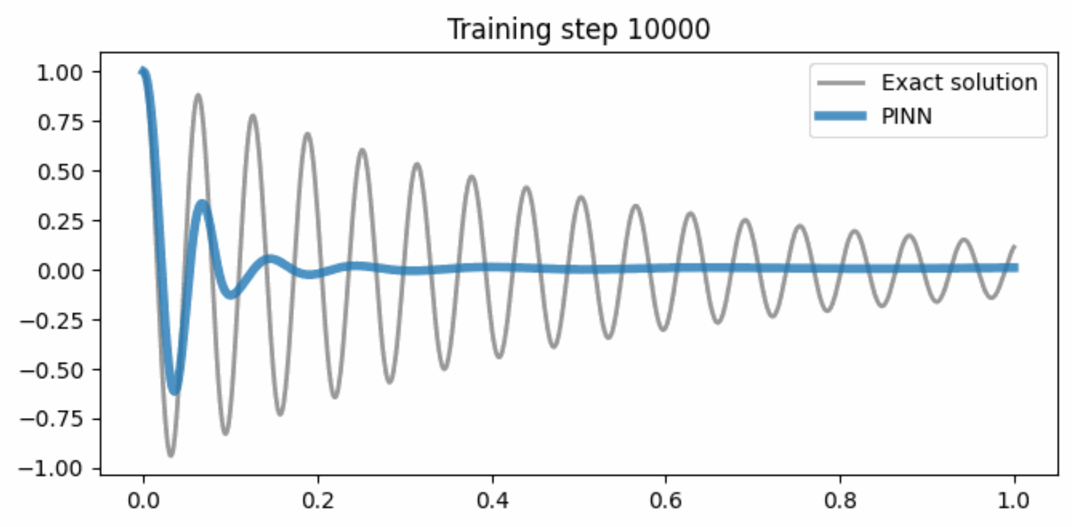
- **Mesh-free**
- Can jointly solve forward and inverse problems
- Often performs well on "**messy**" problems (where some observational data is available)
- Mostly **unsupervised**
- Can perform well for high-dimensional PDEs

## Limitations of PINNs

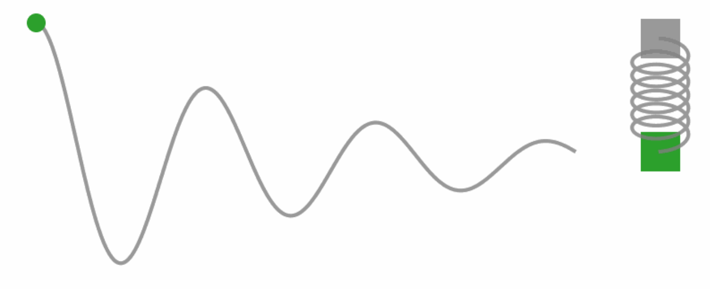
- **Computational cost** often high (especially for forward-only problems)
- Can be hard to **optimise**
- Challenging to **scale** to high-frequency, multi-scale problems



# Scaling PINNs to high frequency / multiscale problems

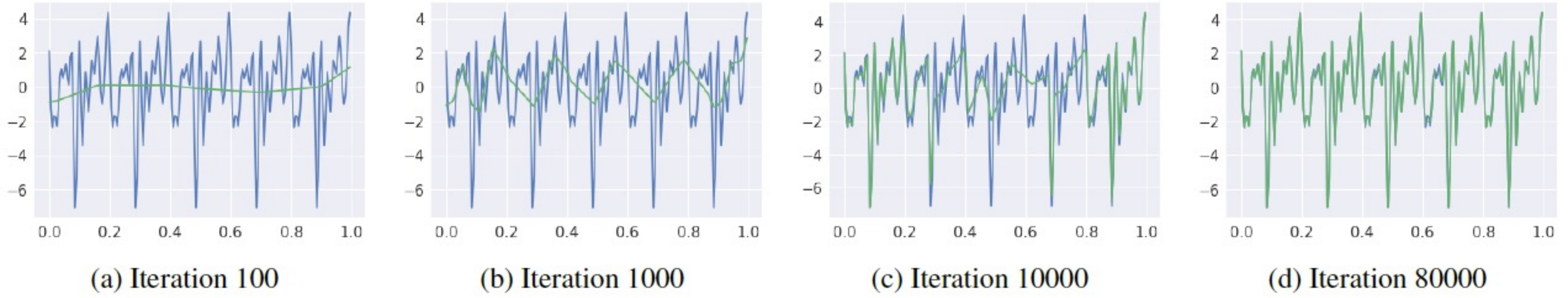


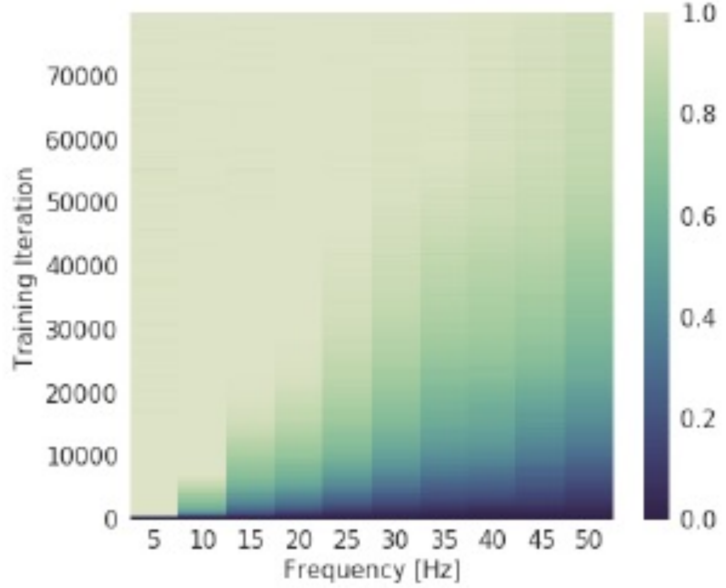

Problem: PINNs **struggle** to solve high-frequency / multiscale problems



Damped harmonic oscillator

# Spectral bias issue



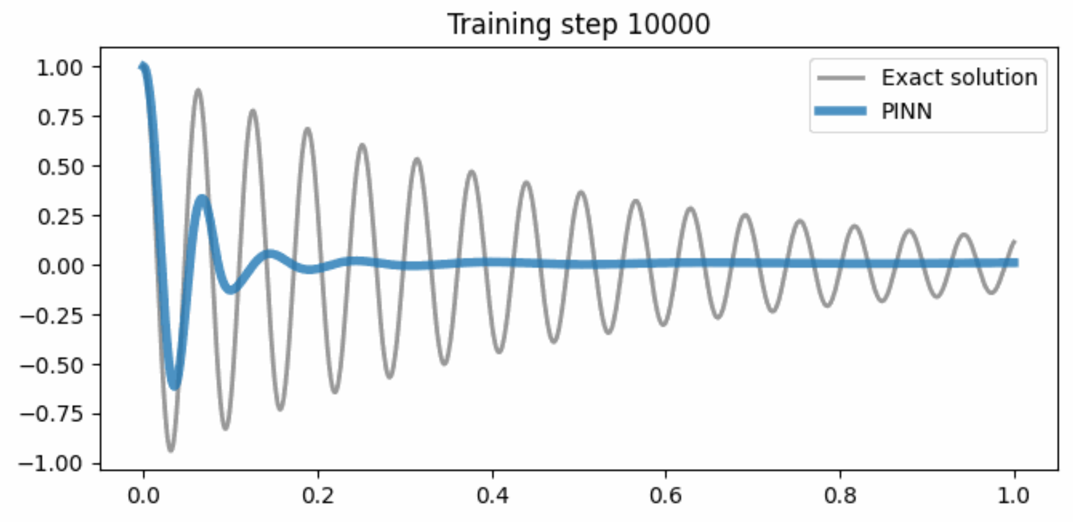




NNs prioritise learning **lower** frequency functions first

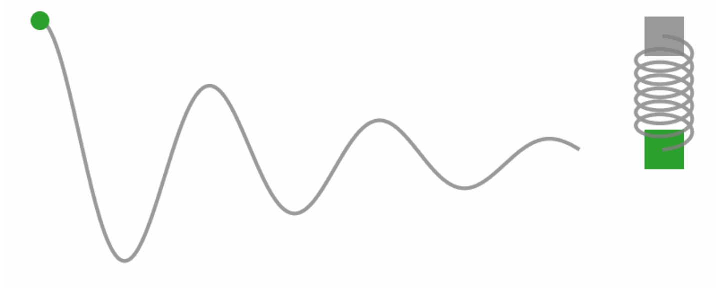
Under certain assumptions can be proved via neural tangent kernel theory


Rahaman, N., et al, On the spectral bias of neural networks. 36th International Conference on Machine Learning, ICML (2019)

# Scaling PINNs to high frequency / multiscale problems




Problem: PINNs **struggle** to solve high-frequency / multiscale problems



Damped harmonic oscillator

As higher frequencies are added:

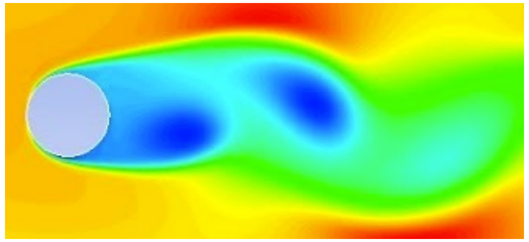
- More collocation points required
- Larger neural network required
- Spectral bias slows convergence

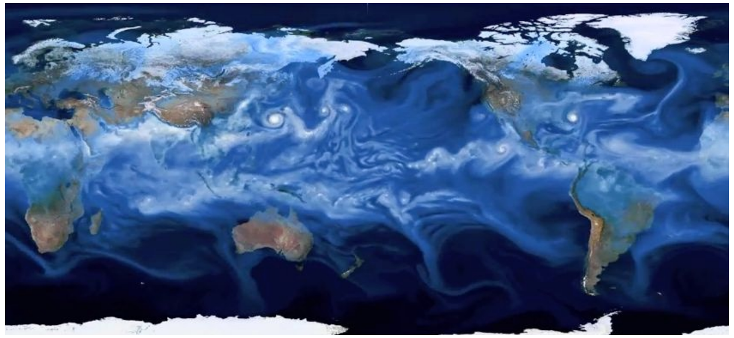
Leading to a significantly **harder** PINN optimization problem

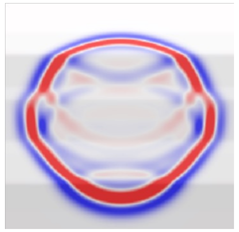
# Scaling PINNs to high frequency / multiscale problems

The majority of PINN papers focus on solving problems with **limited** frequency ranges / small domains

Goal: how can we **scale** PINNs to solve real-world problems?







