

"Real-Time" Simulations Methods for Flood Events and Data Assimilation

* A Computational Sciences Talk *

by J. Monnier (Prof. INSA-IMT) with :

M. Allabou (PhD INSA-IMT 2026), H. Boulenc (PhD INSA-IMT 2026, *ANR fund*),
T.-V. Nguyen (PhD INSA-IMT started 2025), R. Bouclier (Prof. INSA-ICA/IMT)
and P.-A. Garambois (Res. INRAe Aix).

Outline

- ▷ Goal : simulations in seconds, including data assimilation.
- ▷ Direction #1 : Models reduction by physics - machine learning approaches.
↳ Reduced order models (ROMs).
- ▷ Direction #2 : Model calibration by Physics Informed Neural Networks (PINNs).
↳ Inverse problems - data assimilation.
- ▷ Conclusion & Perspectives.



Goal : flood simulations in "real-time", with data assimilation

▷ Goal.

High-resolution (HR) hydraulic model
with data assimilation (DA)

Reduced order model (ROM)
Simulations in seconds

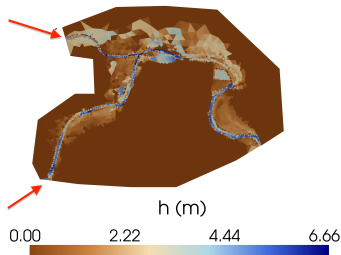
Assimilation of new data
on the fly

▷ Example of a reduced model (2D shallow-water),

here without DA : Aude-Fresquel rivers.

Results : M. Allabou's PhD (INSA-IMT, defended in 2026).

Data case : [Villeneuve et al., 2023], Vigicrues, SCHAPI-DGPR, Meteo-France.



Computational times, for a 52 000 cells mesh.

HR simulation	ROM
≈ 1h15mn (CPU)	≈ 10s (CPU) / 10^{-2} s (GPU)

Laptop CPU, single-core mode, 11th Gen Intel Core i9-11950H processor (2.60 GHz) with 32 GB RAM.

Direction #1 :

Models reduction by physics - machine learning approaches.

↪ Reduced order models (ROMs)

High-Resolution (HR) model & Reduced Order Model (ROM)

Basic principles

▷ High-Resolution (HR) model : μ -parametrized PDEs

Given the parameter $\mu \in \mathcal{P}$, $\mathcal{P} \subset \mathbb{R}^p$, find the state of the system $\mathbf{u}(\mu)(x, t)$ such that :

$$\mathcal{A}(\mu; \mathbf{u}(\mu))(x, t) = \mathcal{F}(\mu)(x, t)$$

For the 2D shallow water (SW) model, the state $\mathbf{u} = (h, q_x, q_y)(x, t)$.

μ can be : boundary value parameters (eg. **hydrograph**), source term parameters (eg. **rain field**), and so on.

▷ **Reduced order model (ROM)** aims at providing an approximation of the HR solution at very low cost.

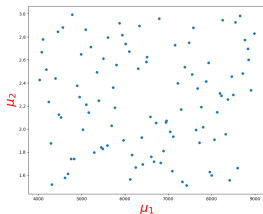
Challenge : Build a **ROM cheap in computational-time, while preserving accuracy** of the HR model.

Reduced Order Model (ROM) : Offline phase

▷ Compute $\mathcal{O}(10^p)$ HR simulations = "snapshots".

p potentially large, depending on $\dim(\mu)$.

Each snapshot $\mathbf{u}(x, t)$ corresponds to a different value of μ .



