

Multi-level domain decomposition-based physics-informed neural networks

Alexander Heinlein¹

28th International Conference on Domain Decomposition Methods (DD28), KAUST, Saudi Arabia,
January 28 - February 1, 2024

¹Delft University of Technology

Based on joint work with Damien Beecroft (University of Washington), Victorita Dolean (TU Eindhoven), Amanda A. Howard and Panos Stinis (Pacific Northwest National Laboratory), and Sid Mishra and Ben Moseley (ETH Zürich)

Artificial Neural Networks for Solving Ordinary and Partial Differential Equations

Isaac Elias Lagaris, Aristidis Likas, *Member, IEEE*, and Dimitrios I. Fotiadis

Published in **IEEE Transactions on Neural Networks, Vol. 9, No. 5, 1998.**

Approach

Solve a general differential equation subject to boundary conditions

$$G(x, \Psi(x), \nabla\Psi(x), \nabla^2\Psi(x)) = 0 \quad \text{in } \Omega$$

by solving an **optimization problem**

$$\min_{\theta} \sum_{x_i} G(x_i, \Psi_t(x_i, \theta), \nabla\Psi_t(x_i, \theta), \nabla^2\Psi_t(x_i, \theta))^2$$

where $\Psi_t(x, \theta)$ is a **trial function**, x_i sampling points inside the domain Ω and θ are **adjustable parameters**.

Construction of the trial functions

The trial functions **explicitly satisfy the boundary conditions**:

$$\Psi_t(x, \theta) = A(x) + F(x, N(x, \theta))$$

- N is a **feedforward neural network** with **trainable parameters** θ and input $x \in \mathbb{R}^n$
- A and F are **fixed functions**, chosen s.t.:
 - A satisfies the boundary conditions
 - F does not contribute to the boundary conditions

Neural Networks for Solving Differential Equations

Approach

Solve a general differential equation subject to boundary conditions

$$G(x, \Psi(x), \nabla\Psi(x), \nabla^2\Psi(x)) = 0 \quad \text{in } \Omega$$

by solving an **optimization problem**

$$\min_{\theta} \sum_{x_i} G(x_i, \Psi_t(x_i, \theta), \nabla\Psi_t(x_i, \theta), \nabla^2\Psi_t(x_i, \theta))^2$$

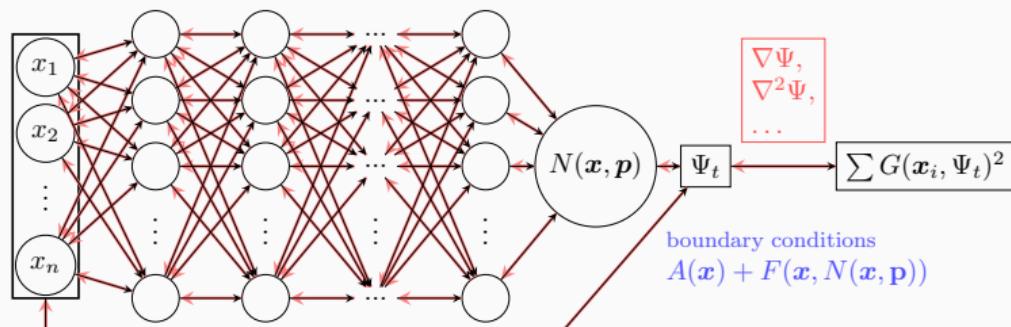
where $\Psi_t(x, \theta)$ is a **trial function**, x_i sampling points inside the domain Ω and θ are adjustable parameters.

Construction of the trial functions

The trial functions **explicitly satisfy the boundary conditions**:

$$\Psi_t(x, \theta) = A(x) + F(x, N(x, \theta))$$

- N is a **feedforward neural network** with **trainable parameters** θ and input $x \in \mathbb{R}^n$
- A and F are **fixed functions**, chosen s.t.:
 - A satisfies the **boundary conditions**
 - F does not contribute to the **boundary conditions**



Physics-Informed Neural Networks (PINNs)

In the **physics-informed neural network (PINN)** approach introduced by [Raissi et al. \(2019\)](#), a neural network is employed to **discretize a partial differential equation**

$$\mathcal{N}[u](\mathbf{x}, t) = f(\mathbf{x}, t), \quad (\mathbf{x}, t) \in [0, T] \times \Omega \subset \mathbb{R}^d.$$

It is based on the approach by [Lagaris et al. \(1998\)](#). The main novelty of PINNs is the use of a **hybrid loss function**:

$$\mathcal{L} = \omega_{\text{data}} \mathcal{L}_{\text{data}} + \omega_{\text{PDE}} \mathcal{L}_{\text{PDE}},$$

where ω_{data} and ω_{PDE} are **weights** and

$$\mathcal{L}_{\text{data}} = \frac{1}{N_{\text{data}}} \sum_{i=1}^{N_{\text{data}}} (u(\hat{\mathbf{x}}_i, \hat{t}_i) - u_i)^2,$$

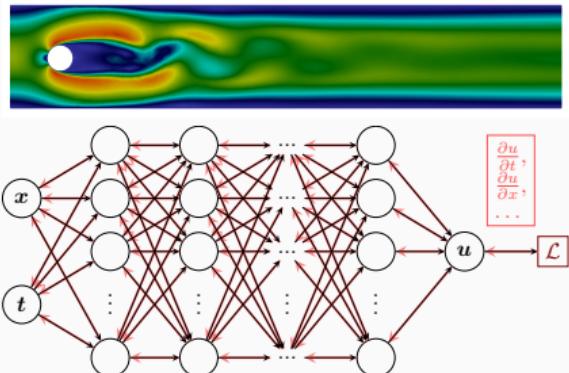
$$\mathcal{L}_{\text{PDE}} = \frac{1}{N_{\text{PDE}}} \sum_{i=1}^{N_{\text{PDE}}} (\mathcal{N}[u](\mathbf{x}_i, t_i) - f(\mathbf{x}_i, t_i))^2.$$

Advantages

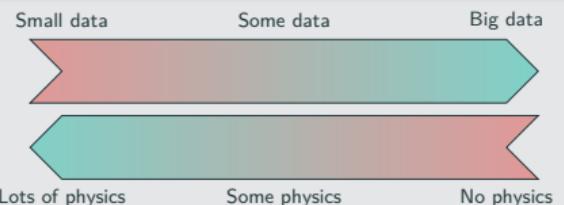
- "Meshfree"
- Small data
- Generalization properties
- High-dimensional problems
- Inverse and parameterized problems

Drawbacks

- Training cost and robustness
- Convergence not well-understood
- Difficulties with scalability and multi-scale problems



Hybrid loss



- Known solution values can be included in $\mathcal{L}_{\text{data}}$
- Initial and boundary conditions are also included in $\mathcal{L}_{\text{data}}$

Mishra and Molinaro. *Estimates on the generalisation error of PINNs, 2022*

Estimate of the generalization error

The generalization error (or total error) satisfies

$$\mathcal{E}_G \leq C_{\text{PDE}} \mathcal{E}_{\mathcal{T}} + C_{\text{PDE}} C_{\text{quad}}^{1/p} N^{-\alpha/p}$$

where

- $\mathcal{E}_G = \mathcal{E}_G(\theta; \mathbf{X}) := \|\mathbf{u} - \mathbf{u}^*\|_V$ (V Sobolev space, \mathbf{X} training data set)
- $\mathcal{E}_{\mathcal{T}}$ is the training error ($/p$ loss of the residual of the PDE)
- C_{PDE} and C_{quad} constants depending on the PDE resp. the quadrature
- N number of the training points and α convergence rate of the quadrature

