Approximately well-balanced Discontinuous Galerkin methods using bases enriched with Physics-Informed Neural Networks

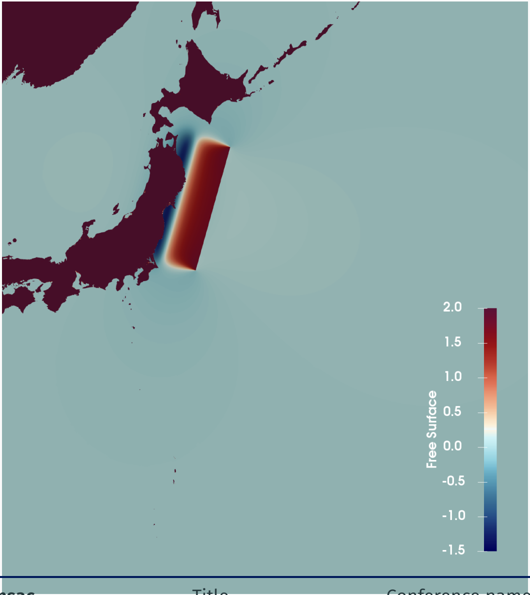
Emmanuel Franck*, Victor Michel-Dansac*, Laurent Navoret*

March 06, 2024
First Workshop on Machine Learning for Fluid Dynamics, **Paris**

*TONUS (soon-to-be MACARON) project-team, Université de Strasbourg, CNRS, Inria, IRMA, France