Plot : $\mu \in \mathbb{R}^2$

HR solutions

Snapshot matrix, $(N_s \times N_t) \times N_h$
 $S = [\mathbf{u}(\mu_1, t), \dots, \mathbf{u}(\mu_{N_s}, t)]$

POD / PCA

Reduced eigenvectors basis

↪ Rectangular basis change matrix $N_h \times N_{rb}$

$$\mathbf{B}_{rb} = [\xi_1(x), \dots, \xi_{N_{rb}}(x)]$$

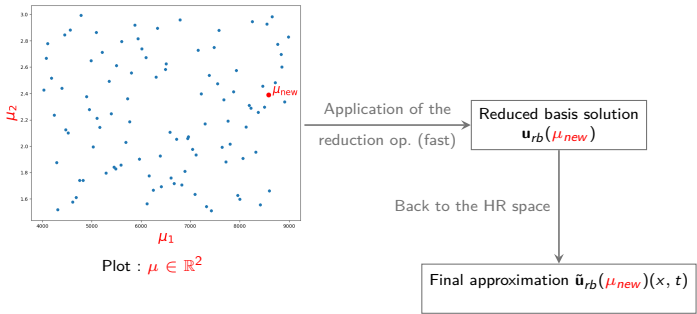
$$\mathbf{u}_{rb}(\mu)(x, t) = \sum_{i=1}^{N_{rb}} \underbrace{\alpha_i(\mu)(t)}_{\text{RB coeff.}} \xi_i(x)$$

For linear PDE models : standard POD reduction method.

Non-affine parametrization in μ ? Additional empirical interpolation techniques is required, see eg. [Quarteroni&Rozza's book].

Reduced Order Model (ROM) : Online phase

▷ Given a **new** parameter value μ_{new} , compute the corresponding state $\mathbf{u}_{rb}(\mu_{new})$ in "real-time" (\approx second).



▷ In the end, do we have $\tilde{\mathbf{u}}_{rb}(\mu_{new}) \approx \mathbf{u}(\mu_{new})$?

For non-linear models, challenge = construction of fast and accurate "reduction operators" (Parameter-to-RB solution operators)

$$\underbrace{\mu \in \mathcal{P} \subset \mathbb{R}^p}_{\text{parameter p-dim}} \mapsto \underbrace{\mathbf{T} \mathbf{B}_{rb} \mathbf{u}_h \in \mathbb{R}^{N_{rb}}}_{\text{solution in RB, } N_{rb} = \mathcal{O}(10)}$$

Reduction operators for **nonlinear** PDEs

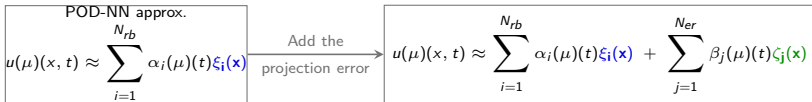
A few recent approaches

▷ **Basic method : POD-NN** (Neural Network) proposed by [Hesthaven et al. 2018].

Given N_s snapshots, the P2RB operator is learned as a NN $\mathcal{N}_\theta : \mu \mapsto \mathbf{B}_{rb}^T \mathbf{u}_h$.

Data=the snapshots $\mathbf{u}_h^{(s)}$. Loss function $\mathcal{L}(\theta) = \frac{1}{N_s} \sum_{s=1}^{N_s} \|\mathcal{N}_\theta(\mu^s) - \mathbf{B}_{rb}^T \mathbf{u}_h^{(s)}\|_2^2$. Training : $\min_\theta \mathcal{L}(\theta)$.

▷ **Enriched version : Error-Aware (EA - POD-NN)** as proposed by [Allabou et al. 2024].



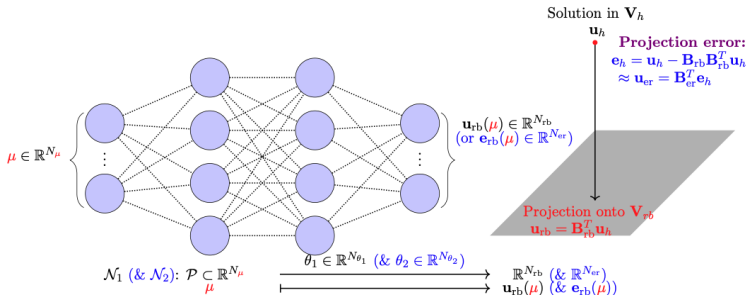
- **Online phase** : given a new parameter value μ_{new} ,

$$\mathbf{u}_{rb}(\mu_{new}) = \left(\mathbf{B}_{rb} \alpha + \underbrace{\mathbf{B}_{er} \beta}_{\text{corrective term}} \right) \text{ with } \mathbf{B}_{er} = \underbrace{[\zeta_1, \dots, \zeta_{N_{er}}]}_{\text{POD of the proj. error}}$$

\mathbf{B}_{er} : a 2nd POD-based matrix of basis change, computable in parallel of \mathbf{B}_{rb} .