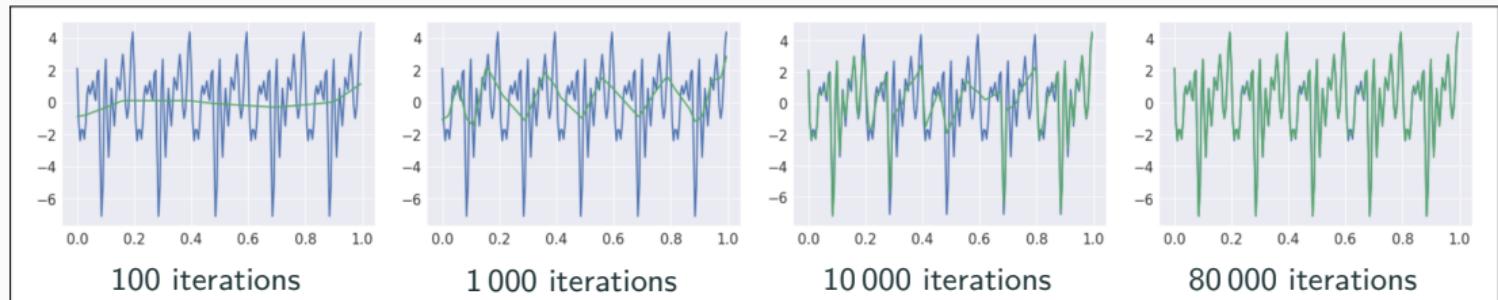
Rule of thumb:

“As long as the PINN is trained well, it also generalizes well”

Scaling Issues in Neural Network Training

Spectral bias

Neural networks prioritize learning lower frequency functions first irrespective of their amplitude.



Rahaman et al., *On the spectral bias of neural networks*, ICML (2019)

- Solving solutions on **large domains and/or with multiscale features** potentially requires **very large neural networks**.
- Training may **not sufficiently reduce the loss** or take **large numbers of iterations**.
- Significant **increase on the computational work**

Convergence analysis of PINNs via the **neural tangent kernel**: Wang, Yu, Perdikaris,
When and why PINNs fail to train: A neural tangent kernel perspective, JCP (2022)

Motivation – Some Observations on the Performance of PINNs

Solve

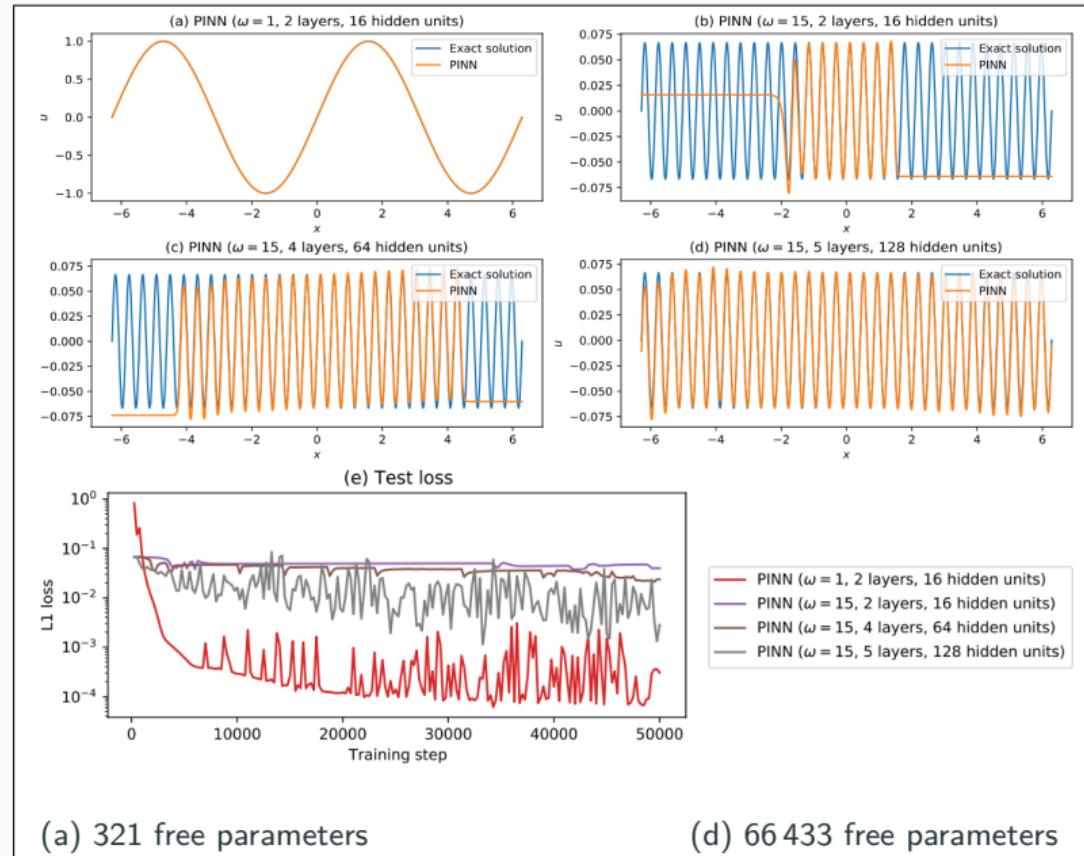
$$\begin{aligned} u' &= \cos(\omega x), \\ u(0) &= 0, \end{aligned}$$

for different values of ω
using PINNs with
varying network
capacities.

Scaling issues

- Large computational domains
- Small frequencies

Cf. Moseley, Markham, and
Nissen-Meyer (2023)



