Multilevel Physics Informed Neural Networks (MPINNs)

E. Ricciietti (LIP-ENS, Lyon)

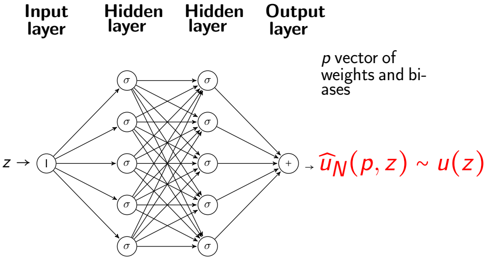
Joint work with: V. Mercier, S. Gratton
(IRIT-INP, Toulouse)

13th JLESC Workshop

14-16 December 2021

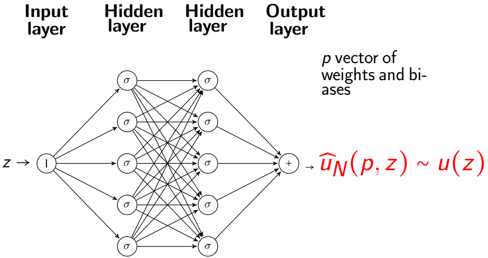
Context: solution of PDEs by neural networks

PDE: $D(z, u(z)) = f(z),\ z \in \Omega$ BC: $u(z) = g(z),\ z \in \partial\Omega$



Context: solution of PDEs by neural networks

PDE: $D(z, u(z)) = f(z),\ z \in \Omega$ BC: $u(z) = g(z),\ z \in \partial\Omega$



Idea: approximate the solution $u(z)$ of the PDE by a neural network by exploiting the physics of the problem:

Physics Informed Neural Networks (PINNs)

Physics informed neural networks (PINNs)

PDE: $D(z, u(z)) = f(z),\ z \in \Omega$ BC: $u(z) = g(z),\ z \in \partial\Omega$
PINNs training problem: find the network weights p by minimizing

$$\mathcal{L}(p) = RMSE_{res}(p) + RMSE_{data}(p)$$
$$RMSE_{res}(p) = \frac{\lambda^r}{N^r}\|D(z, \hat{u}_N(p; z^r)) - f(z^r)\|^2,$$
$$RMSE_{data}(p) = \frac{\lambda^m}{N^m}\|\hat{u}_N(p; z^m) - u(z^m)\|^2,$$

given training points $z^r \in \Omega$ and measurement points $z^m \in \Omega \cup \partial\Omega$

Physics informed neural networks (PINNs)

Advantages

- No need of discretization: we get an analytical expression of the solution, with good generalization properties (also for points outside the interval)
- Natural approach for solving nonlinear equations
- Alleviate the curse of dimensionality

- Overcoming the curse of dimensionality in the numerical approximation of semilinear parabolic partial differential equations (2018).
- Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations (2019)
- Hidden Fluid Mechanics: A Navier-Stokes Informed Deep Learning Framework for Assimilating Flow Visualization Data (2018)

Limitations

Usually trained by SGD:

- convergence may be slow
- convergence depends on the choice of the learning rate
- training is time consuming

Limitations

Usually trained by SGD:

- convergence may be slow
- convergence depends on the choice of the learning rate
- training is time consuming

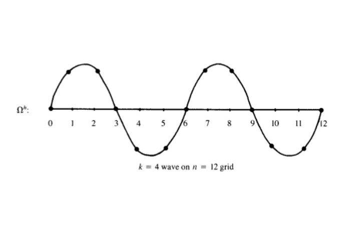
Idea: transpose acceleration methods classically used for PDEs to neural networks

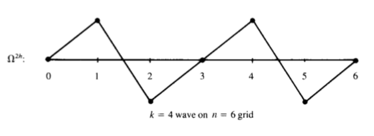
Focus on multigrid methods

Classical multigrid methods

Discretization on grid h: large-scale linear system $A_h u_h = f_h$.

- Relaxation methods fails to eliminate smooth components of the error efficiently.
- Smooth components projected on a coarser grid appear to be more oscillatory.





Multigrid methods

Ingredient 1: coarse grid

Want to solve $A_h u_h = f_h$. Exploit a coarser discretization H. Get a lower dimensional problem: $A_H u_H = f_H$.