Reduction operators for **nonlinear** PDEs

The Error-Aware POD-NN method as proposed by Allabou's PhD



$$\mathcal{J}_1(\theta_1) = \frac{1}{N_{train}} \sum_{s=1}^{N_{train}} \|\mathcal{N}_1(\theta_1)(\mu_s) - \mathbb{B}_{rb}^T \mathbf{u}_h(\mu_s)\|_2^2 ; \quad \mathcal{J}_2(\theta_2) = \frac{1}{N_{train}} \sum_{s=1}^{N_{train}} \|\mathcal{N}_2(\theta_2)(\mu_s) - \mathbb{B}_{er}^T \mathbf{e}_h(\mu_s)\|_2^2.$$

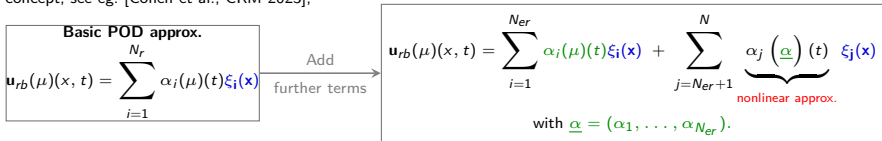
- ▷ \mathcal{N}_1 is train to learn the projection of the snapshots (= the HR solutions = the learnt data) onto the reduced basis,
- ▷ \mathcal{N}_2 is train to learn the error of the projection.
- ▷ Basic idea \sim multigrid-like approach.

Recall. These NNs are trained from the snapshots (=the HR solutions). They can be trained in parallel.

Reduction operators for nonlinear PDEs

Another version : the Non-Linear Compressive (NLC) method as proposed by [Cohen et al., CRM 2023]

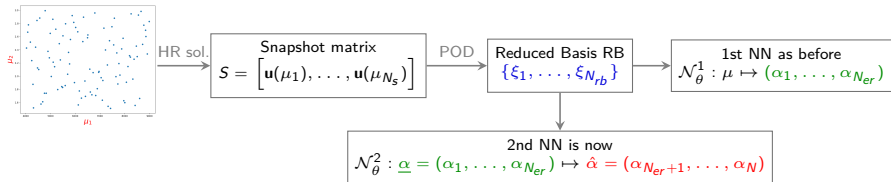
Based on mathematical estimations related to the Gelfand width concept, extension of the Kolmogorov width concept, see eg. [Cohen et al., CRM 2023],



▷ Basic idea ~ multi-scale approach.

Nonlinear prediction of the $(N - N_{er})$ missing coefficients from the N_{er} POD-based retained modes.

▷ **Using NNs as non-linear predictors**, (this could be e.g. random forests, decision tree too)



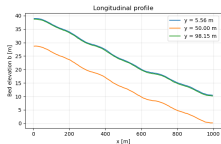
The loss function for \mathcal{N}_θ^2 is : $\mathcal{L}_2(\theta) = \frac{1}{N_s} \sum_{s=1}^{N_s} \|\mathcal{N}_\theta^2(\mu^s) - \hat{\alpha}^s\|_2^2$.

Reduced Order Model (ROM)

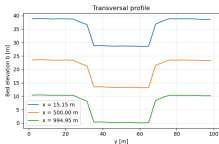
A basic academic case

$\mu \in \mathbb{R}^3$: single inflow discharge parametrization $(Q_{max}, t_{max}) \in \mathbb{R}^2 \oplus t_s$.

Bathymetry & cross-sections profiles :
 main channel + overbank areas



Longitudinal profiles



Transversal profiles

Boundary conditions

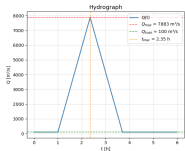
Left side : inflow discharge $Q_{in}(t)$
 parameterized by Q_{max} and t_{max} .