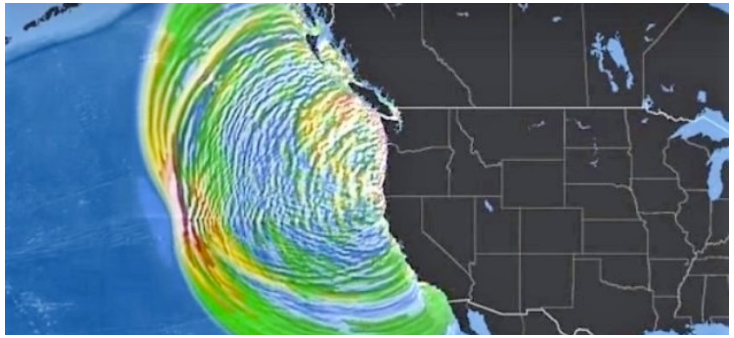
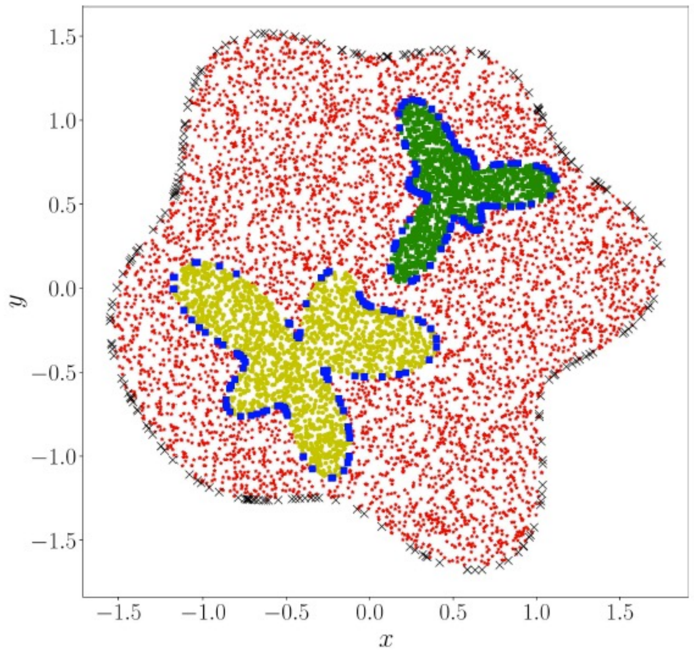


Image credits: Lawrence Berkeley National Laboratory / NOAA / NWS / Pacific Tsunami Warning Center

# PINNs + domain decomposition




Jagtap, A., et al., Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Communications in Computational Physics (2020)

Idea:

Take a “**divide-and-conquer**” strategy to model more complex problems:

1. Divide modelling domain into many smaller **subdomains**

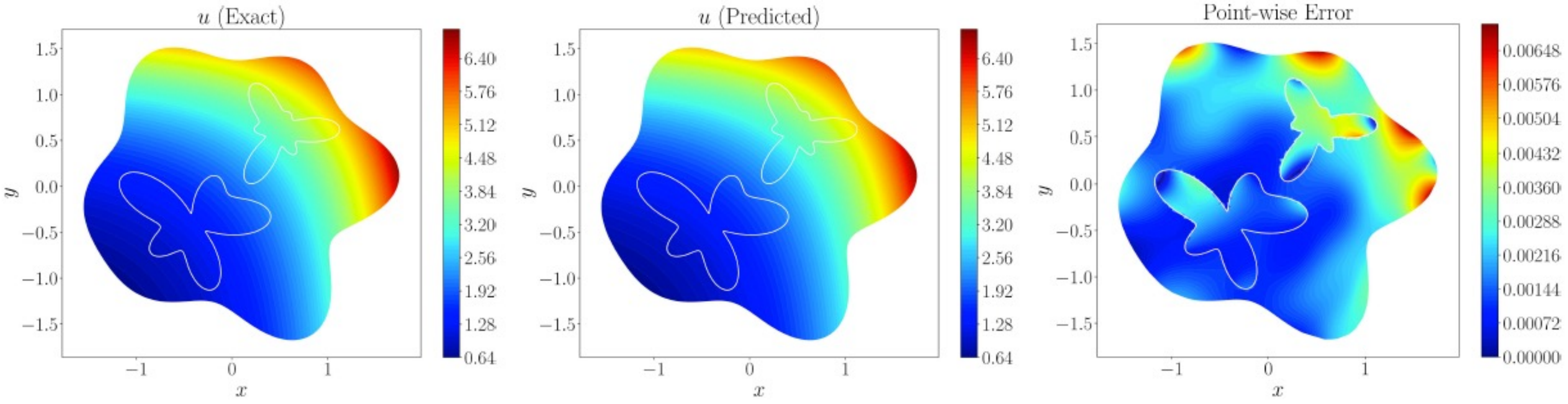
2. Use a separate neural network in each subdomain to model the solution

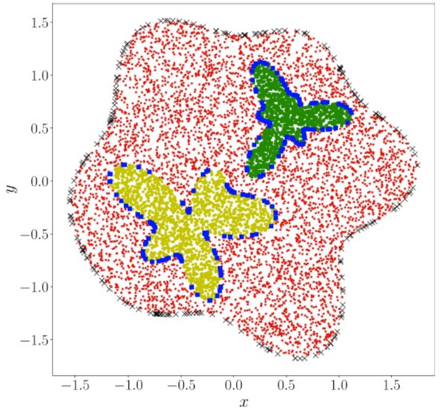
Hypothesis:

The resulting (coupled) local optimization problems are easier to solve than a single global problem

# XPINNs

XPINNs solving 2D Poisson's equation



$$L(\theta_m) = \frac{\lambda_b}{N_b}\sum_{i}^{N_b}(NN(x_i;\theta_m) - u_i)^2$$ Boundary loss

$$+ \frac{\lambda_p}{N_p}\sum_{j}^{N_p}\left(R\left(NN(x_j;\theta_m)\right)\right)^2$$ Physics loss

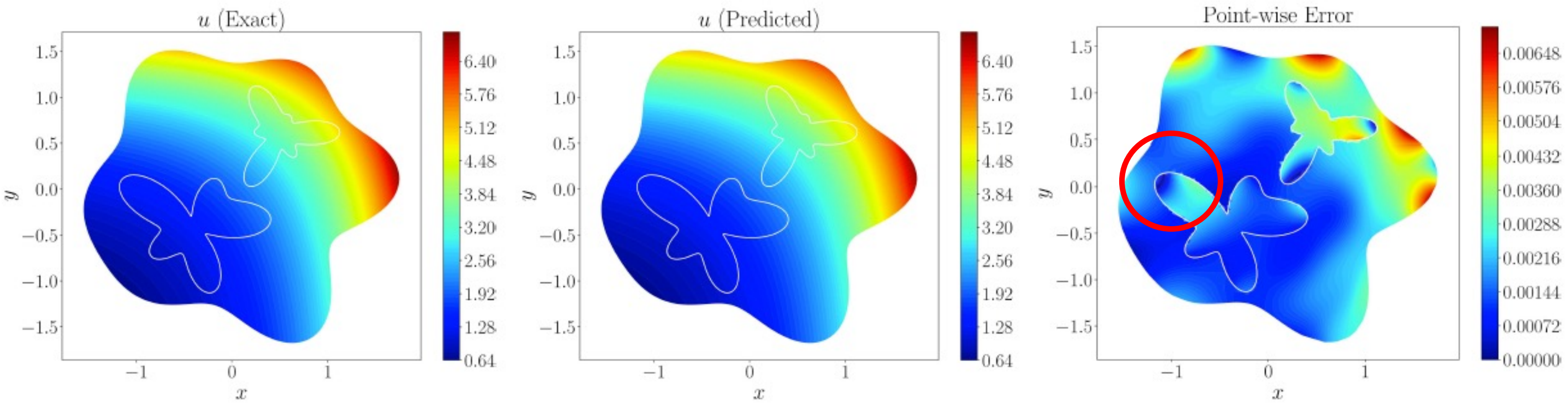
$$+ \sum_{l}^{N_n}\frac{\lambda_l}{N_l}\sum_{k}^{N_l}\left(NN(x_k;\theta_l) - NN(x_k;\theta_m)\right)^2$$ Interface conditions