Tsunami simulation: initial condition

Tsunami initialization

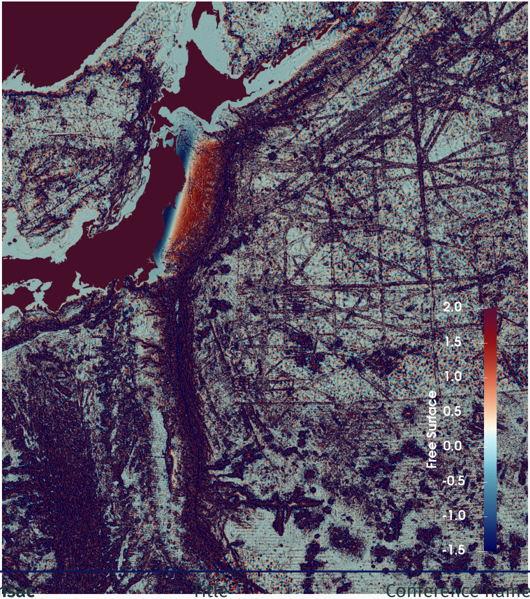


Tsunami simulation: naive numerical method

Starting the simulation with a naive numerical method

Tsunami simulation: naive numerical method

Starting the simulation with a naive numerical method



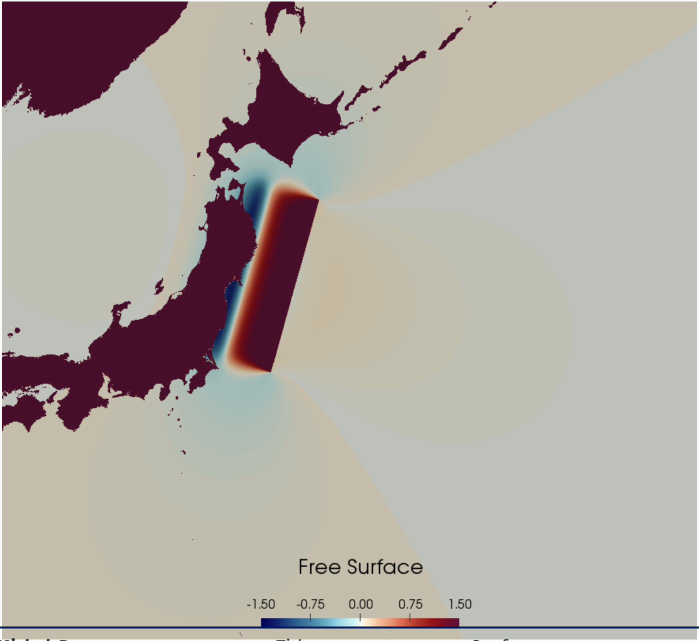
Tsunami simulation: failure

⇝ **The simulation is not usable!**

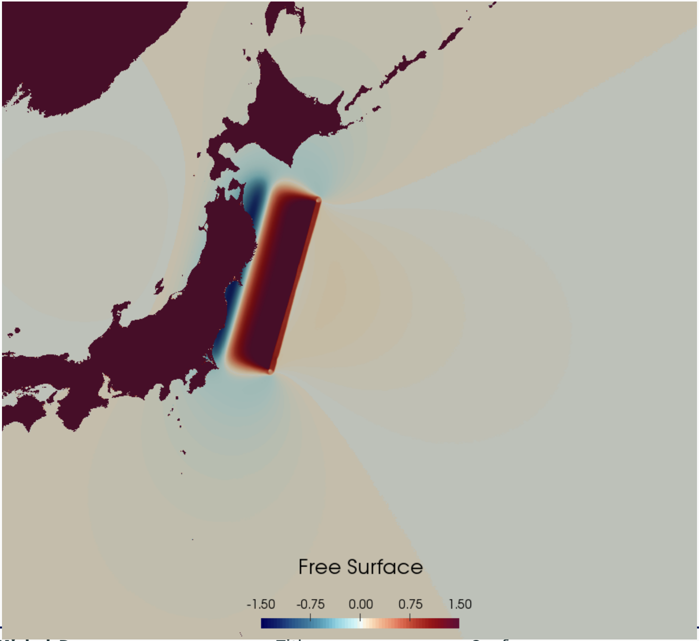
Indeed, the ocean at rest, far from the tsunami, started spontaneously producing waves.

This comes from the non-preservation of stationary solutions, hence the need to develop numerical methods that **preserve stationary solutions**: so-called **well-balanced** methods.

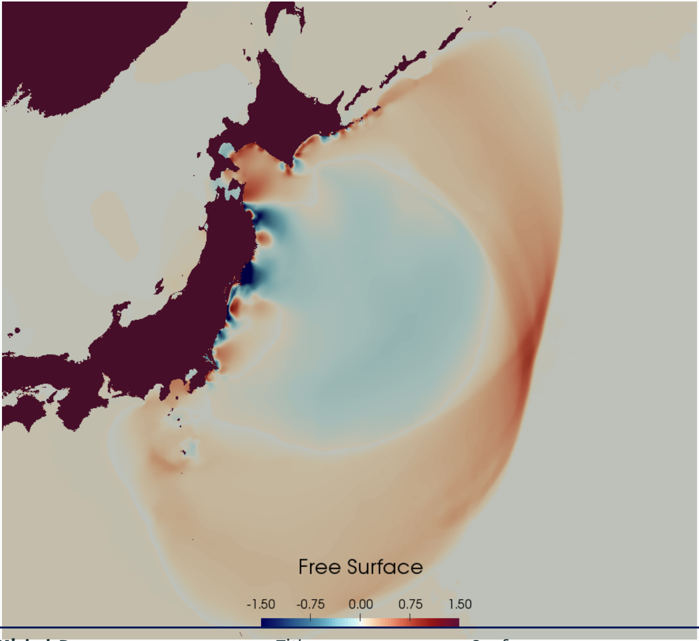
Tsunami simulation: well-balanced method



