Thermal Storage Design: Part 3

Deep Learning in
Scientific Computing
Due date: None

Testing data can be found on the Moodle page: `https://moodle-app2.let.ethz.ch/course/view.php?id=14817`

Task 6: PINNs for solving PDEs (Optional - Not Graded)

In this task we aim at solving the system of equations (1, Thermal Storage Design: Part 1) with physics informed neural networks. In particular we are interested in the solution of the system during the charging phase of the first cycle.

To this end, consider the *non-dimensional* set of equations:

$$\begin{aligned}
&\frac{\partial \bar{T}_f}{\partial t} + U_f \frac{\partial \bar{T}_f}{\partial x} = \alpha_f \frac{\partial^2 \bar{T}_f}{\partial x^2} - h_f(\bar{T}_f - \bar{T}_s) \quad x \in [0,1],\ t \in [0,1],\\
&\frac{\partial \bar{T}_s}{\partial t} = \alpha_s \frac{\partial^2 \bar{T}_s}{\partial x^2} + h_s(\bar{T}_f - \bar{T}_s) \quad x \in [0,1],\ t \in [0,1],
\end{aligned} \tag{1}$$

with the following initial and boundary conditions:

$$\begin{aligned}
&\bar{T}_f(x, t=0) = \bar{T}_s(x, t=0) = T_0, && x \in [0,1],\\
&\left.\frac{\partial \bar{T}_s}{\partial x}\right|_{x=0} = \left.\frac{\partial \bar{T}_s}{\partial x}\right|_{x=1} = \left.\frac{\partial \bar{T}_f}{\partial x}\right|_{x=1} = 0, && t \in [0,1],\\
&\bar{T}_f(x=0, t) = \frac{T_{hot} - T_0}{1 + \exp\big(-200(t - 0.25)\big)} + T_0, && t \in [0,1].
\end{aligned} \tag{2}$$

The values of the constants are:

$$\begin{aligned}
&\alpha_f = 0.05 \quad h_f = 5 \quad T_{hot} = 4 \quad U_f = 1\\
&\alpha_s = 0.08 \quad h_s = 6 \quad T_0 = 1
\end{aligned} \tag{3}$$

In the spirit of physics informed neural networks (Pinns), you can either use:

1. a two-outputs neural network $(t, x) \mapsto (\bar{T}_s^\theta, \bar{T}_s^\theta)$ with tunable parameters θ, or
2. two distinct neural networks $(t, x) \mapsto \bar{T}_f^{\theta_f}$ and $(t, x) \mapsto \bar{T}_s^{\theta_s}$ with distinct sets of tunable parameters θ_s and θ_f,

to approximate the solution of the system of PDEs (1). The python script *PinnsTutorial.py* can be easily modified to address the exercise. You can follow the steps below:

a) initialize the approximate neural network solution in the *Pinns* class;

b) implement the functions *add_collocation_points* *add_initial_points* and *add_boundary_points*;

c) implement the function *appy_initial_condition*;

d) implement the function *appy_boundary_conditions*;

e) implement the function *compute_pde_residuals*;

f) train the model.

Once the model is trained, evaluate the performance of your model by comparing the Pinns solution with the one obtained with a finite difference scheme on a very fine mesh stored in the file *Task6/exact_solution.txt*. The first column of the file corresponds to the time variable t, the second to the space variable x and the last two to $\bar{T}_f$ and $\bar{T}_s$, respectively.

Hint: in the function *appy_boundary_conditions* you need to implement Neumann boundary conditions at $x = 0$ for the solid and $x = 1$ for both the phases. The network derivative at $x = 0$ and $x = 1$ with respect to t and x can be computed as done in the *compute_pde_residuals* function.