$l \in \text{Neighbours}(m)$

Jagtap, A., et al., Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Communications in Computational Physics (2020)

# Limitations of XPINN-like strategies

XPINNs solving 2D Poisson's equation




Limitations:

- Introduces discontinuities in solution at subdomain interfaces
- Requires extra loss terms

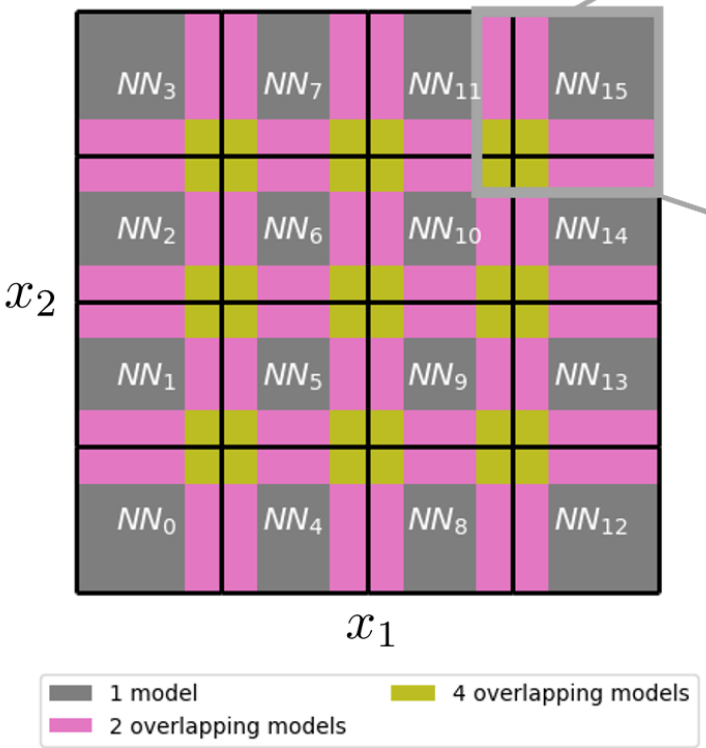
$$L(\theta_m) = \frac{\lambda_b}{N_b}\sum_{i}^{N_b}(NN(x_i;\theta_m) - u_i)^2$$ Boundary loss

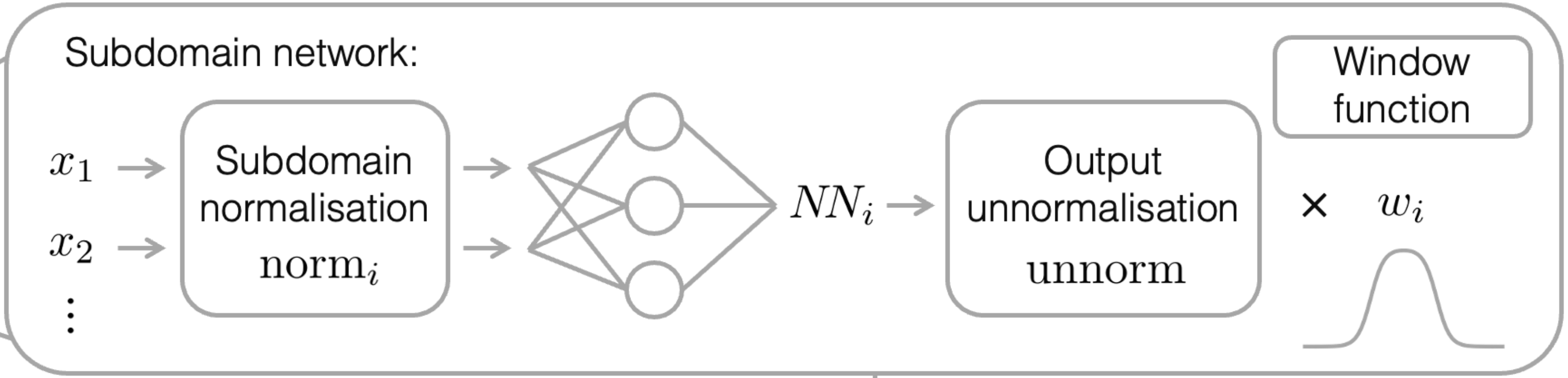
$$+ \frac{\lambda_p}{N_p}\sum_{j}^{N_p}\left(R\left(NN(x_j;\theta_m)\right)\right)^2$$ Physics loss

$$+ \sum_{\substack{l \\ l \in \text{Neighbours}(m)}}^{N_n}\frac{\lambda_l}{N_l}\sum_{k}^{N_l}\left(NN(x_k;\theta_l) - NN(x_k;\theta_m)\right)^2$$ Interface conditions

Jagtap, A., et al., Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Communications in Computational Physics (2020)

# Finite basis PINNs (FBPINNs)




FBPINN solution:

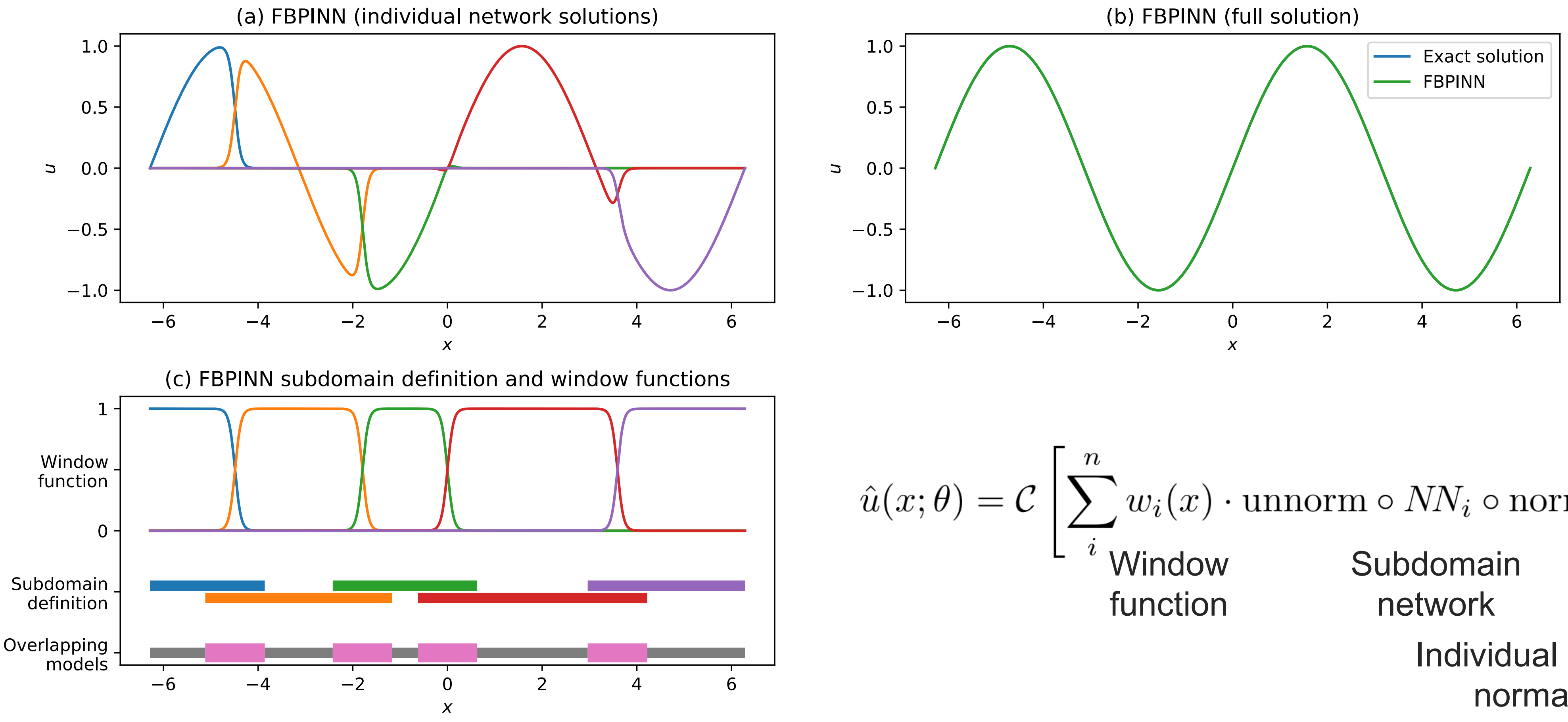
$$\hat{u}(x;\theta) = \mathcal{C}\left[\sum_i^n w_i(x) \cdot \mathrm{unnorm} \circ NN_i \circ \mathrm{norm}_i(x)\right]$$

Summation over all subdomain networks

Moseley et al, Finite Basis Physics-Informed Neural Networks (FBPINNs): a scalable domain decomposition approach for solving differential equations, ArXiv (2021), ACM (2023)

Idea: use **overlapping** subdomains and a **globally** defined solution ansatz

# FBPINNs in 1D




$$\hat{u}(x;\theta) = \mathcal{C}\left[\sum_{i}^{n} w_i(x) \cdot \mathrm{unnorm} \circ NN_i \circ \mathrm{norm}_i(x)\right]$$