Tsunami simulation: well-balanced method



Tsunami simulation: well-balanced method

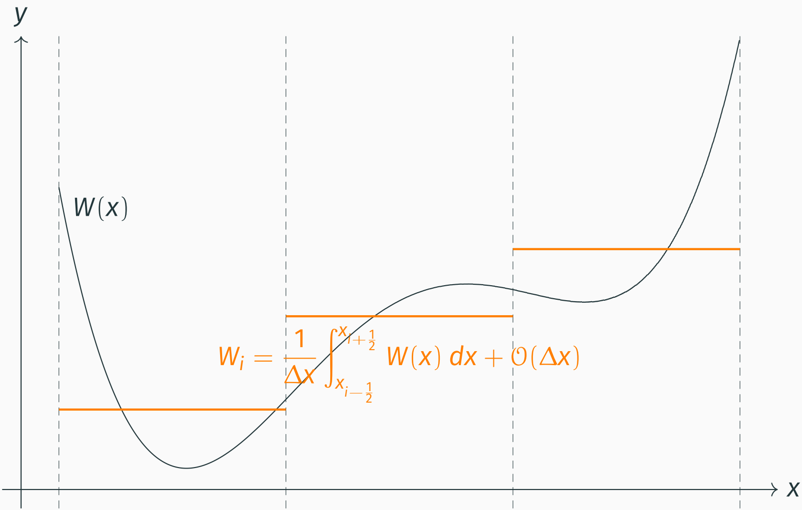


Objectives

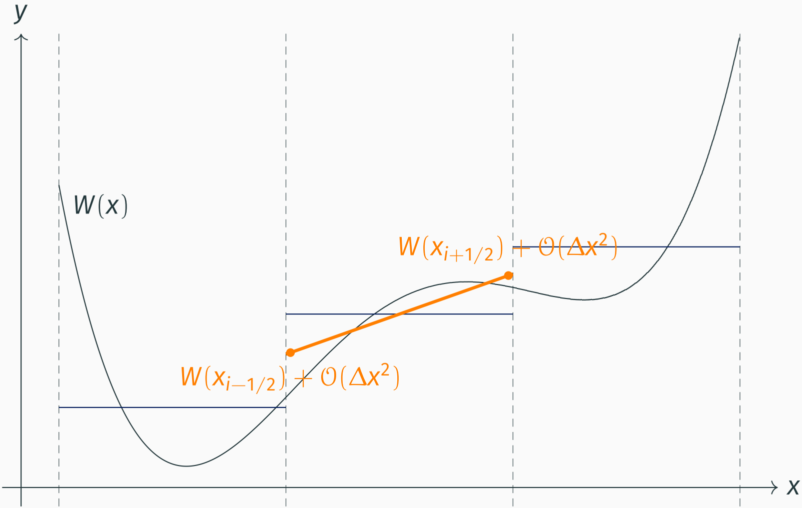
The goal of this work is to provide a numerical method which:

- is able to deal with generic hyperbolic systems of balance laws,
- can provide a very good approximation of families of steady solutions,
- is as accurate as classical methods on unsteady solutions,
- with provable convergence and error estimates.

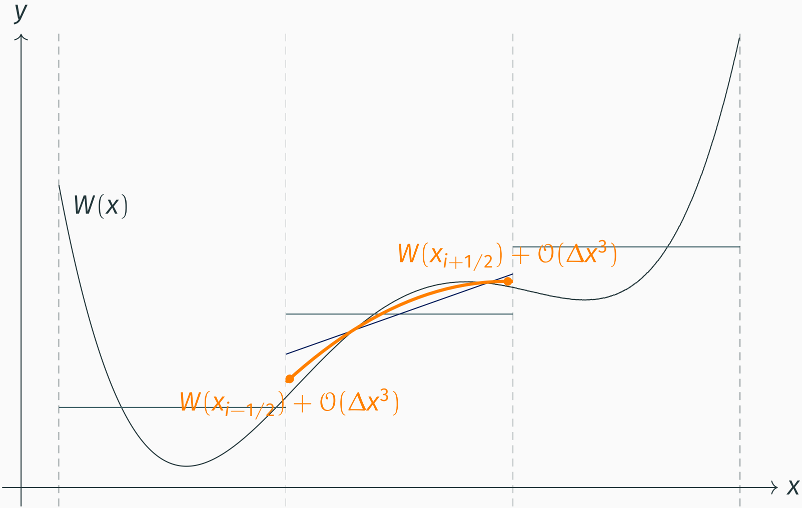
Finite volume method, visualized



Discontinuous Galerkin, visualized



Discontinuous Galerkin, visualized



Discontinuous Galerkin: an example

On the previous slide, the data W is represented by

- a polynomial of degree 2 in each cell (Galerkin approximation),
- which is Discontinuous at interfaces between cells.

Discontinuous Galerkin: an example

On the previous slide, the data W is represented by

- a polynomial of degree 2 in each cell (Galerkin approximation),
- which is Discontinuous at interfaces between cells.

Therefore, in each cell Ω_i, W is approximated by

$$W\big|_{\Omega_i} \simeq W_i^{\mathrm{DG}} := \alpha_0 + \alpha_1 x + \alpha_2 x^2 = \sum_{j=0}^{2} \alpha_j x^j,$$

where the polynomial coefficients α_0, α_1 and α_2 are determined to ensure fitness between the continuous data and its polynomial approximation.

Any polynomial of degree two can be exactly represented this way.

Discontinuous Galerkin: polynomial basis

More generally, we define a polynomial basis $\varphi_0, \dots, \varphi_N$ on each cell Ω_i and approximate the solution in this basis.

