# AI1072: Machine learning, exercise sheet 4

## 1 Preparations (on paper)

Compute the following derivatives!
**a)** $\frac{\partial}{\partial x_3} \sum_{i=1}^{5} x_i$
**b)** $\frac{\partial}{\partial x_6} \sum_{i=1}^{5} x_i$
**c)** $\frac{\partial}{\partial x_3} \sum_{i=1}^{5} (2x_i - 5)^2$
**d)** $\frac{\partial}{\partial x_3} \sum_{i=1}^{5} (x_i^2 - 5x_i)^2$
**e)** $\frac{\partial}{\partial x_3} \cos(x_3)x_3$
**f)** $\frac{\partial}{\partial x_3} \cos(x_3)x_4$
**g)** $\frac{\partial}{\partial x_3} \cos(\exp(x_3))$

## 2 Gradient descent (on paper)

Consider the function $f(\vec{x}) = x_1^2 + 2x_2^2$. Assuming a starting point of $\vec{x}_0 = [1, 3]$ and a step size of $\epsilon = 0.1$: perform 4 steps of gradient descent, that is, compute the values of $x_1$, $x_2$, $x_3$, $x_4$, and $x_5$ by iteration.

## 3 Gradient descent (on the computer)

Perform $N$ steps of gradient descent for $f$ in Python, with starting point and steps sizes that are constants and which can be changed easily. Hint: implement the function f and its gradient as functions that take a single 1D numpy array as argument. What do they return?

## 4 Covariance matrices and PCA

Compute the covariance matrix for the first 500 samples of the MNIST training data (same as last week, using broadcasting). Use *numpy.eigh* for computing the eigenvectors and visualize the first and the last 5 of these! What do you observe?

## 5 PCA and compression

*numpy.eigh* returns the eigenvectors as a matrix of which the eigenvectors are columns. Implement data compression in the following fashion:

- Transform the first 500 MNIST train samples using dot product with this matrix. Consider the shapes in order to determine in what order the dot product needs to be taken.

- set all but the last 5 elements of each transformed vector to 0.
- re-transform the resulting matrix using the transpose of the eigenvector matrix
- visualize the first 3 (row) samples in the resulting array. What do you observe?