Motivation – Some Observations on the Performance of PINNs

Solve

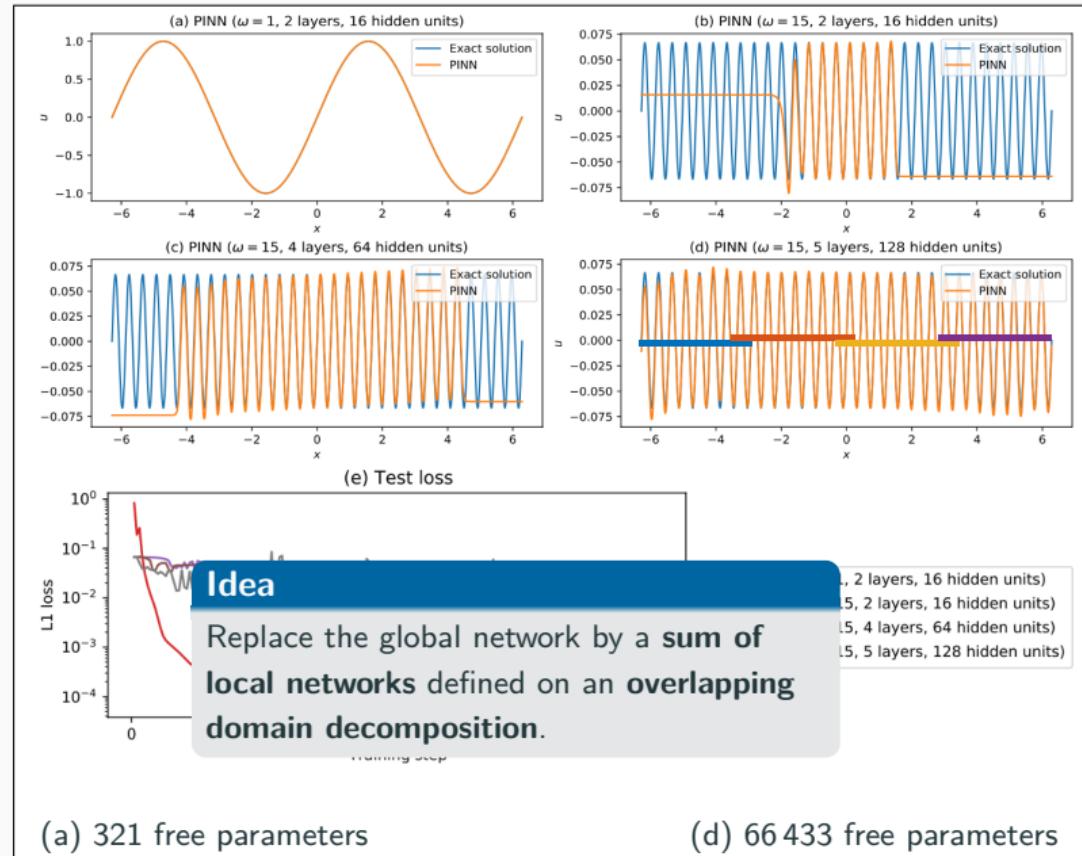
$$\begin{aligned} u' &= \cos(\omega x), \\ u(0) &= 0, \end{aligned}$$

for different values of ω
using PINNs with
varying network
capacities.

Scaling issues

- Large computational domains
- Small frequencies

Cf. Moseley, Markham, and Nissen-Meyer (2023)

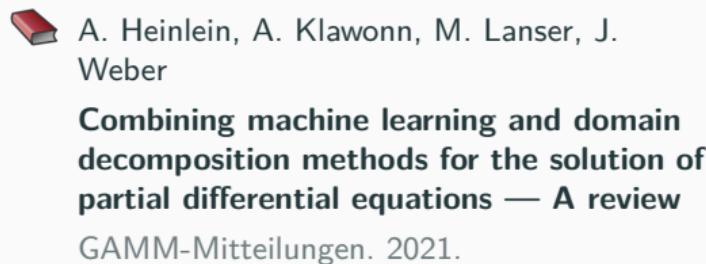


Machine Learning and Domain Decomposition Methods

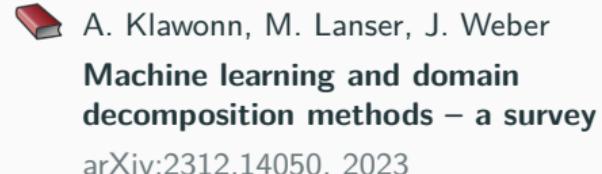
A non-exhaustive overview:

- Machine Learning for adaptive BDDC, FETI-DP, and AGDSW: Heinlein, Klawonn, Lanser, Weber (2019, 2020, 2021, 2021, 2021, 2022); Klawonn, Lanser, Weber (preprint 2022)
- Domain decomposition for CNNs: Gu, Zhang, Liu, Cai (2022); Lee, Park, Lee (2022); Klawonn, Lanser, Weber (arXiv 2023)
- D3M: Li, Tang, Wu, and Liao (2019)
- DeepDDM: Li, Xiang, Xu (2020); Mercier, Gratton, Boudier (arXiv 2021); Li, Wang, Cui, Xiang, Xu (2023); Sun, Xu, Yi (arXiv 2022, arXiv 2023)
- FBPINNs: Moseley, Markham, and Nissen-Meyer (2023); Dolean, Heinlein, Mishra, Moseley (2024, subm. 2023 / arXiv:2306.05486); Heinlein, Howard, Beecroft, Stinis (subm. 2024 / arXiv:2401.07888)
- Schwarz Domain Decomposition Algorithm for PINNs: Kim, Yang (2022, arXiv 2022)
- cPINNs: Jagtap, Kharazmi, Karniadakis (2020)
- XPINNs: Jagtap, Karniadakis (2020)

An overview of the state-of-the-art in early 2021:



An overview of the state-of-the-art in the end of 2023:



Finite Basis Physics-Informed Neural Networks (FBPINNs)

In the **finite basis physics informed neural network (FBPINNs) method** introduced in [Moseley, Markham, and Nissen-Meyer \(2023\)](#), we solve the boundary value problem

$$\begin{aligned} \mathcal{N}[u](\mathbf{x}) &= f(\mathbf{x}), \quad \mathbf{x} \in \Omega \subset \mathbb{R}^d, \\ \mathcal{B}_k[u](\mathbf{x}) &= g_k(\mathbf{x}), \quad \mathbf{x} \in \Gamma_k \subset \partial\Omega. \end{aligned}$$

using the **PINN approach** and **hard enforcement of the boundary conditions**, similar to [Lagaris et al. \(1998\)](#).

FBPINNs use the **network architecture**

$$u(\theta_1, \dots, \theta_J) = \mathcal{C} \sum_{j=1}^J \omega_j u_j(\theta_j)$$

and the **loss function**

$$\mathcal{L}(\theta_1, \dots, \theta_J) = \frac{1}{N} \sum_{i=1}^N \left(n[\mathcal{C} \sum_{x_i \in \Omega_j} \omega_j u_j](\mathbf{x}_i, \theta_j) - f(\mathbf{x}_i) \right)^2.$$

- **Overlapping DD:** $\Omega = \bigcup_{j=1}^J \Omega_j$
- **Window functions** ω_j with $\text{supp}(\omega_j) \subset \Omega_j$ and $\sum_{j=1}^J \omega_j \equiv 1$ on Ω

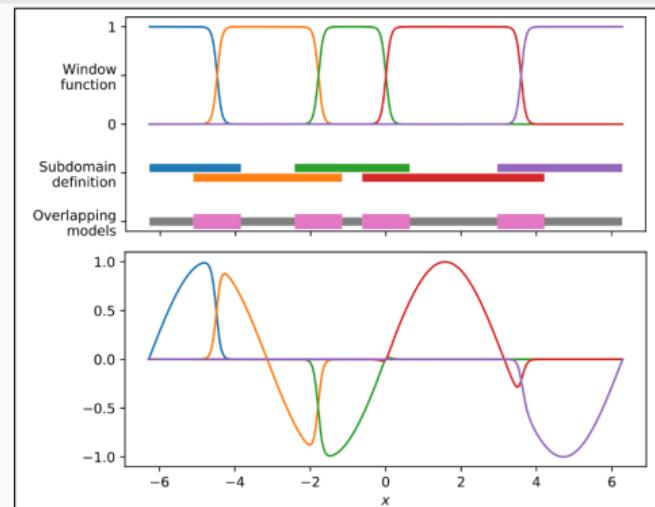
Hard enforcement of boundary conditions

Loss function

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^N (n[\mathcal{C} u](\mathbf{x}_i, \theta) - f(\mathbf{x}_i))^2,$$