A usual example is the following so-called **modal basis**:

$$\forall j \in \{0, \dots, N\}, \quad \varphi_j(x) = x^j.$$

Discontinuous Galerkin: polynomial basis

More generally, we define a polynomial basis $\varphi_0, \dots, \varphi_N$ on each cell Ω_i and approximate the solution in this basis.

A usual example is the following so-called **modal basis**:

$$\forall j \in \{0, \dots, N\}, \quad \varphi_j(x) = x^j.$$

Main takeaway: The DG scheme is exact on every function that can be exactly represented by the basis.

Main idea

Main idea

Enhance the DG basis by using the steady solution!

$\rightsquigarrow$ If **the basis contains the steady solution**, then the enhanced DG scheme will be exact on this steady solution: it will be **well-balanced**.

In practice, the basis will contain an approximation of the steady solution, making the scheme **approximately well-balanced**.

Enhanced DG bases

Assume that you know a **prior** $\overline{W}$ on the steady solution.

The goal is now to **enhance the modal basis** V using $\overline{W}$:

$$V = \{1, x, x^2, \dots, x^N\}.$$

Enhanced DG bases

Assume that you know a **prior** $\overline{W}$ on the steady solution.

The goal is now to **enhance the modal basis** V using $\overline{W}$:

$$V = \{1, x, x^2, \dots, x^N\}.$$

First possibility: multiply the whole basis by $\overline{W}$

$$\bar{V}_* = \{\overline{W}, x\,\overline{W}, x^2\,\overline{W}, \dots, x^N\,\overline{W}\}.$$

Second possibility: replace the first element with $\overline{W}$

$$\bar{V}_+ = \{\overline{W}, x, x^2, \dots, x^N\}.$$

Convergence and error estimate

Denoting by

- P_* the projector onto basis $\bar{V}_*$,
- q the order of the DG scheme,
- Δx the step size,

we prove the following result for a scalar problem:

$$\|W - P_*(W)\|_{L^2(\Omega)} \lesssim \left| \frac{W}{\overline{W}} \right|_{H^{q+1}(\Omega)} \Delta x^{q+1} \|\overline{W}\|_{L^\infty(\Omega)}.$$

Namely, we prove that the prior $\overline{W}$ needs to provide a **good approximation of the derivatives** of the steady solution (in addition to the steady solution itself).

⇝ A Physics-Informed Neural Network (**PINN**) is the ideal candidate!

Parametric PINNs

A parametric PDE is nothing but the following problem, with $\mathcal{D}$ a differential operator:

$$\text{find } W \text{ such that } \mathcal{D}(W, x; \boldsymbol{\mu}) = 0 \text{ for all } x \in \Omega \text{ and } \boldsymbol{\mu} \in \mathbb{P} \subset \mathbb{R}^m.$$

The parametric PINN $W_\theta(x; \boldsymbol{\mu})$ should approximately satisfy the above PDE, and the problem becomes:

$$\text{find } \theta_{\text{opt}} \text{ such that } \mathcal{D}(W_{\theta_{\text{opt}}}, x; \boldsymbol{\mu}) \simeq 0 \text{ for all } x \in \Omega \text{ and } \boldsymbol{\mu} \in \mathbb{P} \subset \mathbb{R}^m.$$

Parametric PINNs

A parametric PDE is nothing but the following problem, with $\mathcal{D}$ a differential operator:

$$\text{find } W \text{ such that } \mathcal{D}(W, x; \boldsymbol{\mu}) = 0 \text{ for all } x \in \Omega \text{ and } \boldsymbol{\mu} \in \mathbb{P} \subset \mathbb{R}^m.$$

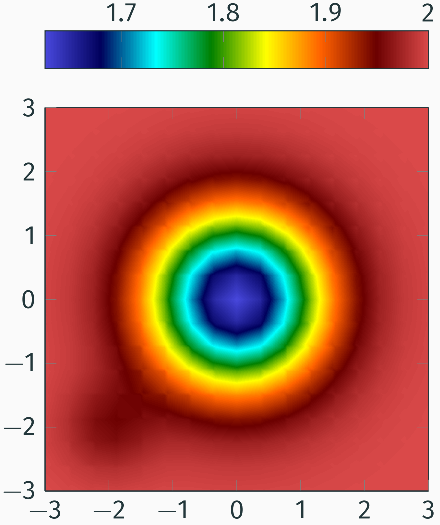
The parametric PINN $W_\theta(x; \boldsymbol{\mu})$ should approximately satisfy the above PDE, and the problem becomes:

$$\text{find } \theta_{\text{opt}} \text{ such that } \mathcal{D}(W_{\theta_{\text{opt}}}, x; \boldsymbol{\mu}) \simeq 0 \text{ for all } x \in \Omega \text{ and } \boldsymbol{\mu} \in \mathbb{P} \subset \mathbb{R}^m.$$