Window function

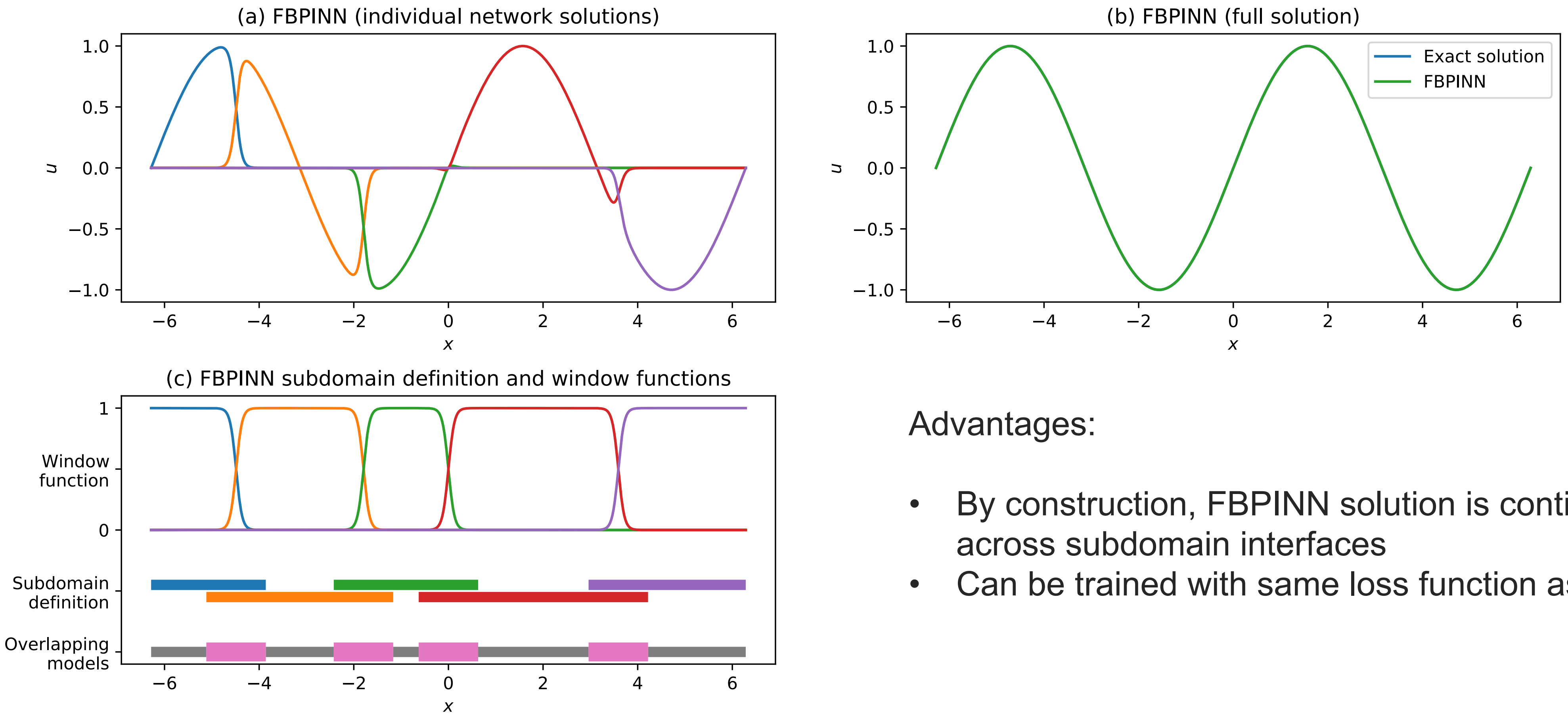
Subdomain network

Individual subdomain normalisation

Moseley et al, Finite Basis Physics-Informed Neural Networks (FBPINNs): a scalable domain decomposition approach for solving differential equations, ArXiv (2021), ACM (2023)

Idea: use **overlapping** subdomains and a **globally** defined solution ansatz

# FBPINNs in 1D

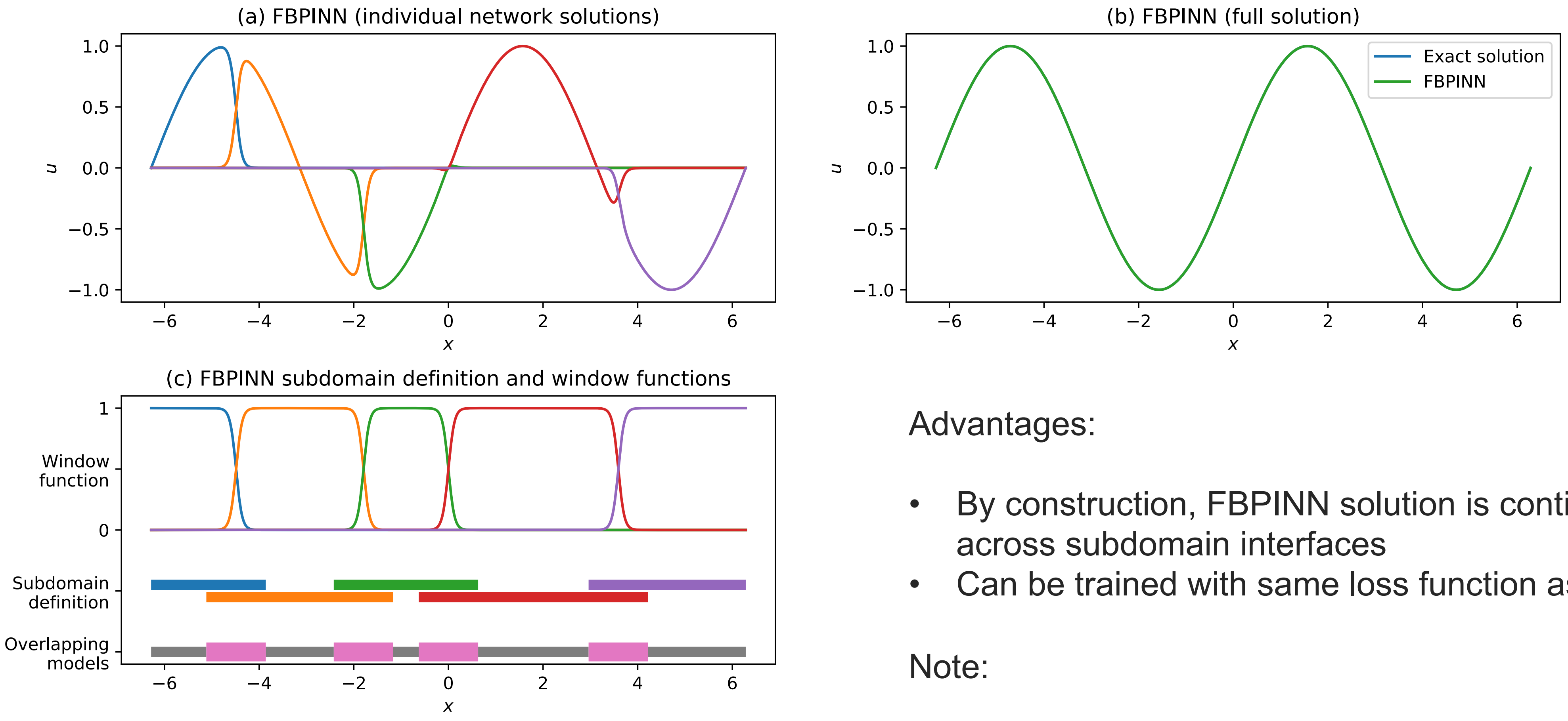



Advantages:

- By construction, FBPINN solution is continuous across subdomain interfaces
- Can be trained with same loss function as PINNs

Moseley et al, Finite Basis Physics-Informed Neural Networks (FBPINNs): a scalable domain decomposition approach for solving differential equations, ArXiv (2021), ACM (2023)

# FBPINNs in 1D




Advantages:

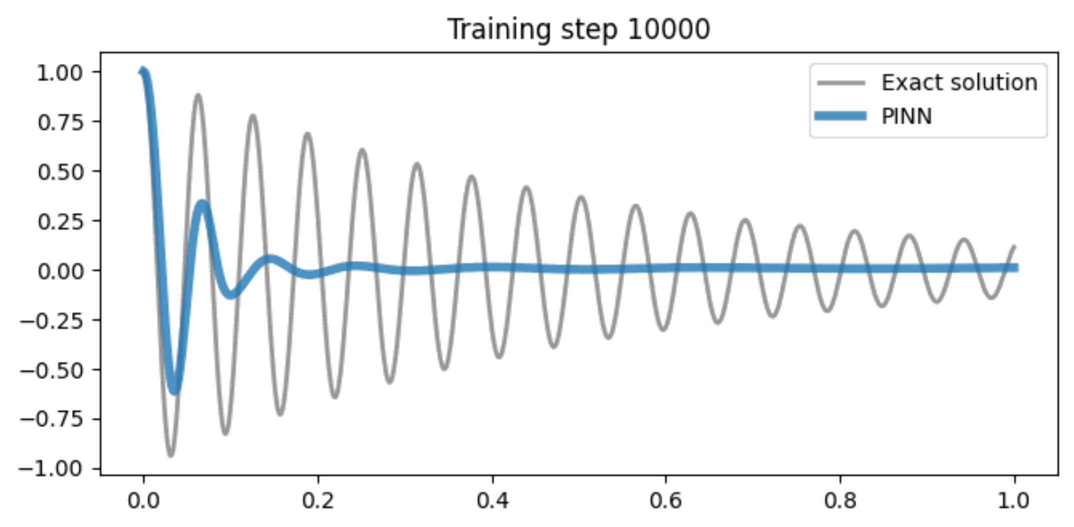
- By construction, FBPINN solution is continuous across subdomain interfaces
- Can be trained with same loss function as PINNs