Right side : imposed water level.

Numerical parameter values

Variables	Values
Q_{min}	$100 \text{ m}^3/\text{s}$
Q_{max} range	$[4000, 9000] \text{ m}^3/\text{s}$
Final simulation time	6 h
t_{rise}	1 h
t_{max} range	$[1.5, 2.4] \text{ h}$
N_t	33
N_s snapshots	$10^2 \times 33 = 3\,300$

Inflow discharge $Q_{in}(t)$ example



Simple academic channel case

The flow at a given instant

Longitudinal Curve Comparison (at $t = 2.17$ h)

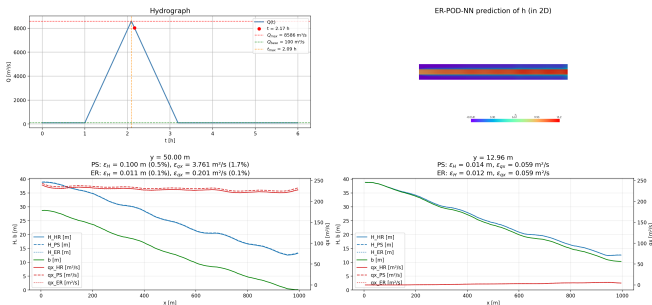
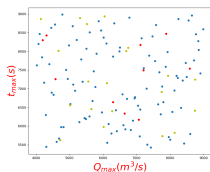


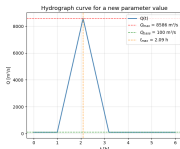
FIGURE — Longitudinal profiles : (L) along the main channel ; (R) along the overbank.
 Curves : bathymetry $b(x)$, surface elevation $H(x)$ and $q_x(x)$ from (HR & reduced) models.

Simple academic channel case

Comparisons of 4 ROM for tiny mesh and $\dim(\mu)$



Parameter distribution (t_{max} , Q_{max}) used for snapshot generations



Inflow discharge $Q_{in}(t)$ corresponding to the new parameter value μ_{new} , $\mu_{new} \in \mathbb{R}^2 \oplus t_s$.

▷ **Computational time comparison** for 2673 cells. Computations on a single NVIDIA RTX 2000 Ada Generation GPU (8 GB VRAM, 60 W TDP), running CUDA 13.0.

Method		POD-NN	Error Aware (EA) -POD-NN	Non Linear Compressive (NLC)	Extra Residual (ER) -POD-NN
Offline phase	Generation of each snapshot (CPU-time) (same time for all methods)	158.2 s	158.2 s	158.2 s	158.2 s
	RB construction - POD (GPU-time)	2.4 s	3.8 s	2.4 s	2.4 s
	NN trainings (GPU-time)	31.8 s	98.5 s	46.8 s	980.1 s
including the 100 residuals evaluation at each epoch.					
Online phase (GPU time)		139 μ s	281 μ s	286 μ s	273 μ s

▷ **Accuracy** (relative 2-norm errors wrt the reference HR solution).

Method	POD-NN	EA-POD-NN	NLC	ER-POD-NN
depth h	0.97 %	0.91 %	0.91 %	0.39 %
longitudinal discharge q_x	1.17 %	1.15 %	1.15 %	0.30 %
cross-discharge q_y	8.69 %	4.82 %	3.34 %	8.56 %

Garonne river case (Marmande-la Reole station)

Real-like data

▷ Configuration : same as the academic test case.

- ▶ Triangle hydrograph parametrized by $(Q_{max}, t_{max}) \Rightarrow \mu \in \mathbb{R}^2 \oplus t_s$.
- ▶ $N_s = (10^2 \times 43)$ snapshots.
(= the training database).
- ▶ Mesh of 123 960 cells.
Bathymetry data from IMFT [Roux et al.].
DassFlow2D computations by T.-V. Nguyen,
IMT-INSA.



Water height h .