The minimization problem for parametric PINNs is the following:

$$\theta_{\text{opt}} = \underset{\theta}{\text{argmin}} \int_{\mathbb{P}} \int_{\Omega} \|\mathcal{D}(W_\theta, x; \boldsymbol{\mu})\|_2^2 \, dx d\boldsymbol{\mu}.$$

Perturbation of a shallow water steady solution



initial condition:
steady solution with perturbation

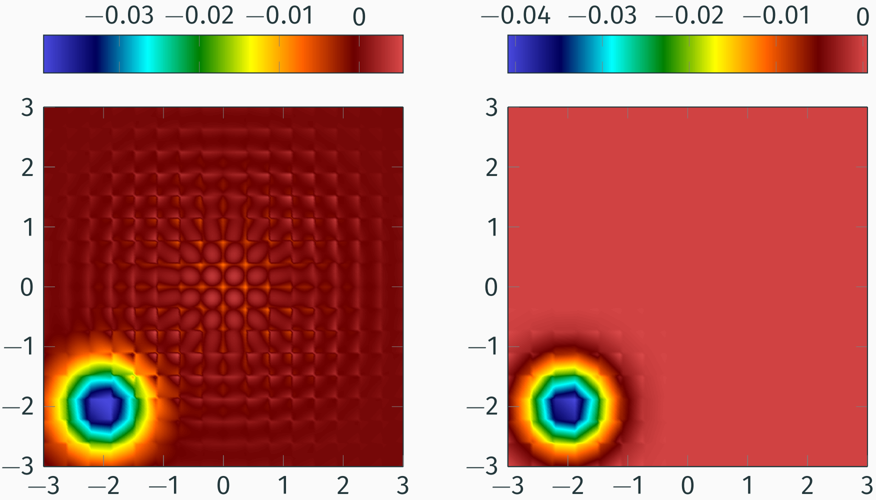
PINN trained on a parametric steady solution, driven by the topography

$$Z(x;\boldsymbol{\mu}) = \Gamma \exp\Big(\alpha\big(r_0^2 - \|x\|^2\big)\Big),$$

with physical parameters

$$\boldsymbol{\mu} \in \mathbb{P} \iff \begin{aligned} \alpha &\in [0.25, 0.75], \\ \Gamma &\in [0.1, 0.4], \\ r_0 &\in [0.5, 1.25]. \end{aligned}$$

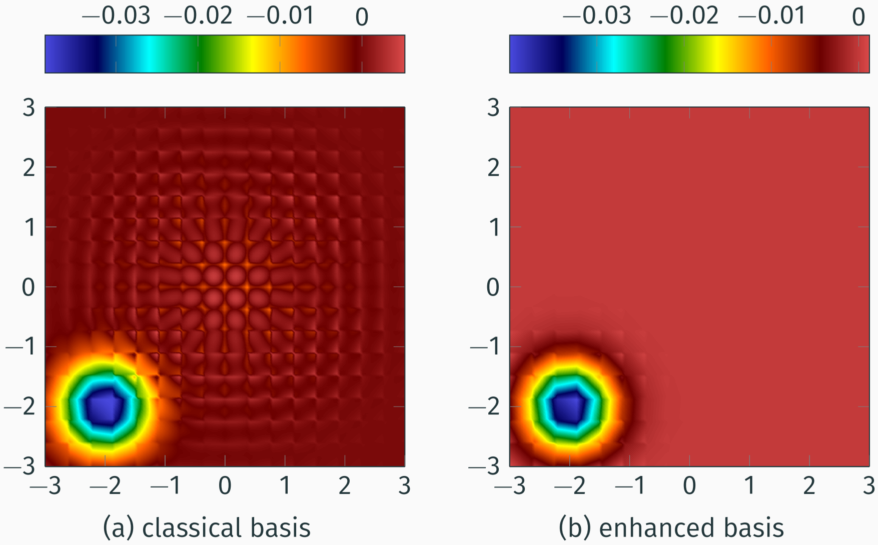
Perturbation of a shallow water steady solution



(a) classical basis

(b) enhanced basis

Perturbation of a shallow water steady solution



Conclusion and perspectives

We have obtained:

- an approximately well-balanced DG scheme,
- for parameterized families of steady solutions,
- which works for arbitrary hyperbolic balance laws.