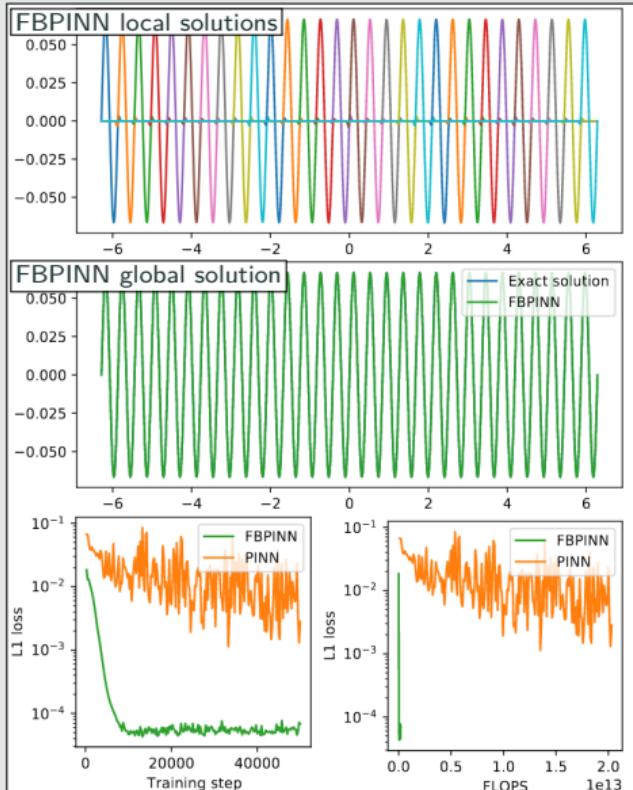
with constraining operator \mathcal{C} , which **explicitly enforces the boundary conditions**.

→ Often **improves training performance**



Numerical Results for FBPINNs

PINN vs FBPINN (Moseley et al. (2023))



Scalability of FBPINNs

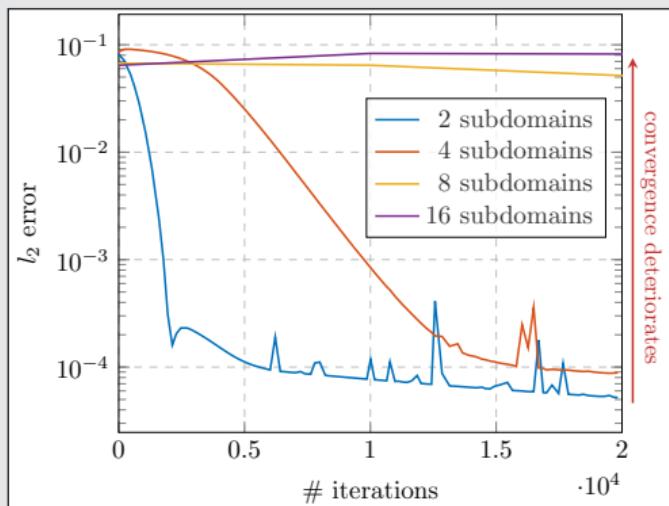
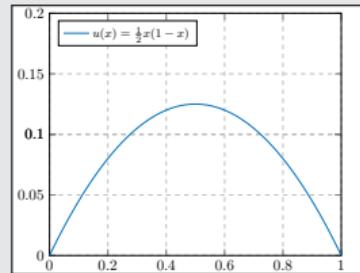
Consider the simple boundary value problem

$$-u'' = 1 \quad \text{in } [0, 1],$$

$$u(0) = u(1) = 0,$$

which has the solution

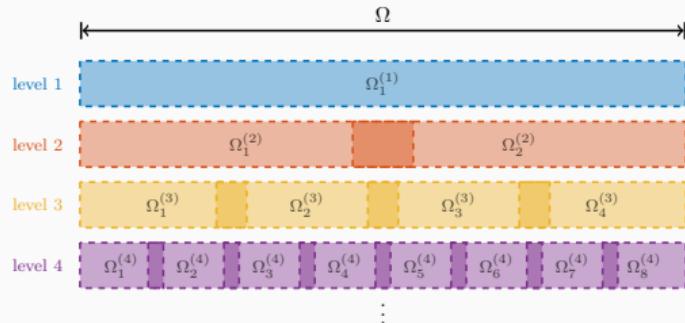
$$u(x) = \frac{1}{2}x(1 - x).$$



Multi-Level FBPINN Algorithm

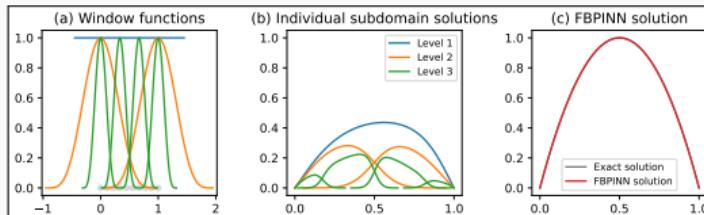
Extension of FBPINNs to L levels; Cf. Dolean, Heinlein,

Mishra, Moseley (submitted 2023 / arXiv:2306.05486).



L -level network architecture

$$u(\theta_1^{(1)}, \dots, \theta_{J(L)}^{(L)}) = \mathcal{C} \left(\sum_{l=1}^L \sum_{i=1}^{N^{(l)}} \omega_j^{(l)} u_j^{(l)}(\theta_j^{(l)}) \right)$$



Multi-Frequency Problem

Let us now consider the two-dimensional multi-frequency Laplace boundary value problem

$$\begin{aligned} -\Delta u &= 2 \sum_{i=1}^n (\omega_i \pi)^2 \sin(\omega_i \pi x) \sin(\omega_i \pi y) && \text{in } \Omega, \\ u &= 0 && \text{on } \partial\Omega, \end{aligned}$$

with $\omega_i = 2^i$.

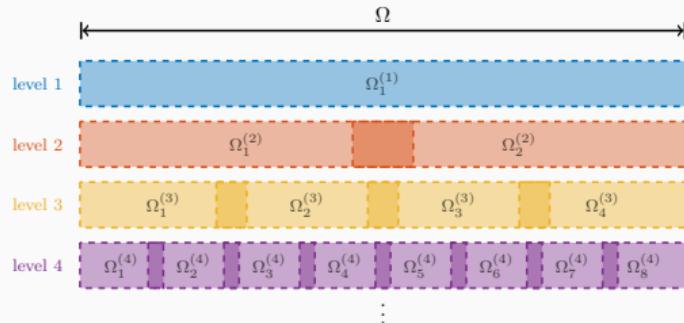
For increasing values of n , we obtain the analytical solutions:



Multi-Level FBPINN Algorithm

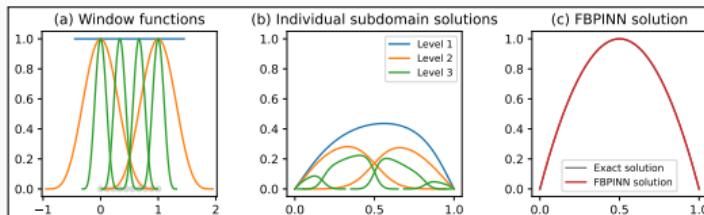
Extension of FBPINNs to L levels; Cf. Dolean, Heinlein,

Mishra, Moseley (submitted 2023 / arXiv:2306.05486).