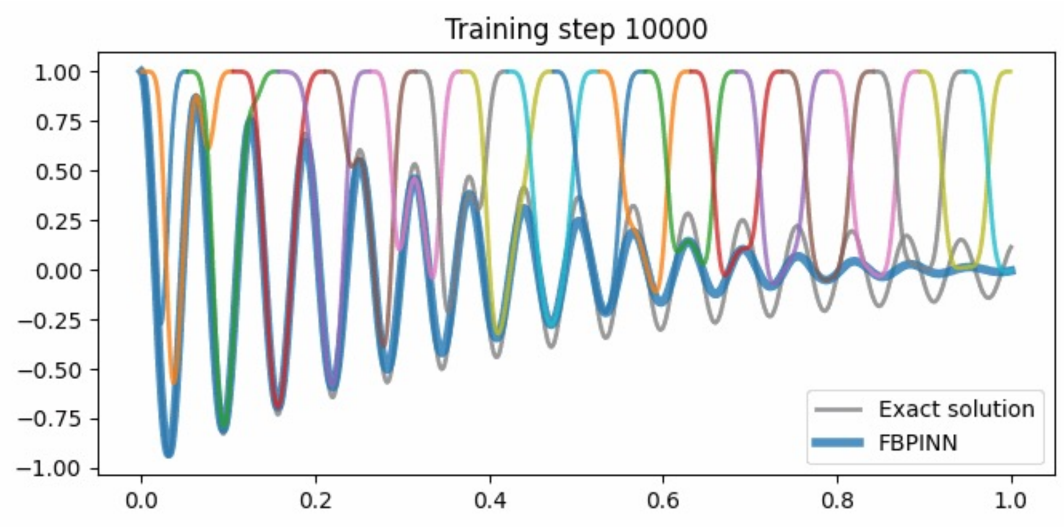
Note:

- Communication is carried out when summing subdomains in their overlap regions
- FBPINNs can simply be thought of as a custom NN architecture for PINNs

Moseley et al, Finite Basis Physics-Informed Neural Networks (FBPINNs): a scalable domain decomposition approach for solving differential equations, ArXiv (2021), ACM (2023)

# FBPINNs vs PINNs

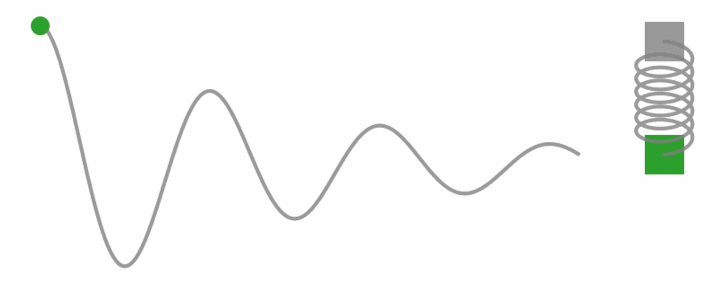



Problem: physics-informed neural networks **struggle** to model high-frequency / multiscale problems

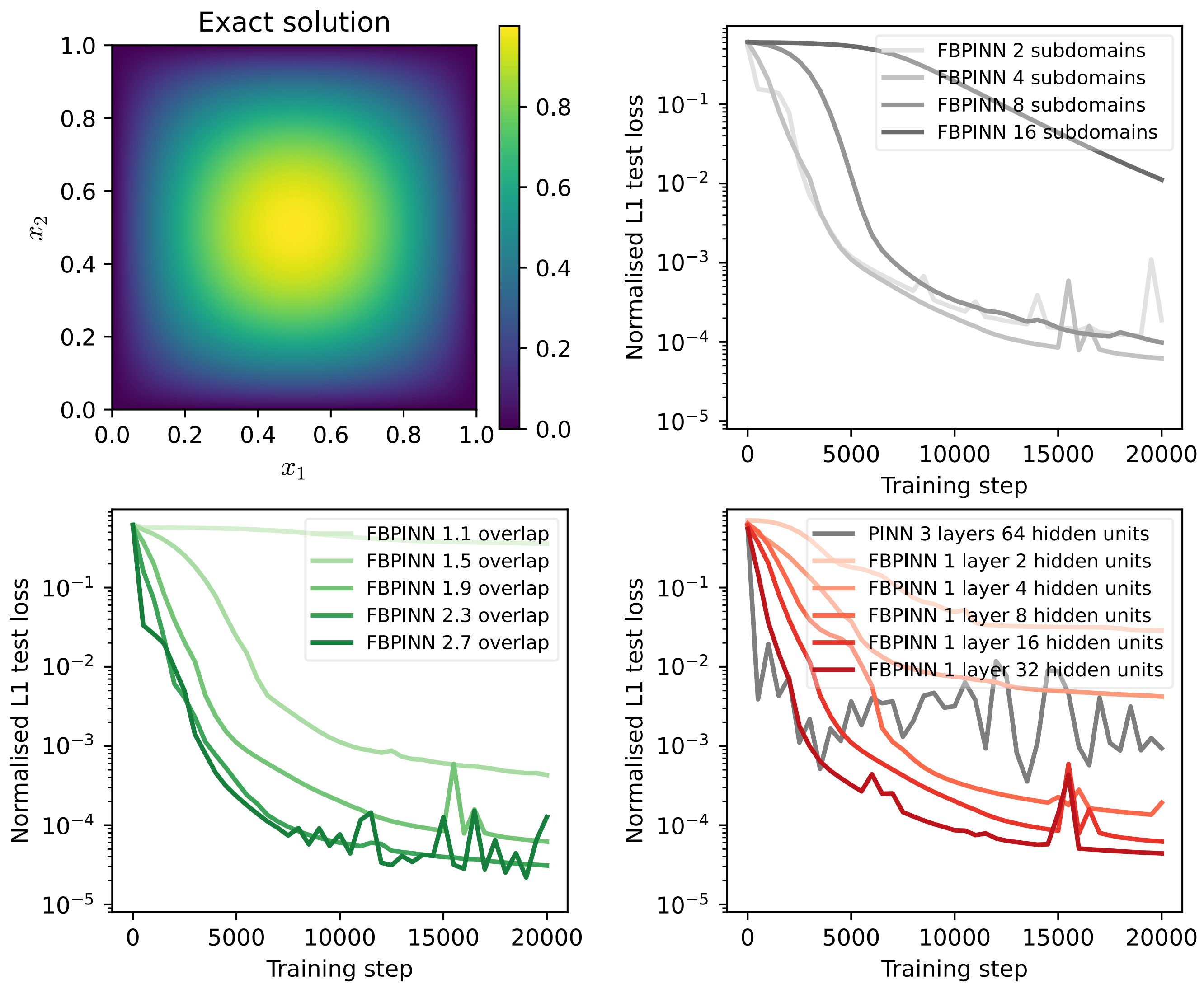
FBPINN solution

Number of subdomains: 20
Subdomain network size: 1 hidden layer, 16 hidden units



Damped harmonic oscillator

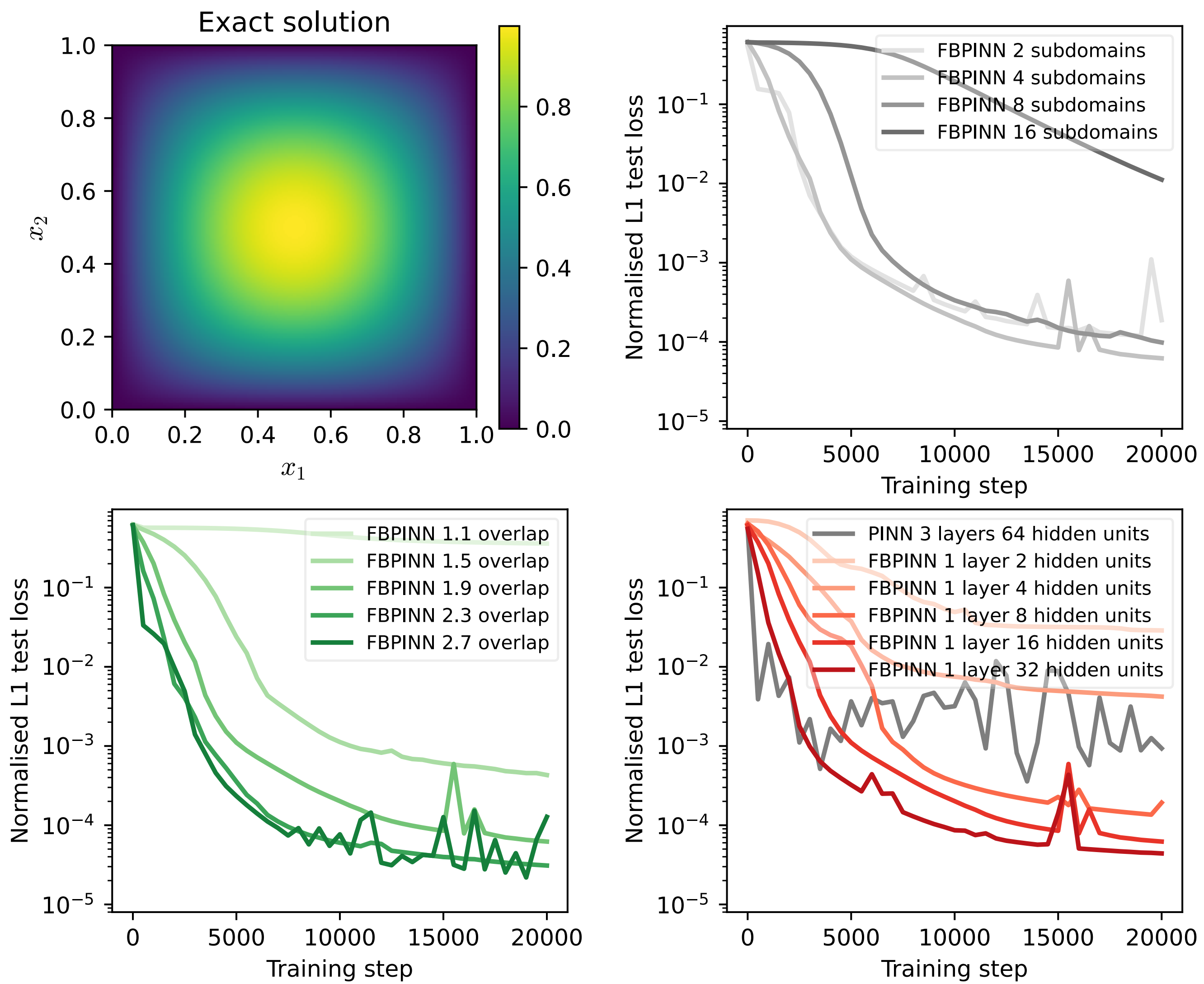
# FBPINN hyperparameter sensitivities




Laplacian problem:

$$-\Delta u = f \ \text{in} \ \Omega = [0,1]^2$$
$$u = 0 \ \text{on} \ \partial\Omega$$
$$f = 32\big(x_1(1-x_1) + x_2(1-x_2)\big)$$

