Hamburg University of Technology
Institute of Mathematics
Chair Numerical Mathematics

Jens-Peter M. Zemke
Finn Sommer
Lars Stietz

Advanced Machine Learning
Winter Semester 2025/2026
Exercise Sheet 2
Week 3, 27.10.2025 – 31.10.2025

Homework:

Exercise 1: (2 points) level •○○ relevance •○○
Check the activation function $\mathsf{SoftAbs}_k(x)$ for monotonicity, symmetry, asymptotic behavior[1], and calculate the derivative.

Exercise 2: (2+1 points) level •○○ relevance ••○
Use the skeleton `exercise2_skel.py` to set up a feedforward neural network in TensorFlow with one hidden layer for the so-called MNIST dataset.

a) What test accuracy do you reach with
   - 100 neurons in the hidden layer,
   - the ReLU activation function in the hidden layer,
   - the identity (linear) activation function in the output layer?

b) What happens if you choose different activation functions in the hidden and/or output layers and/or a different number of neurons in the hidden layer?

Hint: You only need to use the `get_model` function to define the network.

Exercises (in class):

Exercise 3: (2+2 points) level ••○ relevance •••
Implement a simple feedforward network. Complete the skeleton `feedforward_skel.py`, which already implements the initialization of the weights and biases as layer-wise list.

a) Complete the `__call__` method of the class `FeedforwardNet`. This method should perform the feedforward evaluation of the network.
b) Test your implementation with your results from exercise 1 of exercise sheet 1. You can use the already implemented methods `set_weights` and `set_bias` to set the weights and the bias of a layer.

Exercise 4: (1+2+1+2 points) level •○○ relevance ••○
Use the skeleton `exercise4_skel.py` to test your implementation of a feedforward neural network with the weights and biases supplied in `MNIST_params.npz`.

a) Create a network with the same structure as in exercise 2a), i.e., one hidden layer with 100 neurons and an output layer with 10 neurons.

[1] See the lecture notes, pp. 33–34

b) Use the function `load_parameters` to load the weights and the biases from the file `MNIST_params.npz` and the methods `set_weights` and `set_bias` of `FeedforwardNet` to set weights and biases in the network to the loaded ones.
c) Evaluate the accuracy of the network on the test data of the MNIST dataset.
d) What are visible differences between our NumPy implementation and the TF/Keras implementation of a feedforward neural network?