Computation times

		Non Linear Compressive (NLC)
Offline phase	Generation of each snapshot (CPU-time)	$\approx 1h30$
	RB construction - POD (GPU-time)	20 s
	NN trainings (GPU-time)	55 s
Online phase (GPU time)		3 ms

Accuracy vs the HR solution

	NLC
height h	0.69 %
discharge q_x	1.28 %
discharge q_y	1.25 %

Reduction operators for **nonlinear** PDEs

Conclusion of this part

▷ Recalls :

The data used for training are the snapshots.

Each snapshot is a finely tuned and expensive HR space-time simulation.

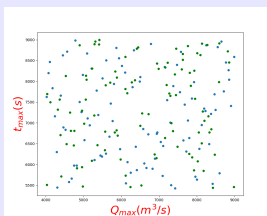
And, the parameter space $\mathcal{P} \subset \mathbb{R}^p$ needs to be covered by the snapshots...

⊕ The good news.

- ▶ **Reduction of the 2D shallow water model** in real-world geometries (nonlinear hyperbolic system with complex wave interactions) is **possible with good accuracy** [M. Allabou's PhD 2026, T.-V. Nguyen On-Going].
- ▶ The methods aforementioned enables parallel training of the NNs (\mathcal{N}_θ^1 and \mathcal{N}_θ^2).

⊖ What can be an issue.

- ▶ The number of necessary snapshots is related to $\dim(\mu)$, therefore potentially large... Say $O(10^p)$ with p related to $\dim(\mu)$...



On-going improvements :
densify the parameter space
using residuals of the model
only (vs additional snapshots).

T.-V. Nguyen's PhD, INSA-IMT,
ANITI-CNES funds, started in 2026.

Direction #2 :

Model calibration in "real-time" ?
Using Physics Informed Neural Networks (PINNs) ?

↪ Inverse problems - data assimilation topics...

Ref. [H. Boulenc's PhD 2026], INSA-IMT.
ANR fund. Collaboration INRAe Aix.
[Boulenc et al., Inverse Problems 2025].

Example : calibration of the infiltration coefficient K_i in a watershed.

Model : 2D shallow water with rainfall-runoff - infiltration source terms.

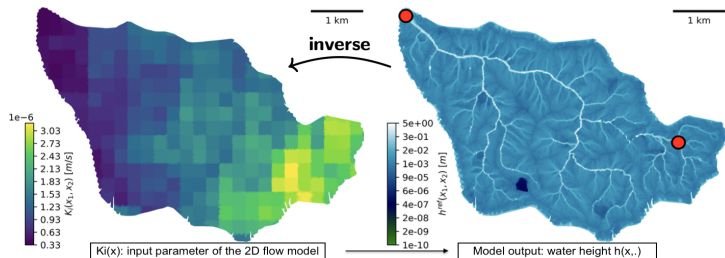


FIGURE – (Left) The spatially distributed infiltration coefficient $K_i(x)$ (Green-Ampt) (here, 279 coeffs). (Right) Water height $h(x, \cdot)$.

Case Réal-Colobrier provided by CEREMA - INRAe Aix-en-Provence. DassFlow2D computations.

▷ Calibration by the standard Variational Data Assimilation (VDA) method [DassFlow2D et al.] is effective : see [Pujol et al. AWR 2025]. Eg. given hydrographs at stations \oplus ..., infer $K_i(x)$.

Question : Can we achieve similar accuracy and robustness as VDA with much lighter computations, and a simpler methodology as well ?

Inversion - calibration by Machine Learning

Learning of the complete parameter-to-state operator

▷ k -parametrized direct model : $\mathcal{M} : k(x, t) \mapsto \mathbf{y}(k; (x, t))$.

For the 2D shallow-water model, the state of the system $\mathbf{y} = (h, q_x, q_y)$.

▷ **Inverse problems** : given data/observations \mathbf{z}^{obs} to be compared to \mathbf{y} , infer $k(x, \cdot)$.

