EXERCISE DAY 3
CPDE SUMMER SCHOOL:
A PRACTICAL INTRODUCTION TO CONTROL, NUMERICS AND MACHINE LEARNING

DANIËL VELDMAN

Recall from the lecture that the training of a deep (residual) neural network can be viewed as an optimal control problem in which the cost function

$$J(\mathbf{V},\mathbf{b}) = \frac{1}{2}\sum_{i=1}^{I} |\mathbf{x}^i(T) - \mathbf{y}^i_{\text{out}}|^2 + \frac{w_1}{2}\sum_{i=1}^{I}\int_0^T |\mathbf{x}^i(t) - \mathbf{y}^i_{\text{out}}|^2 \,\mathrm{d}t + \frac{w_2}{2}\int_0^T \left(\|\mathbf{V}(t)\|_F^2 + |\mathbf{b}(t)|^2\right)\mathrm{d}t, \tag{1}$$

should be minimized subject to the dynamics (for $i = 1, 2, 3, \dots, I$)

$$\mathbf{x}^i(0) = \mathbf{x}^i_{\text{in}}, \qquad \dot{\mathbf{x}}^i(t) = \mathbf{V}(t)\sigma(\mathbf{x}^i(t) + \mathbf{b}(t)), \tag{2}$$

with $\mathbf{x}^i_{\text{in}} \in \mathbb{R}^N$. In this exercise we will explore the effectivity of the six different gradient-based algorithms discussed in the lecture.

Note: all files for this exercise work both in Matlab and Octave.

a. Implement the gradient descent algorithm with a fixed step size (learning rate) $\beta = 0.1$ by completing the missing lines in `CPDESS_Exercise3`.
Note that the function `NN_compute_gradients` is defined different as in the exercise for Day 2. It now takes both the batch size `batch_size` and the total size of of the data set I as inputs. Before the code will work, you also need to complete the last two lines in `NN_compute_gradients` such that the batch size is used correctly. Note that for the deterministic algorithm considered in part a., `batch_size` $= I$.
b. Implement the Stochastic Gradient Descent (SGD) algorithm (with batch size 1). Note that you can simply call `NN_compute_state` with $I = 1$ and the part of the initial condition

$$\mathbf{X}(0) = \begin{bmatrix} \mathbf{x}^1_{\text{in}} \\ \mathbf{x}^2_{\text{in}} \\ \vdots \\ \mathbf{x}^I_{\text{in}} \end{bmatrix}. \tag{3}$$

corresponding to the selected data sample. For the computation of the adjoint state, you can call `NN_compute_adjoint` with $I = 1$ and the part of the final condition

$$\mathbf{Y}_{\text{out}} = \begin{bmatrix} \mathbf{y}^1_{\text{out}} \\ \mathbf{y}^2_{\text{out}} \\ \vdots \\ \mathbf{y}^I_{\text{out}} \end{bmatrix}, \tag{4}$$

corresponding to the selected data sample.

c. Implement the Stochastic Gradient Descent with mini batch size 4 based on the ideas from part b.
d. Implement the momentum stochastic gradient descent (with batch size 1).
e. Implement the ADAM algorithm for stochastic gradient descent (again with batch size 1).
f. Compare the quality of the results and the required computational times for 100 epochs of training with the 6 considered algorithms. What do you observe?