# FBPINN hyperparameter sensitivities




Accuracy increases with:

- Size of overlap between subdomains
- Size (expressivity) of subdomain networks

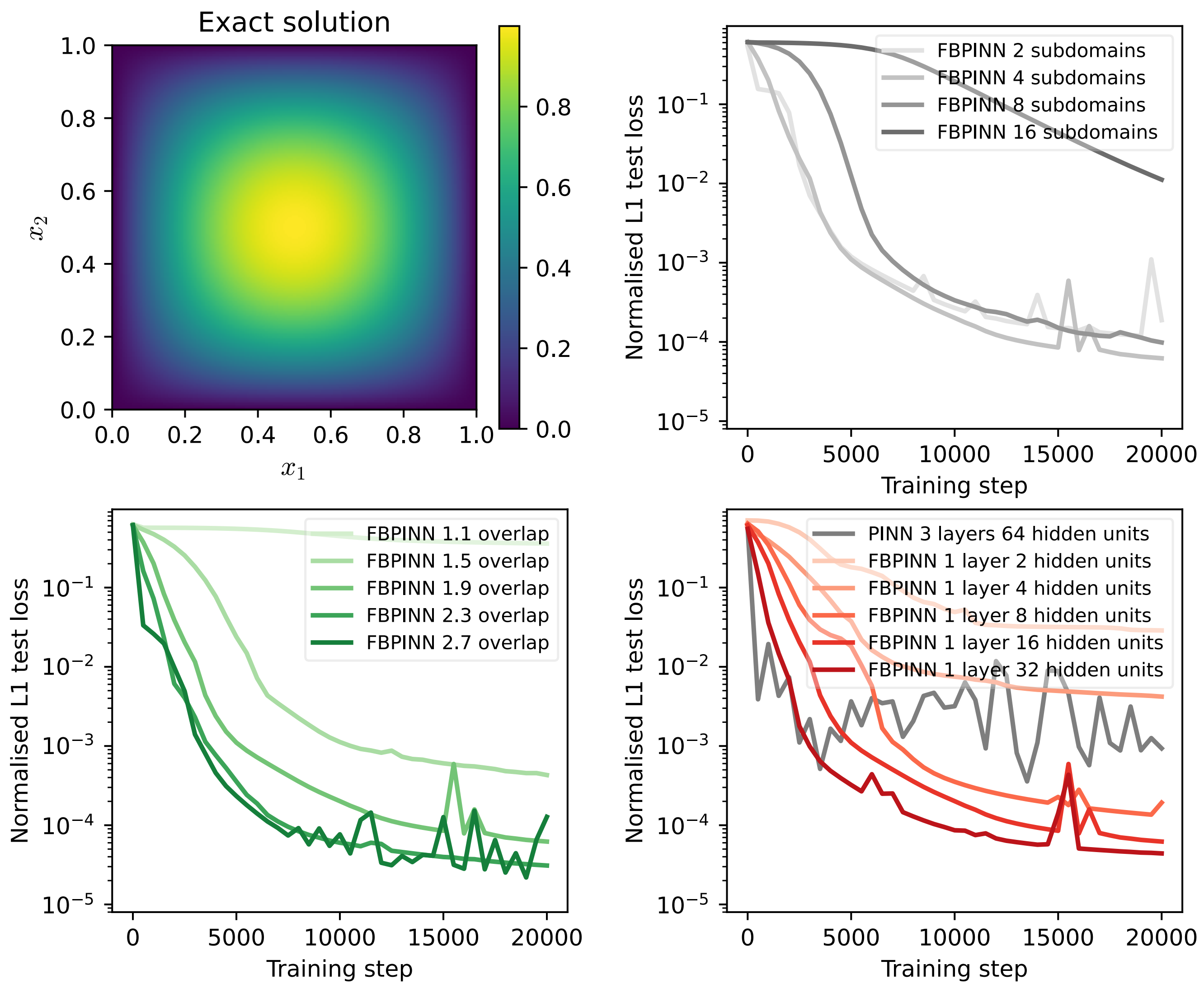
But reduces with:

- Number of subdomains (!)

Laplacian problem:

$$-\Delta u = f \ \text{ in } \ \Omega = [0,1]^2$$
$$u = 0 \ \text{ on } \ \partial\Omega$$
$$f = 32\big(x_1(1-x_1) + x_2(1-x_2)\big)$$

# FBPINN hyperparameter sensitivities




Accuracy increases with:

- Size of overlap between subdomains
- Size (expressivity) of subdomain networks

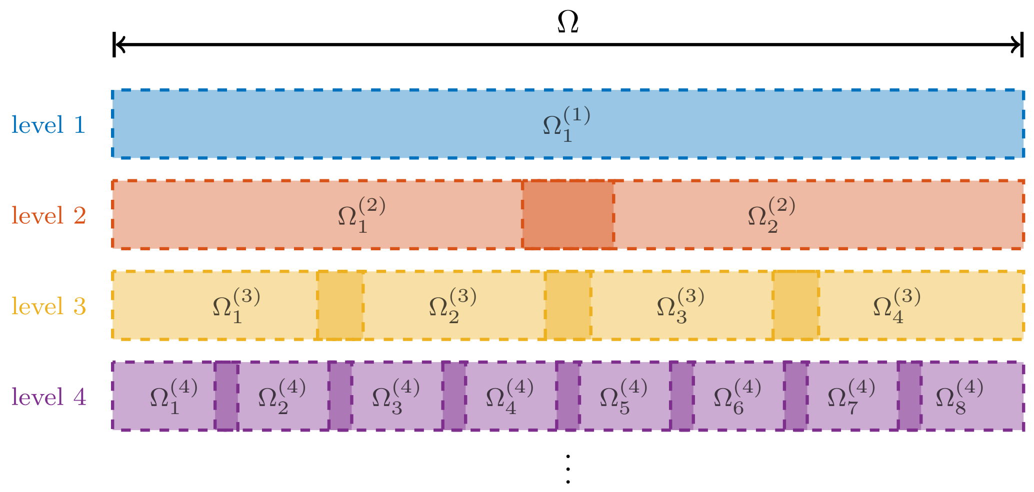
But reduces with:

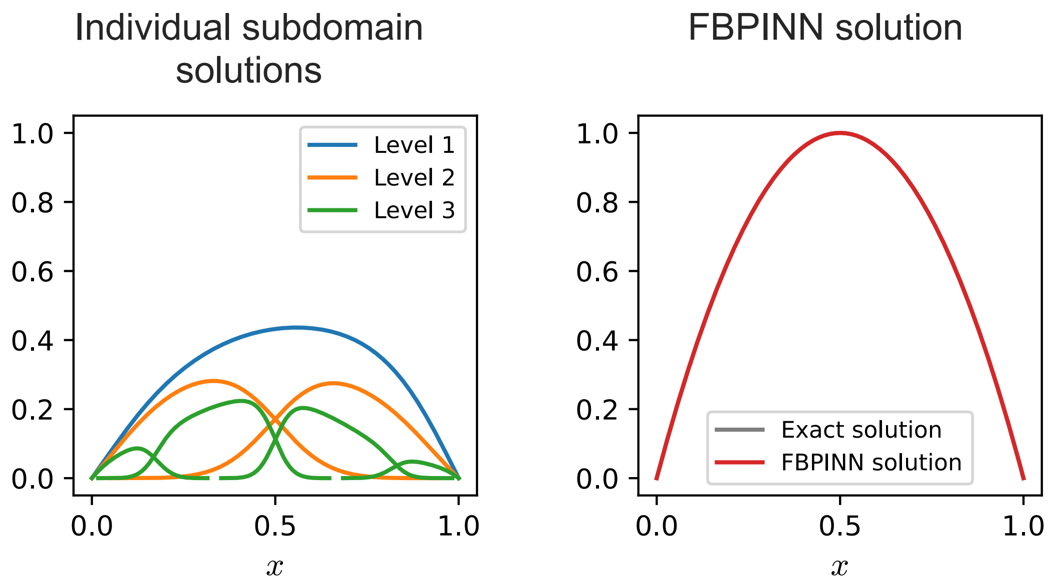
- Number of subdomains (!)

