Deep Learning in Scientific Computing - Project Report

Jonas Grütter

October 19, 2022

The whole project has been realized using Pytorch and the code is mainly inspired by the notebooks presented during the tutorials.

1 Task 1: Noisy Function Approximation

Data are manipulated first. For the temperature of the solid, at the peaks, data too far from the mean of the peaks are removed and a value at $t = 0$ is added by linearly interpolating the first 4 values. Concerning the temperature to the fluid, it is set at the peaks to a constant value (mean of the data on the peak) as the temperature of the entering fluid should be constant. Moreover, when the points form clusters, points too far from the mean are removed. Then, in order to approximate the maps $T_s^0 : t \mapsto T_s(0,t) \quad T_f^0 : t \mapsto T_f(0,t)$, two different neural networks (one for each temperature) are used. Both of them are feed forward neural network. Hyper-parameters of these networks were found by grid-search and k-fold cross validation. Predictions of model performing well were averaged in order to reduce the learned noise.

2 Task 2: Learning Observables in High Dimensional Space

To approximate the map $CF : y \mapsto CF(y)$ given the 8 input variables, Multi-level training is performed using the available solutions at different mesh resolutions and a feed forward neural network with regularization. The notebook presented in the tutorial has been adapted to this problem. It is important to note that the inputs used during the training are the Sobol points.

3 Task 3: Time Series Forecasting

In order to approximate the maps $T_s^0 : t \mapsto T_s(0,t) \quad T_f^0 : t \mapsto T_f(0,t)$ in the future, the data are first transformed such that the problem becomes a supervised learning problem: the last $n \in \mathbb{N}$ steps are associated with the current step for all steps. Then the fitting is done using an univariate LSTM Network. As in task 1, two neural networks are used (1 for each temperature).

4 Task 4: Inference of Fluid Velocity

(a) The parameters of the probability distribution of the observed data is found by maximizing a polynomial likelihood function so that the observed data is most probable. This is achieved by minimizing the norm between a polynomial and the given sequence by altering the weights of the polynomial. This allowed me to find the map $\sigma = \sigma(t)$.

(b) First the map $T_f : (t,u) \mapsto T_f(t,u)$ is learned by a feed-forward neural network (on the training data). Than the velocity is found by minimizing the cost function $u^* = \arg\min_u G\left(t_{1:N}, T^*_{f,1:N}, u\right)$ where $G\left(t_{1:N}, T^*_{f,1:N}, u\right)$ is the mean squared error between the approximated fluid temperature and the recorded measurements. The minimization is done using the method in the tutorial about Optimal Design using a LBFGS optimizer with a very small learning rate and 100 different initial velocities.

(c) The tutorial about Bayesian inference was modified in order to solve this task. First, the measured data were replaced by the one provided. Second, the parameters of the prior and likelihood distributions have also been modified by those given. The mean of the likelihood was approximated by a neural network mapping the time and velocity onto the fluid temperature (as in task 4 b)) trained on the training data.

5 Task 5: Design of Storage Geometry

First the map $CF : (D,v) \mapsto C_F$ is learned by a feed-forward neural network CF_θ using the training data. Then, as suggested, the 1000 control parameters $y_k = (D_k, v_k)$, $k = 1,...,1000$ such that the capacity factor is exactly $CF_{ref} = 0.45$ were found by solving the following minimization problem: $(D^*, v^*) = \arg\min_{D,v} G(CF(D,v))$. Where $G(CF(D,v))$ is the mean squared error between $CF_{ref} = 0.45$ and the approximated capacity factor (by the neural network). In order to do the minimization, the method presented in the tutorial about Optimal Design is used: Pairs of random volumes and diameters (D,v) are generated until 1000 of these pairs satisfy $CF_\theta(D,v) = 0.45$ up to machine precision and $(D,v) \in [2,20] \times [50,400]$.

6 Task 6: PINNs for solving PDEs

To solve the considered equation system with physics informed neural networks, a two-outputs neural network $(t,x) \mapsto \left(\bar{T}_s^\theta, \bar{T}_s^\theta\right)$ with tunable parameters θ is used. As suggested, the PinnsTutorial.py script has been modified to achieve this: The architecture of the neural network, the functions applying the boundary and initial conditions, the computation of the residuals and the loss have been adapted to the studied problem.