Perspectives include:

- using a space-time DG method and time-dependent priors,
- replacing PINNs with neural operators for added flexibility,
- coding the method in the SciMBA framework.

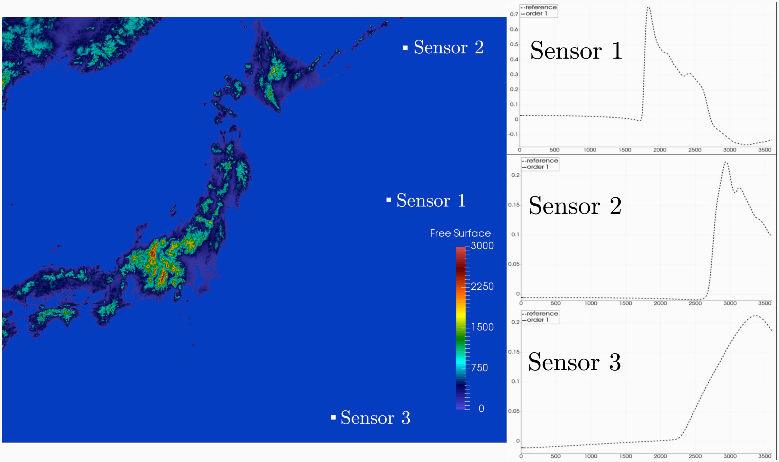
Related preprint: E. Franck, V. Michel-Dansac and L. Navoret.
"Approximately WB DG methods using bases enriched with PINNs."
git repository: https://github.com/Victor-MichelDansac/DG-PINNs

We often have **open positions** (Master theses, PhD students or postdocs). Please do not hesitate to contact us!

Thank you for your attention!

Ingredients required for a numerical simulation

Fourth step: Verification of the numerical results

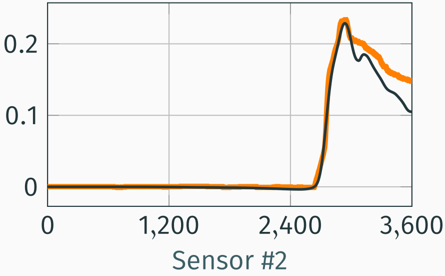


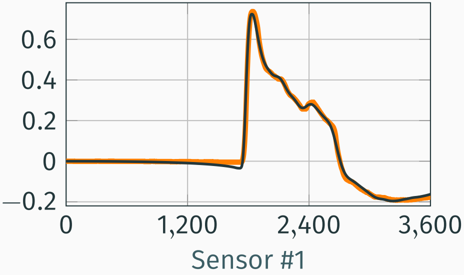
Simulation of the 2011 Japan tsunami

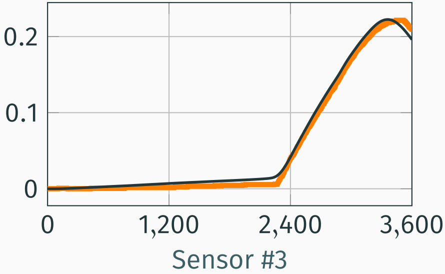
Water depth at sensors:

- #1: 5700 m;
- #2: 6100 m;
- #3: 4400 m.

Plots of the time variation
of the water depth (in meters).
data in black, simulation in orange



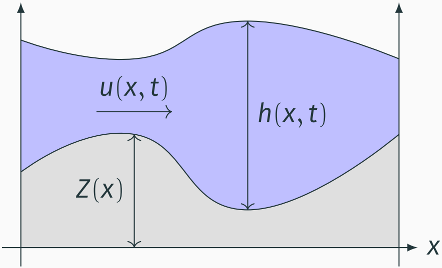




The shallow water equations

The shallow water equations are governed by the following PDE:

$$\begin{cases} \partial_t h + \partial_x q = 0, \\ \partial_t q + \partial_x \left(\dfrac{q^2}{h} + \dfrac{1}{2} g h^2 \right) = -g h \partial_x Z(x). \end{cases}$$