L -level network architecture

$$u(\theta_1^{(1)}, \dots, \theta_{J(L)}^{(L)}) = \mathcal{C} \left(\sum_{l=1}^L \sum_{i=1}^{N^{(l)}} \omega_j^{(l)} u_j^{(l)}(\theta_j^{(l)}) \right)$$



Multi-Frequency Problem

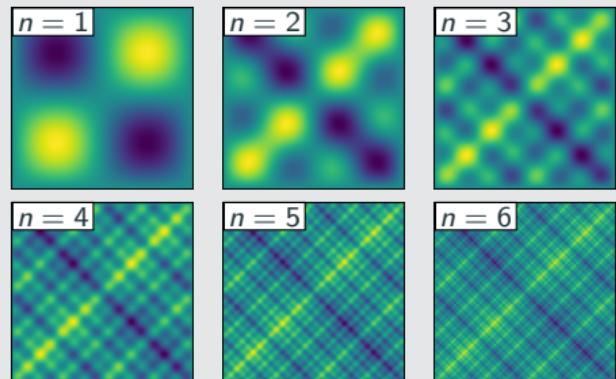
Let us now consider the **two-dimensional multi-frequency Laplace boundary value problem**

$$-\Delta u = 2 \sum_{i=1}^n (\omega_i \pi)^2 \sin(\omega_i \pi x) \sin(\omega_i \pi y) \quad \text{in } \Omega,$$

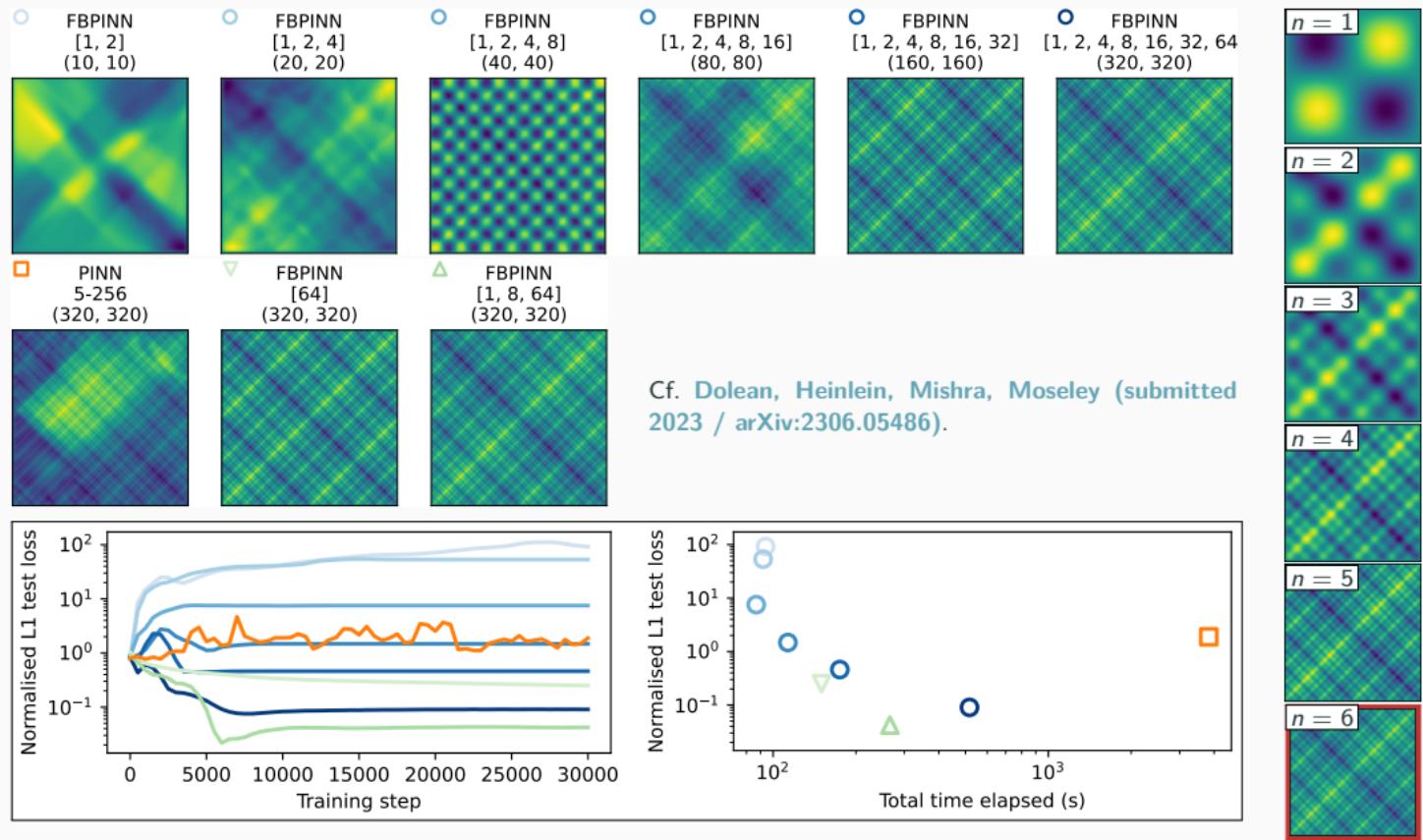
$$u = 0 \quad \text{on } \partial\Omega,$$

with $\omega_i = 2^i$.

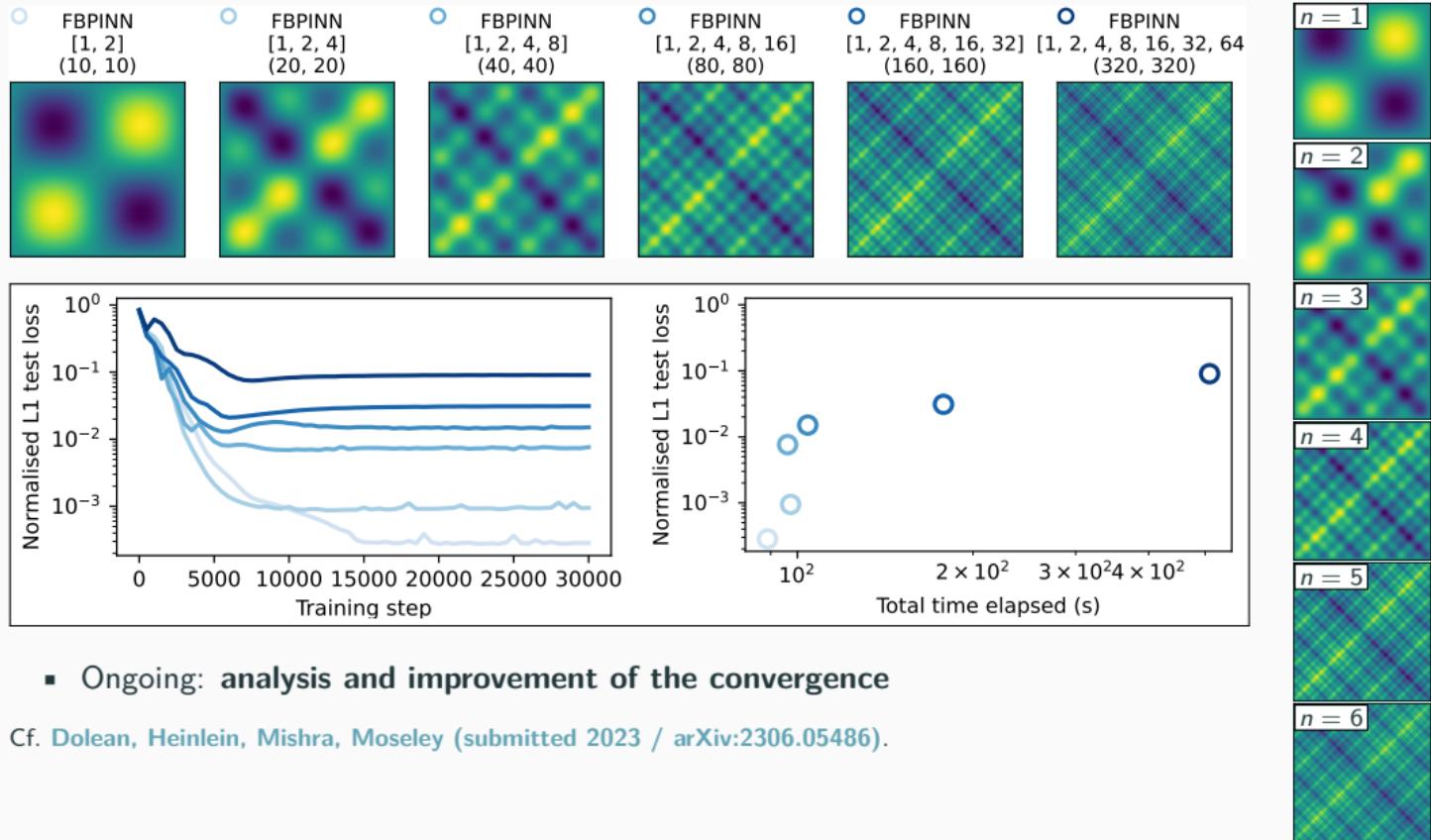
For increasing values of n , we obtain the **analytical solutions**:



Multi-Level FBPINNs for a Multi-Frequency Problem – Strong Scaling



Multi-Level FBPINNs for a Multi-Frequency Problem – Weak Scaling



- Ongoing: analysis and improvement of the convergence

Cf. [Dolean, Heinlein, Mishra, Moseley \(submitted 2023 / arXiv:2306.05486\)](#).

Helmholtz Problem

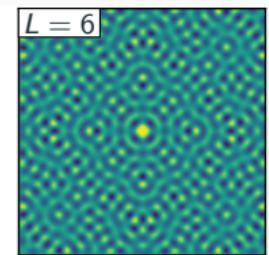
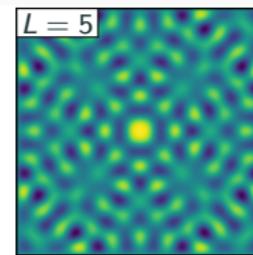
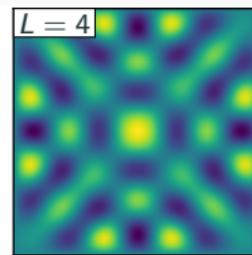
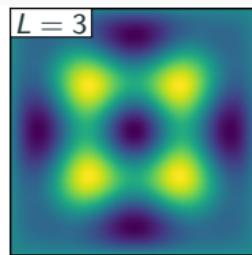
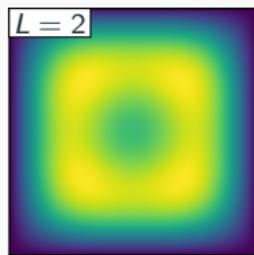
Finally, let us consider the **two-dimensional Helmholtz boundary value problem**

$$\Delta u - k^2 u = f \quad \text{in } \Omega = [0, 1]^2,$$

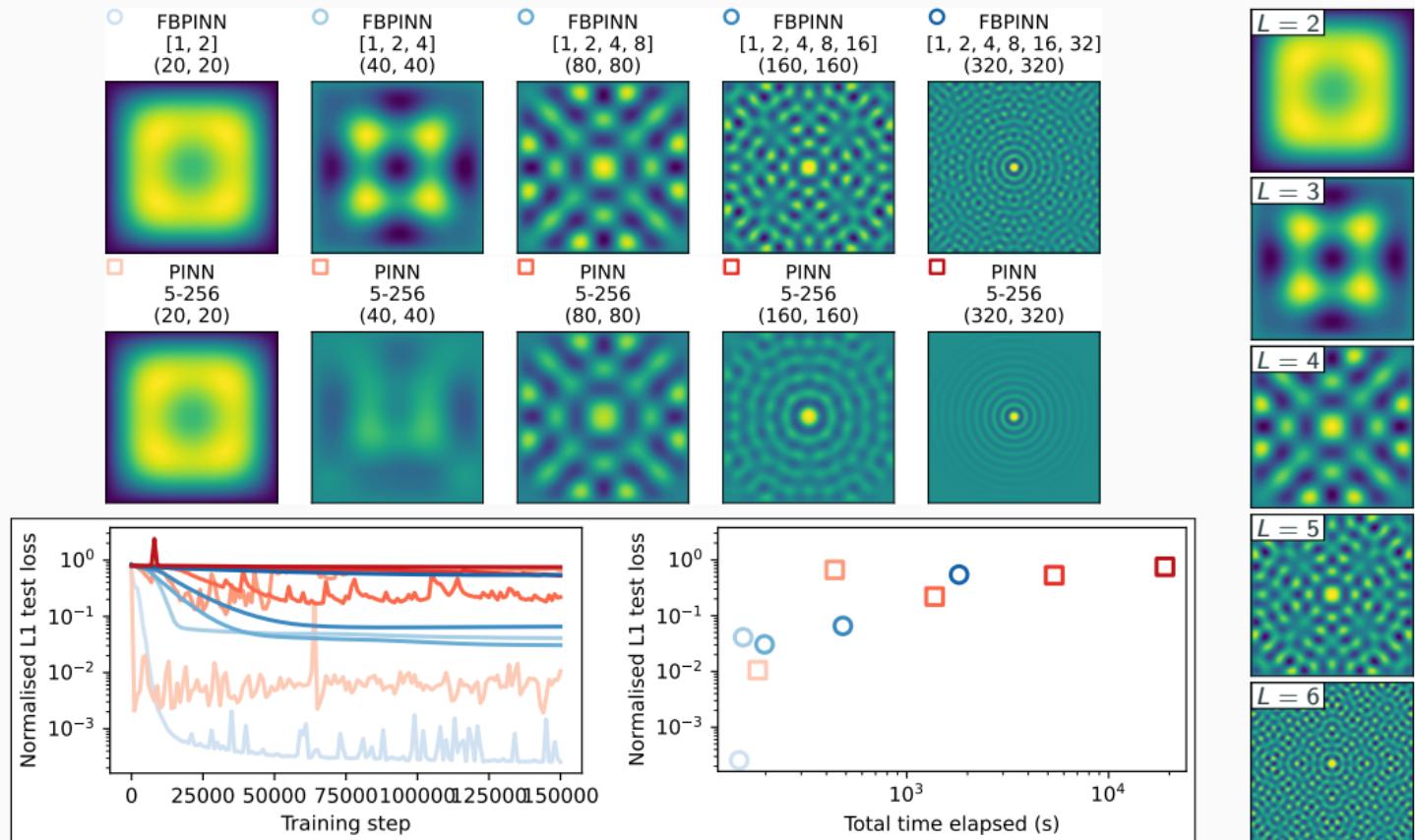
$$u = 0 \quad \text{on } \partial\Omega,$$

$$f(\mathbf{x}) = e^{-\frac{1}{2}(\|\mathbf{x}-0.5\|/\sigma)^2}.$$

With $k = 2^L \pi / 1.6$ and $\sigma = 0.8 / 2^L$, we obtain the **solutions**:



Multi-Level FBPINNs for the Helmholtz Problem – Weak Scaling

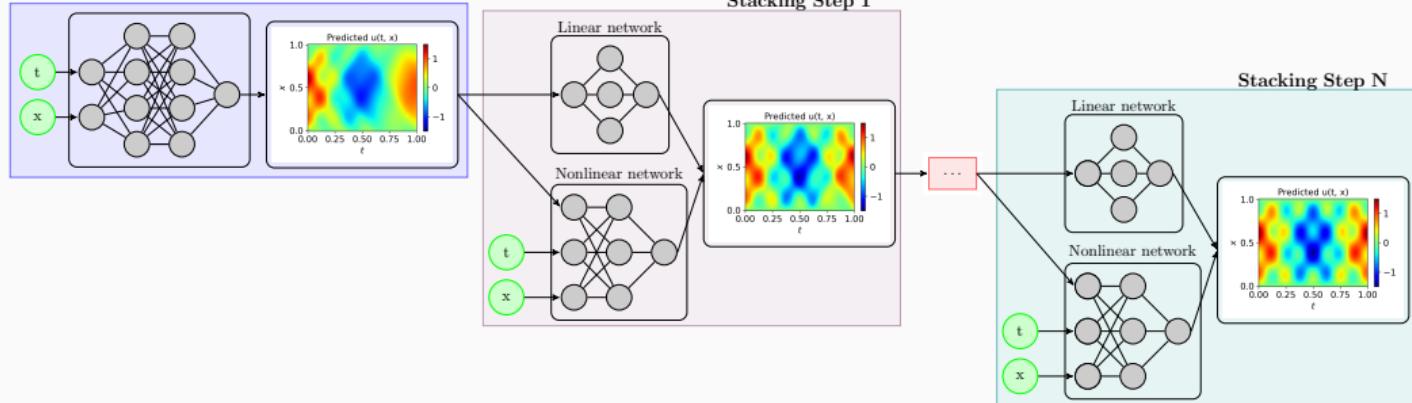


Stacking Multifidelity FBPINNs

In the **stacked PINNs** approach in [Howard, Murphy, Ahmed, Stinis \(arXiv 2023\)](#), **multiple networks are stacked on top of each other** in a recursive way. In particular, the next model \hat{u}^{MF} is trained as a corrector for the previous model \hat{u}^{SF} :

$$\hat{u}^{MF}(x, \theta^{MF}) = (1 - |\alpha|) \hat{u}_{linear}^{MF}(x, \hat{u}^{SF}, \theta^{MF}) + |\alpha| \hat{u}_{nonlinear}^{MF}(x, \hat{u}^{SF}, \theta^{MF})$$

Step 0



Stacking multifidelity FBPINNs

We **combine stacking multifidelity PINNs with FBPINNs** by using an FBPINN model (with an increasing number of subdomains) in each stacking step. → **Sequential/multiplicative coupling** of the levels

[Heinlein, Howard, Beecroft, Stinis \(subm. 2024 / arXiv:2401.07888\)](#)

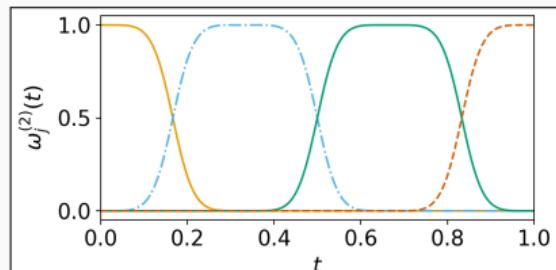
Numerical Results – Pendulum Problem

First, we consider a **pendulum problem** and compare the stacking multifidelity PINN and FBPINN approaches:

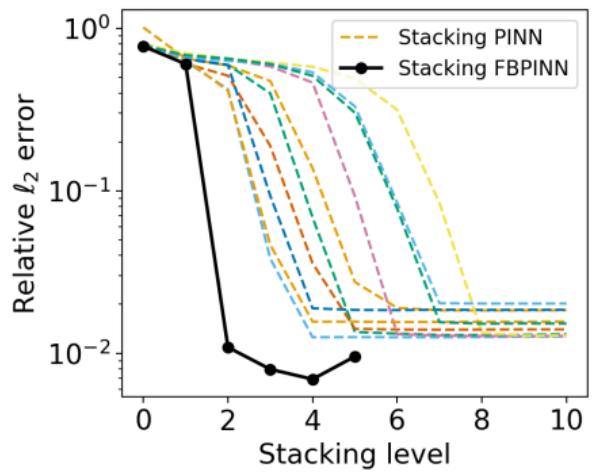
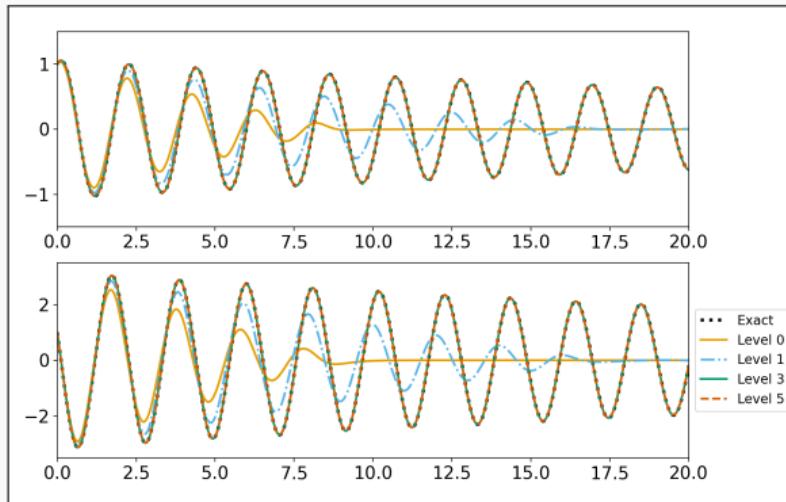
$$\frac{ds_1}{dt} = s_2,$$

$$\frac{ds_2}{dt} = -\frac{b}{m}s_2 - \frac{g}{L} \sin(s_1)$$

with $m = L = 1$, $b = 0.05$, $g = 9.81$, and $T = 20$.



