Machine Learning per il Calcolo Scientifico

Fifth laboratory exercises

General guide

For each laboratory, an incomplete Python notebook will be provided with exercises (steps) that must be completed in the given order (some of the exercises will be needed in future laboratories). In step zero, all the Python packages that are needed in order to complete the notebook are listed. This PDF includes the text for the exercises and the expected outcomes for each step. While following the notebook is recommended, you are also welcome to attempt the exercises without using it.

Step Zero

Here are the required Python (`https://www.python.org/`) packages for this laboratory: PyTorch (`https://pytorch.org/`), Numpy (`https://numpy.org/`), Matplotlib (`https://matplotlib.org/`).

Step one: inverse problem with poisson equation

We consider the following second-order partial differential equation (PDE):

$$\begin{cases} -\alpha \frac{d^2 u}{dx^2} = f(x), \quad x \in \Omega = (0,1), \\ u(0) = u(1) = 0. \end{cases}$$

The target function $\tilde{u}$ is given by:

$$\tilde{u} = \sin(\pi x)$$

The function $f(x)$, derived from the assumed true solution, is specified as:

$$f = 2\pi^2 \sin(\pi x).$$

Our goal is to compute the value of α that best aligns the solution $u(x)$ of the PDE with the given $\tilde{u}(x)$. To achieve this, we can utilize a Physics-Informed Neural Network (PINN) that integrates the differential equation as a part of its loss function. This approach ensures that the learned solution not fits the data while respecting the underlying "physics" described by the PDE.

a. Define the PINN model with 1 neuron in input, 1 neuron in output and 4 hidden layers with 50 neurons each. Use the tanh activation function. Then define a parameter α as a tensor with requires_grad=True that will be trained in order to approximate the true value of α.

b. Define the boundary conditions take 150 random points for the train of the PINN, 60 equispaced points for the data interpolation and 100 points for the test.

c. Define the exact solution, the right-hand side of the PDE and the loss function for the PINN and the exact value of α.

d. Define the derivative of the Neural Network using torch.autograd. Then try to train the network and plot the results obtained.

Step two: Inverse problem with heat equation

Consider the one-dimensional heat equation:

$$u_t(t,x) = k(t,x)u_{xx}(t,x) + s(t,x), \quad t \in [0,T],\ x \in [-1,0]$$

with zero Dirichlet boundary conditions

$$u_b(t,-1) = u_b(t,0) = 0,$$

and initial condition

$$u(0,x) = u_0(x) = -\sin(\pi x)$$

The equation parameter $k : [0,T] \times [-1,0] \mapsto \mathbb{R}$ denotes the medium conductivity and $s : [0,T] \times [-1,0] \mapsto \mathbb{R}$ is a source term. Suppose that $s(t,x)$ is given and we want to estimate $k(t,x)$ and approximate $u(t,x)$ with two different neural networks.

We will use a physics-informed neural network (PINN) to solve this inverse problem with tunable parameters θ and ϕ:

$$u_\theta(t,x) \approx u(t,x), \quad k_\phi(t,x) \approx k(t,x).$$

a. Initialize u_θ as a neural network with 2 input, 1 output and 5 hidden layers with 40 neurons each. Initialize $k_\phi(t,x) = k_\phi(x)$ as a neural network with 1 input, 1 output and 5 hidden layers with 40 neurons each, both the networks use the hyperbolic tangent as activation function.

b. Define the initial conditions, exact solution, the exact conductivity term, the right-hand side (source term) of the PDE and the loss function for the PINN and the exact value of α.

c. Define the derivative of the Neural Network using torch.autograd. Then try to train the network and plot the results obtained.