▷ A potentially simple and efficient approach : **brute-force operator learning** as

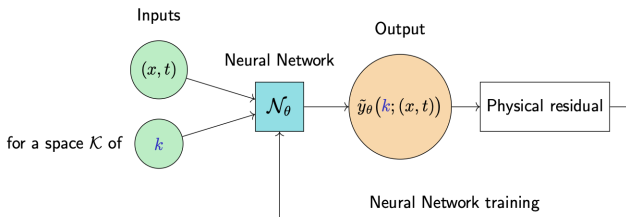


FIGURE – Operator learning by PINNs-like methodology. Residual = $\|\mathcal{A}(k; y) - \mathcal{F}(k)\|_{\Omega \times (0, T)}^2$

▷ **First limitation** : dimension of the parameter k must be small ($\dim \approx 5$ max.).

Inversion - calibration by Machine Learning

Learning of the "semi-parametrized" parameter-to-state operator

- ▷ Same k -parametrized direct model : $\mathcal{M} : k(x, t) \mapsto \mathbf{y}(k; x, t)$
- ▷ **Inverse problems** : given data/observations \mathbf{z}^{obs} to be compared to \mathbf{y} , infer $k(x, \cdot)$.
- ▷ Another simple approach : learning of the "semi-parametrized" operator (=PINNs)

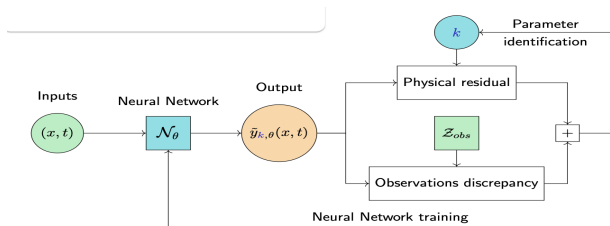


FIGURE – Operator learning by PINNs methodology. $\text{Residual} = \|\mathcal{A}(k; \mathbf{y}) - \mathcal{F}(k)\|_{\Omega \times (0, T)}^2$.

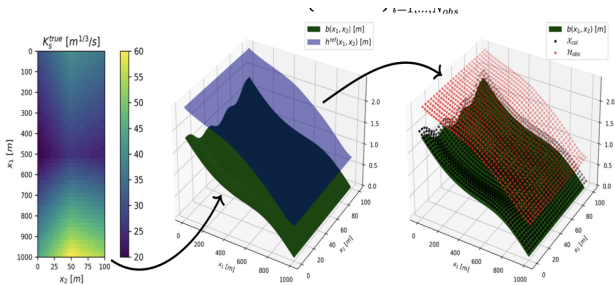
Here, simultaneous learning of the state \mathbf{y} and k from the data (data assimilation).

- ▷ **Advantage** : Semi-Parametrized version $\Rightarrow \dim(k)$ can be large e.g. $O(10^3)$.
- ▷ **Issue** : Minimizing an equation residual does not necessarily imply an accurate approximation of the solution $\mathbf{y}(x, t)$... Here $\mathbf{y} = (h, q_x, q_y)$.

Inversion - calibration by PINNs

A toy test case with extremely dense observations

- ▷ Calibration of the friction coefficient $K_s(x)$, given the (quasi-complete) knowledge of the surface elevation $H(x, \cdot)$:
 (48 × 24) surface data with (48 × 24) values of $K_s(x)$ to identify.

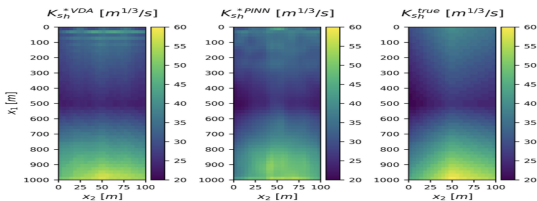


$K_s^{true}(x)$ (48 × 24), $b(x)$ and $h^{ref}(x)$ ($t = 21\,600\,s$), col. (50 × 30) and obs. (48 × 24) points.

Similar toy experiments can be performed to infer spatially-distributed infiltration coefficients $K_i(x)$.

Inversion - calibration by PINNs

Toy test case with extremely dense observations : very efficient



Left: VDA after 200 M1QN3 ite. Middle: SP-PINN after 1500 L-BFGS ite. Right: K_s^{true} .

Comparison with the traditional Variational Data Assimilation method.

[H. Boulenc's PhD].