This is analogous to single-level classical **Schwarz** domain decomposition methods

Because **communication** is limited to overlapping regions

# Multilevel FBPINNs




Idea:

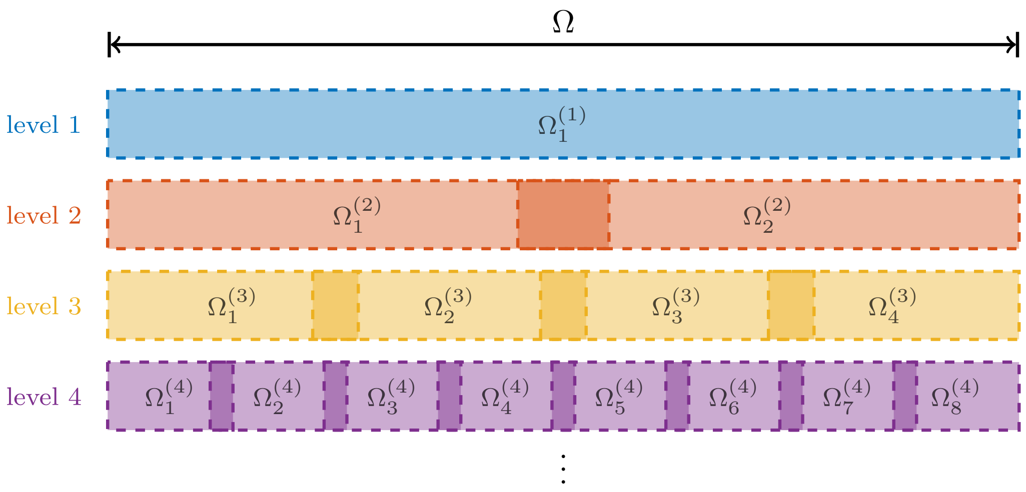
Use **multiple levels** of domain decompositions

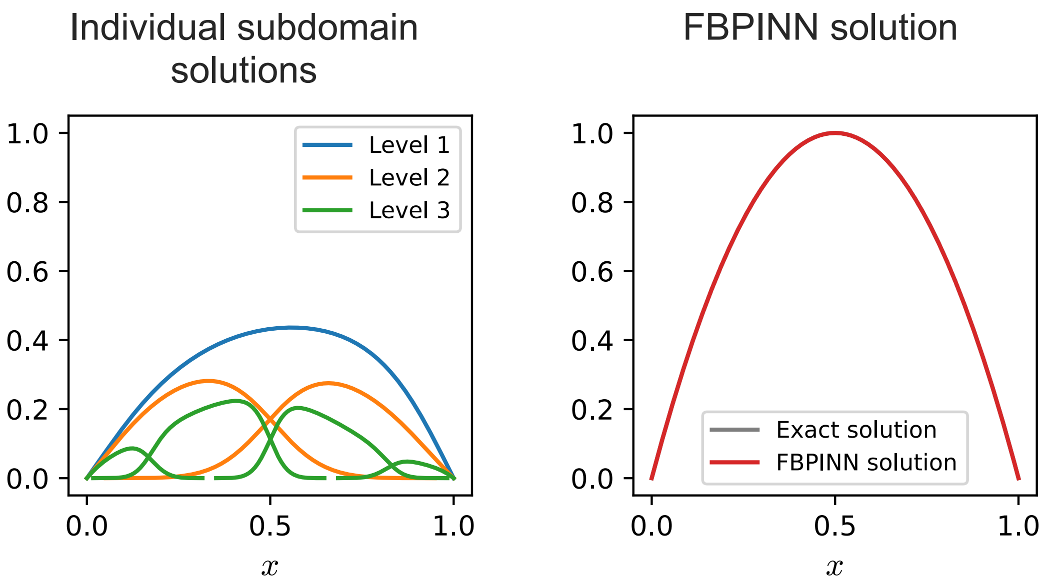
Hypothesis:

Improves global **communication** and helps model **multi-scale** solutions

Dolean, V., et al, Multilevel domain decomposition-based architectures for physics-informed neural networks, ArXiv (2023)

# Multilevel FBPINNs




Idea:

Use **multiple levels** of domain decompositions

Hypothesis:

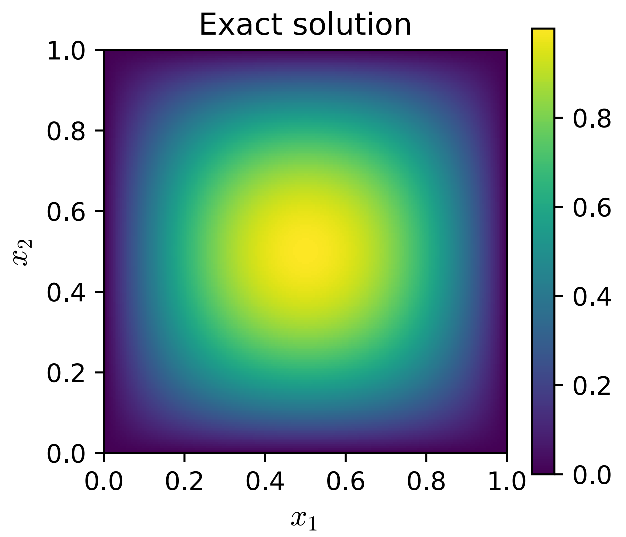
Improves global **communication** and helps model **multi-scale** solutions

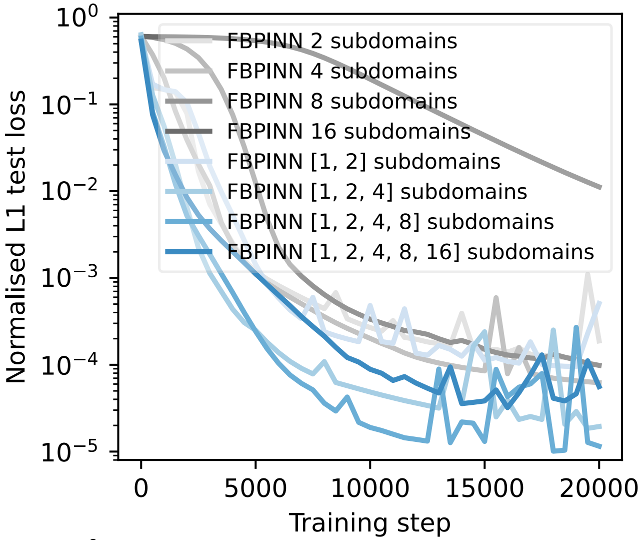
$$\hat{u}(x;\theta) = \mathcal{C}\left[\frac{1}{L}\sum_{l}^{L}\sum_{j}^{J^{(l)}} w_j^{(l)}(x)\cdot \mathrm{unnorm}\circ NN_j^{(l)}\circ \mathrm{norm}_j^{(l)}(x)\right]$$

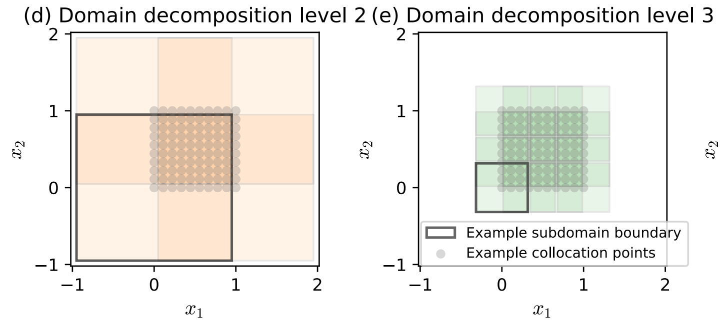
Represent solution as a summation over **all** levels

Dolean, V., et al, Multilevel domain decomposition-based architectures for physics-informed neural networks, ArXiv (2023)

