Thermal Storage Design: Part 2

ETH zürich

Deep Learning in
Scientific Computing
Due date: June 12th, 2022

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

IMPORTANT INFORMATION

To get ECTS credits for the course Deep Learning in Scientific Computing you need to submit and obtain a passing grade on the project. The project consists of two parts and the **submission deadline is the 12th of June** for both of them. Each part accounts for several tasks where you will be asked to train a learning model and provide predictions on a testing set. Therefore, **the final submission consists of the predictions files together with a report of maximum 2000 characters** for each task where you succinctly describe the procedure followed. Those have to be collected in a zip folder named as *yourfirstname_yoursecondname_yourleginumber.zip*. **Only the students that submit the report and the prediction files will be graded.**

Task 4: Inference of Fluid Velocity

Let us assume that **noisy** measurements of the fluid temperature $\{(t_j, T^*_{f,j}),\ j = 1, ..., N\}$ are taken at the bottom end of the storage $x = L$ during the charging phase of the first cycle and stored in the file **Task4/MeasuredData.txt** (see Figure 1 for a representation of the data). Further assume, without loss of generality, that the measurements are i.i.d and drawn from a Gaussian distribution:

$$P(T^*_{f,j}|t_j, u = u^*) = N\Big(T_f(t_j; u^*), \sigma(t_j)\Big),$$

where $T_f(t_j; u^*)$ is solution of system (1, Thermal Storage Design: Part 1) at time frames $\{t_j, j = 1, ..., N\}$ obtained with a fluid velocity $u_f = u^*$, and σ being standard deviation of the Gaussian distribution.

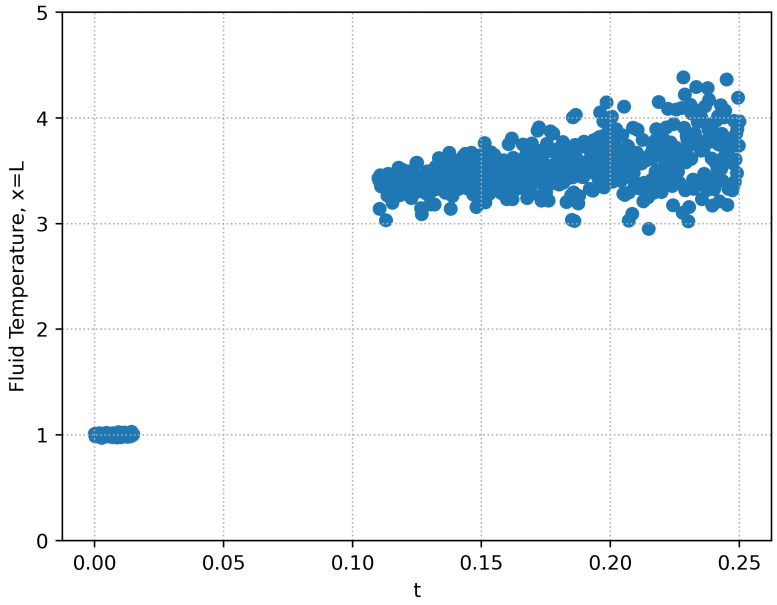

Figure 1: Task 4 measured data

To solve the task you are also provided with a training set $S = \{s_i,\ i = 1, ..., 1024\}$, $s_i = (t_j, u_k, T_{f,jk})$, $j = 1, ..., 128$, $k = 1, ..., 8$, generated by solving with a finite difference scheme the system of equations (1, *Thermal Storage Design: Part 1*) for 8 values of the velocity u_k, $k = 1, ..., 8$ and collecting 128 tuples $(t_j, T_{f,jk})$, $j = 1, ..., 128$, at $x = L$. The training data can be found in the file **Task4/TrainingData.txt**.

4a. As shown in Figure 1, the measurements' noise is a function of time t, $\sigma = \sigma(t)$. Make use of the *measured data* to infer the map $\sigma = \sigma(t)$ and provide predictions of the standard deviation on the data stored in the file **Task4/TestingDataA.txt**. These have to be saved in the folder *yourfirstname_yoursecondname_yourleginumber* as *Task4a.txt*. **The format of the file has to be the same as the file Task4/SubExampleA.txt.**