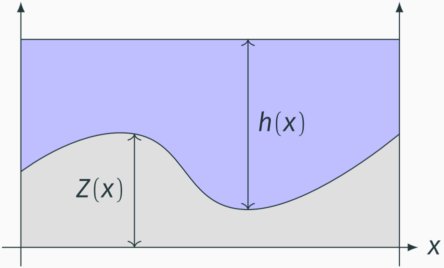


- $h(x,t)$: water depth
- $u(x,t)$: water velocity
- $q = hu$: water discharge
- $Z(x)$: known topography
- g: gravity constant

The shallow water equations: steady solutions

The steady solutions of the shallow water equations are governed by the following ODEs:

$$\begin{cases} \partial_x q = 0, \\ \partial_x \left(\dfrac{q^2}{h} + \dfrac{1}{2} g h^2 \right) = -g h \partial_x Z(x). \end{cases}$$



For the shallow water equations, if the velocity vanishes, we obtain the lake at rest steady solution:

$$h + Z = \text{cst}.$$

PINNs

Remark: Neural networks are smooth functions of the inputs (provided smooth activation functions are used!).

Since their derivatives are easily computable by automatic differentiation, they are therefore **natural objects to approximate solutions to PDEs or ODEs**.

PINNs

Remark: Neural networks are smooth functions of the inputs (provided smooth activation functions are used!).

Since their derivatives are easily computable by automatic differentiation, they are therefore **natural objects to approximate solutions to PDEs or ODEs**.

Definition: PINN

A PINN is a neural network with input x and trainable weights θ, approximating the solution to a PDE or ODE, and denoted by $W_\theta(x)$.

Hence, the PINN W_θ will approximate the solution to the PDE

$$\mathcal{D}(W, x) = 0,$$

with $\mathcal{D}$ a differential operator.

PINNs: loss function

Omitting boundary conditions, the problem becomes

$$\text{find } W \text{ such that } \mathcal{D}(W, x) = 0 \text{ for all } x \in \Omega \subset \mathbb{R}^d.$$

Based on this observation, the PINN W_θ should approximately satisfy the above PDE, and the problem becomes:

$$\text{find } \theta_{\text{opt}} \text{ such that } \mathcal{D}(W_{\theta_{\text{opt}}}, x) \simeq 0 \text{ for all } x \in \Omega \subset \mathbb{R}^d.$$

PINNs: loss function

Omitting boundary conditions, the problem becomes

$$\text{find } W \text{ such that } \mathcal{D}(W, x) = 0 \text{ for all } x \in \Omega \subset \mathbb{R}^d.$$

Based on this observation, the PINN W_θ should approximately satisfy the above PDE, and the problem becomes:

$$\text{find } \theta_{\text{opt}} \text{ such that } \mathcal{D}(W_{\theta_{\text{opt}}}, x) \simeq 0 \text{ for all } x \in \Omega \subset \mathbb{R}^d.$$

The idea behind PINNs training is to find the optimal weights θ_{opt} by **minimizing a loss function built from the ODE residual**:

$$\theta_{\text{opt}} = \underset{\theta}{\operatorname{argmin}} \int_\Omega \|\mathcal{D}(W_\theta, x)\|_2^2 \, dx.$$

The Monte-Carlo method is used for the integrals, which makes the whole approach **mesh-less** and able to deal with **parametric PDEs**.

PINNs: advantages and drawbacks

Once trained, PINNs with Monte-Carlo integration are able to

- quickly provide an approximation to the steady solution,
- in a mesh-less fashion,
- independently of the dimension.

PINNs: advantages and drawbacks

Once trained, PINNs with Monte-Carlo integration are able to

- quickly provide an approximation to the steady solution,
- in a mesh-less fashion,
- independently of the dimension.

However, PINNs

- have trouble generalizing to $x \notin \Omega$;
- are **not competitive with classical numerical methods for computational fluid dynamics**: to reach a given error (if possible), training takes longer than using a classical numerical method.

PINNs: advantages and drawbacks

Once trained, PINNs with Monte-Carlo integration are able to

- quickly provide an approximation to the steady solution,
- in a mesh-less fashion,
- independently of the dimension.

However, PINNs

- have trouble generalizing to $x \notin \Omega$;
- are **not competitive with classical numerical methods for computational fluid dynamics**: to reach a given error (if possible), training takes longer than using a classical numerical method.

The most interesting use of PINNs, in our case, is to deal with **parametric ODEs and PDEs**, where dimension-insensitivity is paramount.