	SP-PINN	VDA
Global relative RMSE on K_s^*	5.9%	6.2%
Computation time on 1 CPU	~ 70 s	~ 30 h
Computation time in parallel	~ 35 s	~ 5 h

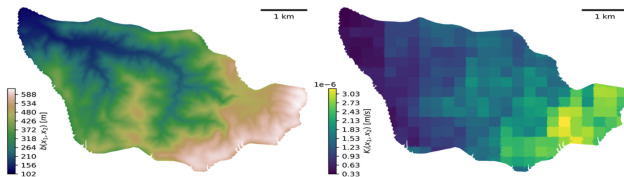
$\times \mathcal{O}(500)$

Parallel computing refers to GPU with 4096 CUDA cores and 6 GB of memory for the SP-PINN and 6 CPU threads for VDA.

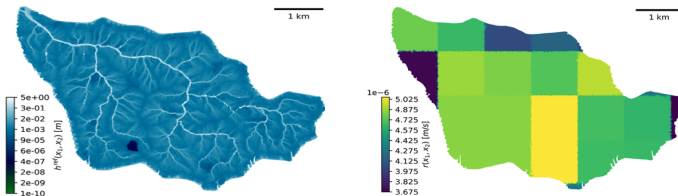
Inversion - calibration by PINNs

Back to the Réal-Colobrier test case

▷ 2D shallow-water flow, steady-state, DassFlow2D code, 53 967 cells.



Bathymetry and infiltration coefficient for the Malière test case with $\dim(K_i^{true}) = 279$.



reference water height (log scale).

inverted rain for the Malière test case.

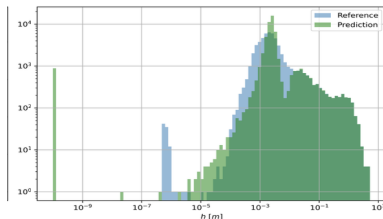
Inversion - calibration by PINNs

Back to the Réal-Colobrier watershed case

▷ 2D shallow-water flow, steady-state, DassFlow2D code, 53 967 cells.

▷ Given the complete knowledge of the flow, $(h, q_x, q_y)(x)$, the PINNs is trained to learn the steady-state flow. Optim. algorithms (ADAM + L-BFGS), during 60 000 iterations.

Fig. : Comparison of h in log scale, after training (green) vs the reference one (blue).



▷ The PINN fails to accurately represent the wide range of magnitudes for h , even if using area descriptors, recent techniques from image reconstruction and so on.

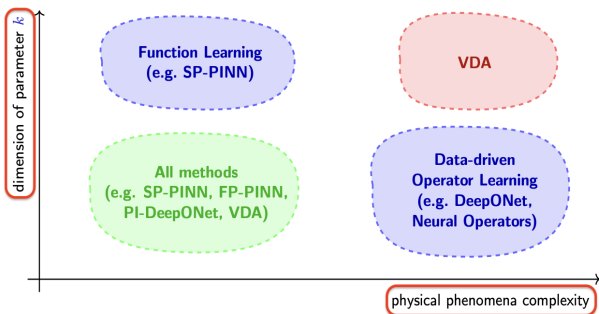
▷ Minimizing the residual $\|\mathcal{A}(k; y) - \mathcal{L}(k)\|$ does not necessarily implies a small error $\varepsilon(y) = \|y - y_{ref}\| \dots$

↪ PINN solvers vs accurate Finite Volume solvers...

Inversions - Model calibration from data

Brief outline of methods & Conclusion of this part

- ▷ For complex non-linear wave interactions like those appearing in the 2D shallow-water models / flood dynamics,
- using PINNs only, we did not manage to set up **accurate** calibrated 2D shallow water based flow models...
 - on the contrary if using variational data assimilation (VDA), [DassFlow2D et al.].



General conclusion

▷ Given a "good" numerical model, simulations corresponding to a **new configuration**, that is from μ_{new} , can be performed in **less than 1 second with acceptable accuracy**.

The basic simulations can be based on your favorite hydrodynamics software.

They were here based on DassFlow2D, which includes data assimilation capabilities.