4b. The objective of Task 4b is to infer the reference velocity of the fluid u^* generating the recorded measurements. This can be done by minimizing a suitable cost function $G = G(t_{1:N}, T^*_{f,1:N}, u)$,

$$u^* = \arg\min_u G(t_{1:N}, T^*_{f,1:N}, u), \tag{1}$$

where $1:N$ is a short notation to denote the entire set of measurements. Design a procedure and a suitable cost function G that, given the provided training set S and the measured data, allow you to infer the target velocity u^*. Register the obtained value in a file called *yourfirstname_yoursecondname_yourleginumber/Task4b.txt*. **The format of the file has to be the same as the file Task4a/SubExampleB.txt.**

4c. Perform Bayesian inference and provide 10000 samples drawn from the posterior distribution

$$P(u|t_{1:N}, T^*_{f,1:N}).$$

In order to do so, use the following prior and likelihood:

$$P(u) = N(12, 4), \quad P(T^*_{f,j}|t_j, u) = N\Big(f(t_j, u), 0.075\Big),$$

with f being a suitable model for the distribution's mean, dependent on t and u. Save the posterior samples in the file *yourfirstname_yoursecondname_yourleginumber/Task4c.txt*. **The format of the file has to be the same as the file Task4/SubExampleC.txt.**

Task 5: Design of Storage Geometry

In this task we are interested in the optimal design of the thermal storage. Formally, the aim of the task is to find the control parameters $y = (D, v)$, with $D \in [2, 20]$ being the diameter of the storage and $v \in [50, 400]$ its volume, such that the capacity factor is exactly $CF_{ref} = 0.45$. This can be done by solving the following minimization problem:

$$(D^*, v^*) = \arg\min_{D,v} G\big(CF(D, v)\big) \tag{2}$$

with $G\big(CF(D, v)\big)$ being a suitable *cost function*.

However, the resulting optimal control parameter (D^*, v^*) is not unique. Specifically, the minimizers of the cost function $G\big(CF(D, v)\big)$ correspond to points lying on a curve γ in the control parameter space $[2, 20] \times [50, 400]$ that can be defined by a function $v^* = v^*(D^*)$. The objective of this task is to find the optimal curve, namely to provide $N = 1000$ different minimizers (D^*_k, v^*_k), $k = 1, ..., N$ of the cost function $G\big(CF(D, v)\big)$.

To solve the task you are provided with a training set $S = \{(D_i, v_i, CF_i),\ i = 1, ..., 50\}$ stored in the file **Task5/TrainingData.txt**. Register $N = 1000$ minimizers of the cost function in a file called
yourfirstname_yoursecondname_yourleginumber/Task5.txt. **The format of the file has to be the same as the file Task5/SubExample.txt.**

Task 6: PINNs for solving PDEs

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:

$$\begin{aligned} \frac{\partial \bar{T}_f}{\partial t} + U_f \frac{\partial \bar{T}_f}{\partial x} &= \alpha_f \frac{\partial^2 \bar{T}_f}{\partial x^2} - h_f(\bar{T}_f - \bar{T}_s) \quad x \in [0,1],\ t \in [0,1], \\ \frac{\partial \bar{T}_s}{\partial t} &= \alpha_s \frac{\partial^2 \bar{T}_s}{\partial x^2} + h_s(\bar{T}_f - \bar{T}_s) \quad x \in [0,1],\ t \in [0,1], \end{aligned} \tag{3}$$

with the following initial and boundary conditions:

$$\begin{aligned} &\bar{T}_f(x, t=0) = \bar{T}_s(x, t=0) = T_0, && x \in [0,1], \\ &\frac{\partial \bar{T}_s}{\partial x}\Big|_{x=0} = \frac{\partial \bar{T}_s}{\partial x}\Big|_{x=1} = \frac{\partial \bar{T}_f}{\partial x}\Big|_{x=1} = 0, && t \in [0,1], \\ &\bar{T}_f(x=0, t) = \frac{T_{hot} - T_0}{1 + \exp\big(-200(t-0.25)\big)} + T_0, && t \in [0,1]. \end{aligned} \tag{4}$$

The values of the constants are:

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

Approximate the solution of the system of PDEs (3) with a physics informed neural network (Pinns). To this end, 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.

The python script *PinnsTutorial.py* can be easily modified to address the task. You can follow the steps below:

1. initialize the approximate neural network solution in the *Pinns* class;
2. implement the functions *add_collocation_points* *add_initial_points* and *add_boundary_points*;
3. implement the function *appy_initial_condition*;
4. implement the function *appy_boundary_conditions*;
5. implement the function *compute_pde_residuals*;
6. train the model.

Once the model is trained, predictions of your model on the data stored in *Task6/TestingData.txt* and save them in *yourfirstname_yoursecondname_yourleginumber/Task6.txt*. **The format of the file has to be the same as the file Task6/SubExample.txt.**

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.