Ingredient 2: iterative refinement

Given some approximation v to u, we define

$$\begin{aligned} e &= u - v, \\ r &= f - Av, \\ Ae &= r \ \ \text{(residual equation)} \end{aligned}$$

To improve v, we solve the residual equation and set $v = v + e$.

Classical multigrid scheme

V-cycle on two levels

- Relax ν_1 times on $A_h u_h = f_h$ to obtain an approximation v_h
- Compute the residual $r_h = f_h - Av_h$.
- Project the residual on the coarse level $r_H = Rr_h$
- Relax ν_2 times on the residual eq. $A_H e_H = r_H$ to obtain e_H
- Correct the fine level approximation $v_h = v_h + Pe_H$

Classical multigrid scheme

V-cycle on two levels

- Relax ν_1 times on $A_h u_h = f_h$ to obtain an approximation v_h
- Compute the residual $r_h = f_h - Av_h$.
- Project the residual on the coarse level $r_H = Rr_h$
- Relax ν_2 times on the residual eq. $A_H e_H = r_H$ to obtain e_H
- Correct the fine level approximation $v_h = v_h + Pe_H$

State-of-the art method for the solution of PDEs: superior to one-level relaxation methods already on two-levels

Multilevel physics informed neural networks (MPINNs)

- Two discretization levels
  MG: Two grids h, H
  MPINN: $\hat{u}_h(p_h; z_h), \hat{u}_H(p_H; z_H)$
- Fine problem
  MG: $A_h u_h = f_h$
  MPINN: $\min_{p_h} \mathcal{L}_h(p_h) = \frac{1}{N_h^r} \| D(z_h^r, \hat{u}_h(p_h; z_h^r)) - f(z_h^r) \|^2$
- Residual equation
  MG: $A_H e_H = r_H$
  MPINN $\min_{p_H} \mathcal{L}_H(p_H) = \frac{1}{N_H^r} \| D(z_H^r, \hat{u}_H(p_H; z_H^r)) - r(z_H^r) \|^2$
- Fine solution update
  MG: $v_h = v_h + Pe_H$
  MPINNs: $\hat{u}_h(p_h; z_h) = \hat{u}_h(p_h; z_h) + \mathcal{P}(\hat{u}_H(p_H; z_H))$

MPINNs training

The training in this case follows the following scheme:

- Perform ν_1 epochs on the fine problem, get $\hat{u}_h(p_h, z)$ of $u(z)$
- Compute the residual $r_h(z_h^r) = f(z_h^r) - D(z_h^r, \hat{u}_h)$
- Project the residual on the coarse level $r_H = \mathcal{R}(r_h)$
- Perform ν_2 epochs on the residual problem, get $\hat{u}_H(p_H, z)$
- Correct the fine level approximation $\hat{u}_h(p_h, z_h) + \mathcal{P}(\hat{u}_H(p_H, z_H))$.

Transfer operators

MG: linear operators

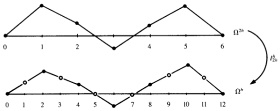


Figure 3.2: *Interpolation of a vector on coarse grid Ω^{2h} to fine grid Ω^{h}.*

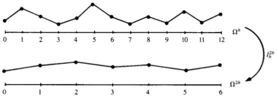


Figure 3.4: *Restriction by full weighting of a fine-grid vector to the coarse grid.*

MPINN: the variables of the optimization problem p don't possess an evident geometry: apply them to the underlying geometrical variable z, and thus we define:

$$\mathcal{R}(\hat{u}_h(p_h, z_h)) := \hat{u}_H(p_H, R_{MG} z_h)$$
$$\mathcal{P}(\hat{u}_H(p_H, z_H)) := \hat{u}_h(p_h, P_{MG} z_H)$$

Restriction is still a neural network, with less parameters and evaluated on a smaller set of grid point

Preliminary results 1D: ADAM

$$u''(z) - u(z) = f(z), z \in [-1, 1]$$

with $f(z) = -(\pi^2 + 1)\sin(\pi z) - (\alpha^2\pi^2 + 1)\sin(\alpha\pi z)$.