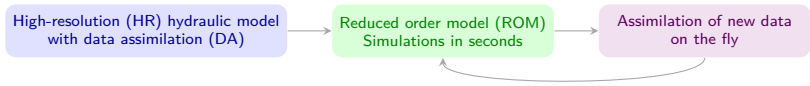
- ▷ For a **large dimensional input parameter μ** , a **preliminary reduction is required** :
 - For hydrographs, a very efficient compressing method has been elaborated in [M. Allabou's PhD 2026].
 - For rain signals, on-going work by [D. Gomez, PhD INSA-IMT / CS Lab - Sopra-Steria].

▷ Concerning **data assimilation on the fly, in "real-time"**,
 -PINNs have not provided the expected results.

Further fundamental (math structures of the system) investigations seem necessary.

- On the contrary, **Data Assimilation methods applied to ROMs constitute a promising approach**.

Our investigations based on the aforementioned understandings (ROMs-PINNs) are still active (3AI ANITI chaire), in particular through T.-V. Nguyen's PhD (INSA-IMT, ANITI-CNES funds) started in 2026.



Perspectives & Other on-going investigations

▷ Uncertainty quantification (UQ) for the aforementioned ROMs & other types of ROMs.

On-going studies with [O. Roustant Prof. INSA-IMT & I. Henderson Prof. ISAE-SupAero, Toulouse] in the context of our 3IA ANITI chaire "PiLearnWater" (Physics-Informed Learning Methods for Continental Waters and Marine Risks).

▷ Our mathematical - computational sciences - algorithm know-hows (VDA, hybrid modeling, inverse problems analyses, PINNs, ROMs, codes) have been and are still directly transferred to our partners.

In particular INRAe Aix ([P.A. Garambois et al.]), SERTIT ([L. Pujol et al.], ex, @Icube), etc, in particular for watershed river networks / H&H coupled models.

▷ Application to peri-urban and urban flood dynamics.

On-going investigations with [D. Gomez's PhD INSA-IMT / CS Lab-Sopra-Steria, 2026-2x] with / CEREMA et al.. 3AI ANITI Chaire context.

Recall, for ROMs, the basic simulations can be based on your favorite hydrodynamics software. Here, they were based on DassFlow2D, which includes high-dimensional data assimilation capabilities, as well as various features for urban flow modeling, fluids with nonlinear rheology (muds), etc.

Talk's publications

- M. Allabou, R. Bouclier, P.-A. Garambois, J. Monnier, "Reduction of the shallow water system by an error aware POD-neural network method : Application to floodplain dynamics". CMAME 2024.
- H. Boulenc, R. Bouclier, P.-A. Garmabois, J. Monnier, "Spatially-Distributed Parameter Identification by Physics-Informed Neural Networks illustrated on the Shallow-Water Equations", Inverse Problems, 2025.
- M. Allabou, PhD thesis, "Reduced-Order Modeling of the Shallow-Water Equations via an Error-Aware POD-Neural Network : Application to Flood Dynamics", INSA- Université de Toulouse, 2026.
- H. Boulenc, PhD thesis, "Solving inverse problems by physics-informed machine learning for flood dynamics", INSA-Université de Toulouse, 2026.
- T.V. Nguyen, R. Bouclier, J. Monnier. "Model reduction methods based on physico-learning". *In preparation*

Key References

- A. Quarteroni, G. Rozza. "Reduced Order Methods for Modeling and Computational Reduction". Springer, 2014.
- J. S. Hesthaven, S. Ubbiali. "Non-intrusive reduced order modeling of nonlinear problems using neural networks". Journal of Computational Physics, 2018.
- A. Cohen, C. Farhat, Y. Maday, A. Sommacal, "Nonlinear compressive reduced basis approximation for PDE's". Comptes Rendus. Mécanique, 2023.

Links

- ANITI chaire, PILearnWater ("PiLearnWater Physics-Informed Learning Methods for Continental Waters and Marine Risks") :
<https://aniti.univ-toulouse.fr/en/physics-informed-learning-methods-for-continental-waters-and-marine-risks/>
- DassFlow software : <https://github.com/DassHydro>
- MathHydroNum team-project.

Thank you for your attention.

Any questions ?