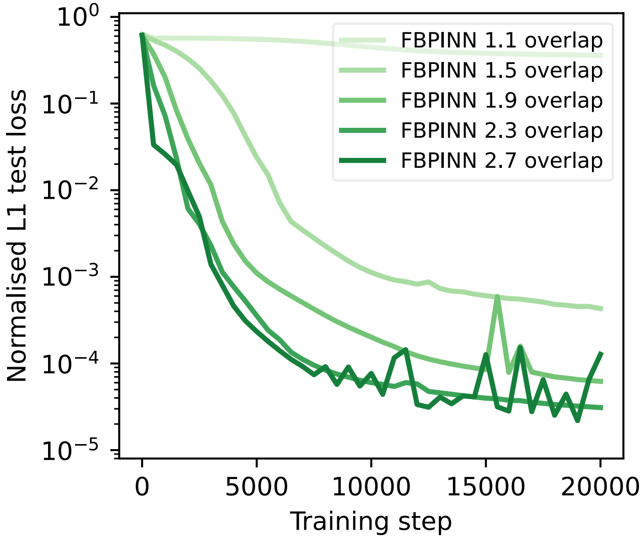
# Multilevel FBPINNs vs FBPINNs

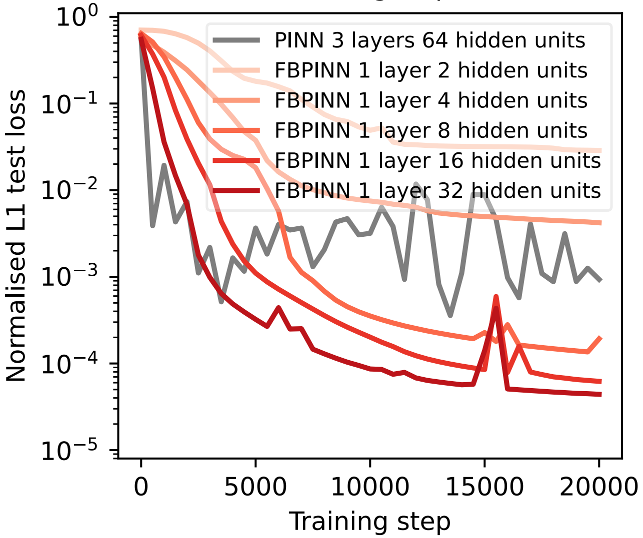



- Adding coarser levels significantly improves accuracy



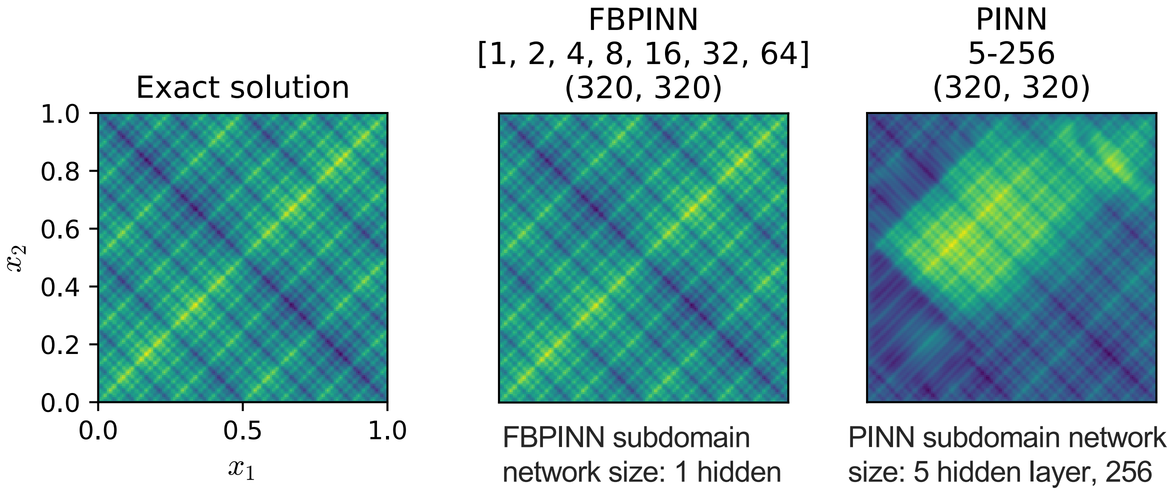

Laplacian problem:

$$-\Delta u = f \ \text{ in } \ \Omega = [0,1]^2$$
$$u = 0 \ \text{ on } \ \partial\Omega$$
$$f = 32\big(x_1(1-x_1) + x_2(1-x_2)\big)$$

# Parallel implementation of FBPINNs

FBPINNs are highly parallelizable:

- Each subdomain solution and its gradients can be computed in parallel
- Only points inside each subdomain are required to train each subdomain network
- Communication is only required when summing solutions in overlap regions
- FBPINNs typically require much smaller subdomain networks than PINNs




FBPINN subdomain network size: 1 hidden layer, 16 hidden units

PINN subdomain network size: 5 hidden layer, 256 hidden units

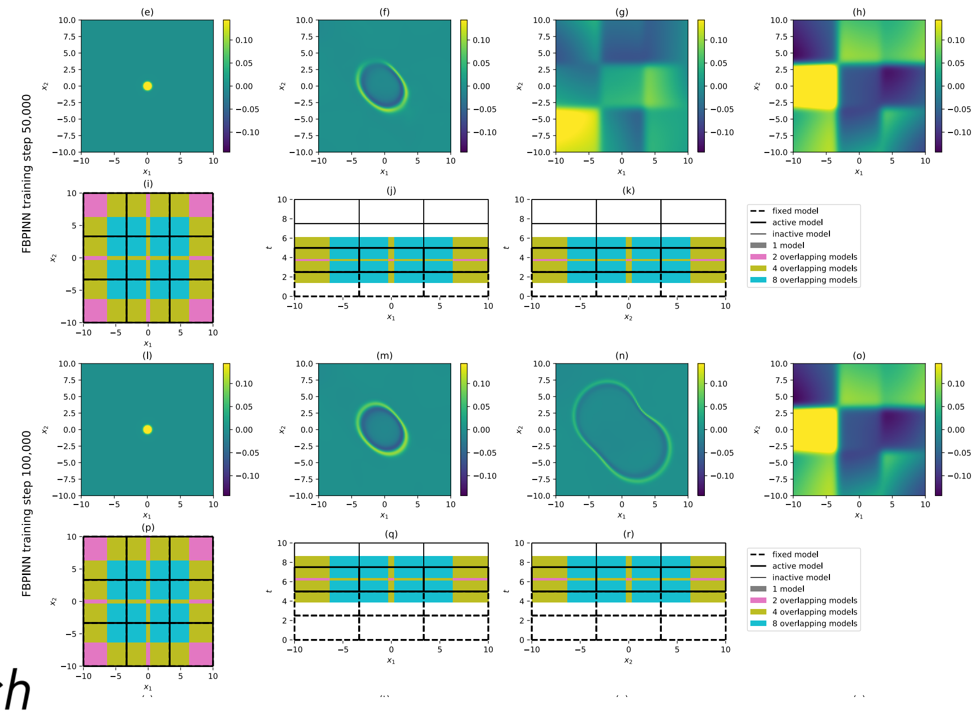
FBPINN training time: **10 mins**
PINN training time: **2 hours**

Problem: 2D multiscale Laplacian
With 7 frequencies spanning $2^1$ to $2^7$

# Parallel implementation of FBPINNs

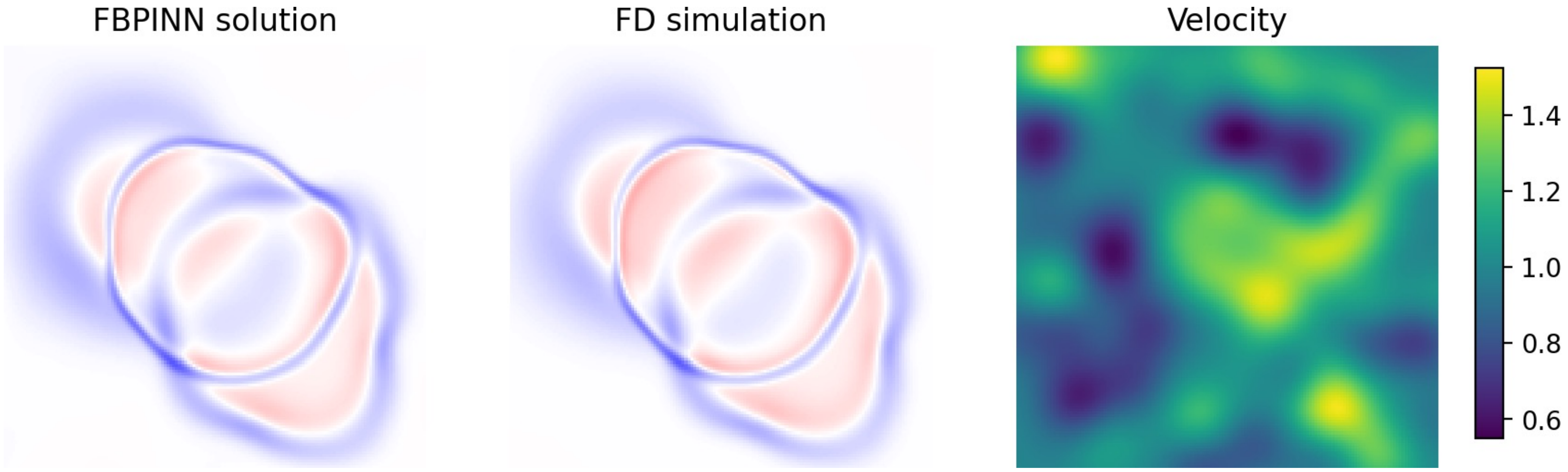
FBPINNs are highly parallelizable:

- Each subdomain solution and its gradients can be computed in parallel
- Only points inside each subdomain are required to train each subdomain network
- Communication is only required when summing solutions in overlap regions
- FBPINNs typically require much smaller subdomain networks than PINNs




- Subdomain scheduling can be used to fix parameters of certain subdomain networks during training
- E.g. time-stepping scheduling for causal problems

# High frequency / multiscale simulation with PINNs




Solving the 2+1D acoustic wave equation:

$$\nabla^2 u(x,t) - \frac{1}{c(x)^2}\frac{\partial^2 u(x,t)}{\partial t^2} = 0$$

Number of subdomains: 30 x 30 x 30 = 27,000
Subdomain network size: 1 hidden layer, 8 hidden units
Total number of free parameters: 1.1 M
Number of collocation points: 150 x 150 x 150 = 3.4 M
Optimiser: Time-stepping scheduling, Adam 0.001 lr
Training time: ~2 hr

# Limitations / future work

Limitations:

- Training time of FBPINNs is still typically slower than FD/FEM simulation for many problems
- Performance for high-dimensional PDEs not studied yet
- Communication of complex boundary conditions can still be challenging

# Limitations / future work

Limitations:

- Training time of FBPINNs is still typically slower than FD/FEM simulation for many problems
- Performance for high-dimensional PDEs not studied yet
- Communication of complex boundary conditions can still be challenging

Ongoing work:

- Learnable / adaptive domain decompositions
- More advanced scheduling / multilevel designs
- Updated open-source FBPINN library
    - V1.0 code available here: github.com/**benmoseley/FBPINNs**
    - V2.0 JAX code coming soon