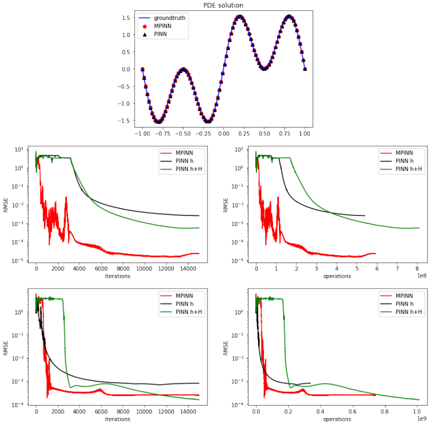


Figure: $\alpha = 3$, ADAM

Preliminary results 1D: ADAM

(h,H)	MPINN	PINN h	PINN $h+H$
(50,25)	1.3e-04, 3.1e-04	1.3e-04, 1.2e-04	7.0e-04, 4.3e-03
(200,100)	2.0e-04, 3.1e-04	1.1e-03, 2.9e-03	2.0e-03, 2.5e-03
(300,150)	1.4e-03, 5.2e-03	6.1e-03, 9.5e-1	>1

Table: $\alpha = 3$. Median and IQR for the RMSE

MPINNs are less sensible to the choice of the learning rate

Preliminary results 1D: BFGS

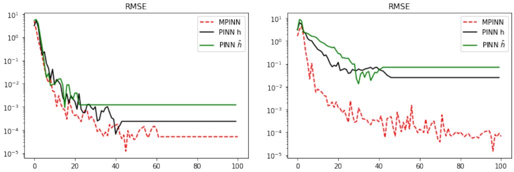
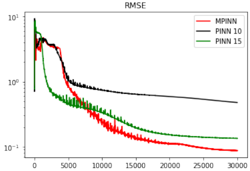


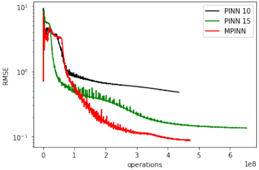
Figure: $\alpha = 7$, BFGS

α	MPINN	PINN h	PINN $\tilde{h}$
8	3.0e-3, 3.0e-3	1.5e-2, 2.2e-2	1.7e-2, 3.0e-2
10	1.0e-2, 3.1e-2	1.3e-1, 2.8e-1	4.0e-2, 1.8e-1
12	3.0e-2, 1.0e-1	1.0e-1, 3.5	1.7e-1, 1.4

Other tests

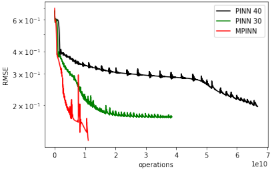
Nonlinear 2D: $-\Delta u + \alpha e^u = f$ in $\Omega = [0,1] \times [0,1]$

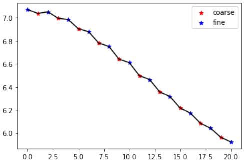




Burger's equation: $$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}.$$

	MPINN	PINN 40	PINN 30
RMSE	1.3e-1,0.4e-1	1.8e-1, 0.4e-1	1.7e-1, 1.5e-2
Operations	1	6.1	3.5





Conclusions and perspectives

- Promising preliminary results
- Need for a deeper numerical investigation (other problems, deeper V-cycles)
- Need for an efficient implementation
- Need for theoretical convergence theory

Thank you for your attention!

Preprint available soon:

Multilevel physics informed neural networks (MPINNs) E.Riccietti, V. Mercier, S. Gratton, 2021

Previous work:

On a multilevel Levenberg-Marquardt method for the training of artificial neural networks and its application to the solution of partial differential equations, H. Calandra, S. Gratton, E. Riccietti X. Vasseur, SIOPT, 2021.

Hyperparameters tuning

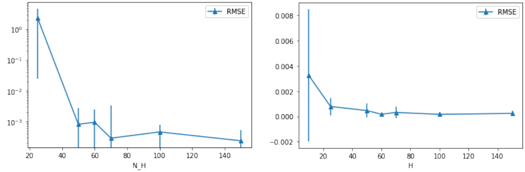


Figure: $\alpha = 3$, N_H number of training points, H number of neurons in the coarse network

N_H	25	50	60	70	100	150
RMSE	2.3	8.0e-4	9.4e-4	2.8e-4	4.5e-4	2.3e-4
Op.	0.85	0.88	0.89	0.84	0.94	1

H	10	25	50	60	70	100	150
RMSE	3.2e-3	7.8e-4	4.6e-4	1.7e-4	3.1e-4	1.6e-4	2.4e-4
Op.	0.88	0.89	0.91	0.93	0.96	0.98	1