Partition of unity in time

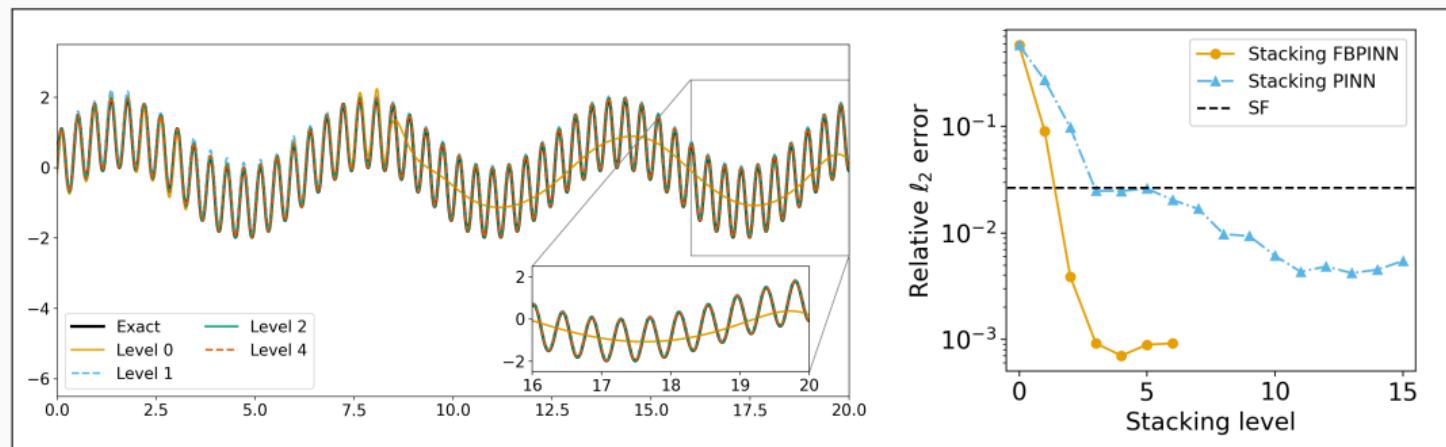


Numerical Results – Allen–Cahn Equation

Second, we consider a **multifrequency problem**:

$$\begin{aligned}\frac{ds}{dx} &= \omega_1 \cos(\omega_1 x) + \omega_2 \cos(\omega_2 x), \\ s(0) &= 0,\end{aligned}$$

on domain $\Omega = [0, 20]$ with $\omega_1 = 1$ and $\omega_2 = 15$.



→ Due to the **multiscale structure of the problem**, the **improvements** due to the **multifidelity FBPINN approach** are **even stronger**.

Numerical Results – Allen–Cahn Equation

Finally, we consider the **Allen–Cahn equation**:

$$s_t - 0.0001s_{xx} + 5s^3 - 5s = 0,$$

$$s(x, 0) = x^2 \cos(\pi x),$$

$$s(x, t) = s(-x, t),$$

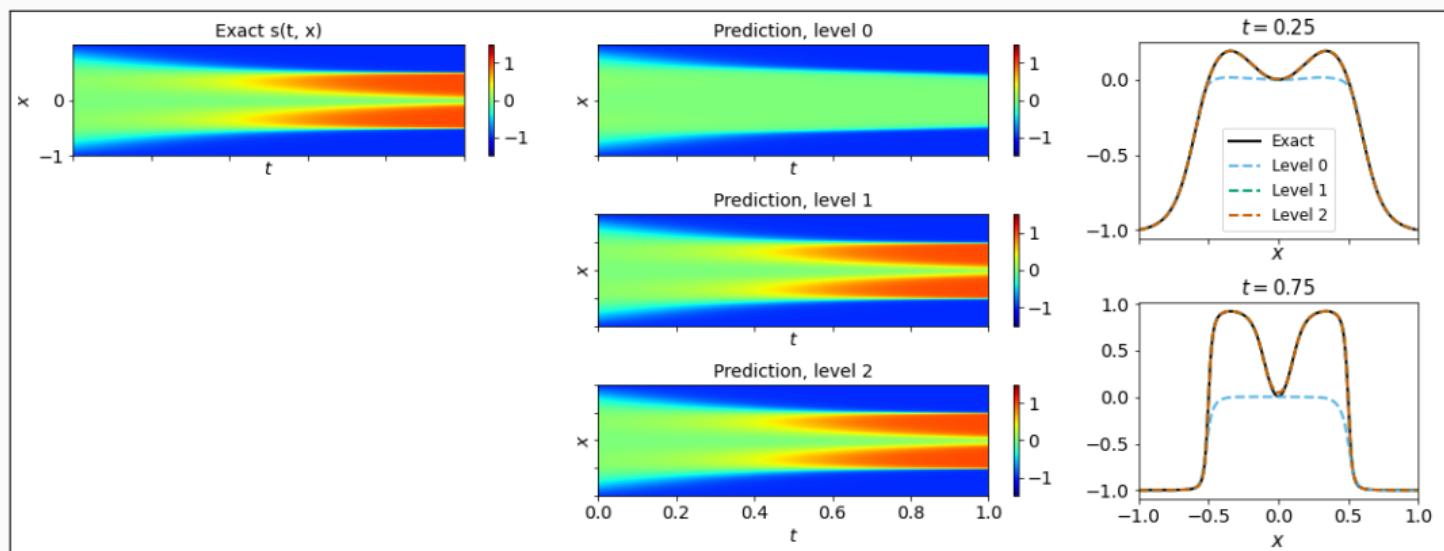
$$s_x(x, t) = s_x(-x, t),$$

$$t \in (0, 1], x \in [-1, 1],$$

$$x \in [-1, 1],$$

$$t \in [0, 1], x = -1, x = 1,$$

$$t \in [0, 1], x = -1, x = 1.$$



PINNs

- **Training of PINNs is often problematic** when:
 - scaling to large domains / high frequency solutions
 - multiple loss terms have to be balanced
- **Convergence of PINNs has yet to be understood better**

(Multilevel) FBPINNs

- Schwarz domain decomposition approaches **improve the scalability of PINNs to large domains / high frequencies**, **keeping the complexity of the local networks low**
- As classical domain decomposition methods, **one-level FBPINNs are not scalable to large numbers of subdomains**; multilevel FBPINNs **enable scalability**.

Multifidelity stacking FBPINNs

- The **combination of multifidelity stacking PINNs with the multilevel FBPINN approach** yields **significant improvements in the accuracy and efficiency** for time-dependent problems.

Thank you for your attention!

Workshop on Computational and Mathematical Methods in Data Science 2024

Details

Date: April 24-26 2024

Location: Delft University of Technology

This workshop brings together scientists from mathematics, computer science, and application areas working on computational and mathematical methods in data science.

Confirmed invited speakers

- Christoph Brune (University of Twente)
- Victorita Dolean (TU Eindhoven)
- Thomas Richter
- ...

For more details, see

<https://searhein.github.io/gamm-cominds-2024/>



TuDelft

4TU.AMI

COMiDS Workshop 2024



Workshop on Computational and Mathematical Methods in Data Science 2024
Delft University of Technology, April 25-26, 2024

About the Workshop

Welcome to the **Workshop on Computational and Mathematical Methods in Data Science 2024**. It is the 2024 edition of the annual workshop of the **GAMM Activity Group on "Computational and Mathematical Methods in Data Science"** (**COMiDS**) and is co-organized by the **Strategic Research Initiative "Bridging Numerical Analysis and Machine Learning"** of the **4TU Applied Mathematics Institute (AMI)**. The workshop will be hosted by **Delft University of Technology** and take place on April 25 and 26, 2024.

This workshop brings together scientists from mathematics, computer science, and application areas working on computational and mathematical methods in data science.

The meeting will be organized under the support of

- the **4TU Applied Mathematics Institute (AMI)** and
- the **TU Delft Institute for Communications, Science and Engineering (I-ComSE)**

