Cognitive Algorithms Assignment 1

Part 2 - Python Preliminaries

Due on Tuesday, May 2, 2017 10 am via ISIS

The aim of this task is to familiarize yourself with Python. Download the python template `python_prep.py` from the ISIS web site. This template contains several function and function stubs. Do not rename the file or the functions and only insert code where indicated. The template also contains a function to help you test your code: `test_prep()`. Please make sure that all test cases are passed.
After completing Task 1 and Task 2, hand in your completed `python_prep.py` via ISIS. Please write your name and your Matrikel Number as the first line of the code. Also hand in a pdf file that contains your code, the plot generate in Task 1.4 and the Answer to Task 2.2 .

Installing Python

As Python does not come with a pre-bundled set of modules for scientific computing, you will need to install several buildings blocks:

- **Python 2.7** – `http://www.python.org/getit/`
- **iPython** – a convenient interactive shell (pretty much like Matlab) `http://ipython.org/`
- **Numpy** – provides powerful numerical arrays objects `http://www.numpy.org/`
- **Scipy** –high-level data processing routines. Optimization, regression, interpolation, etc `http://www.scipy.org/`
- **Matplotlib** – for visualization `http://matplotlib.org/`

You can find a good and detailed tutorial on scientific programming on `http://scipy-lectures.github.io/`. Material of our Python course can be found on `https://wiki.ml.tu-berlin.de/wiki/Main/SS15_PythonKurs` and the linked ISIS2 course.

Task 1

1. **(2 points)** Write the function `generate_data(N)`, which generates a $2 \times N$ array X with N data points from a 2-d gaussian distribution with mean $(1, 2)^T$ and a covariance matrix of your choice.
2. **(2 points)** Write the function `scale_data(X)` which scales the two-dimensional data in X, the x-direction by 2 and the y-direction by 0.5.
3. **(4 points)** Write the function `standardise_data(X)` which standardises data X of arbitrary dimension. The function should transform each row $X[i, :]$ of X such that its mean

$$\mu[i] = \frac{1}{N} \sum_{n=0}^{N-1} (X[i, n])$$

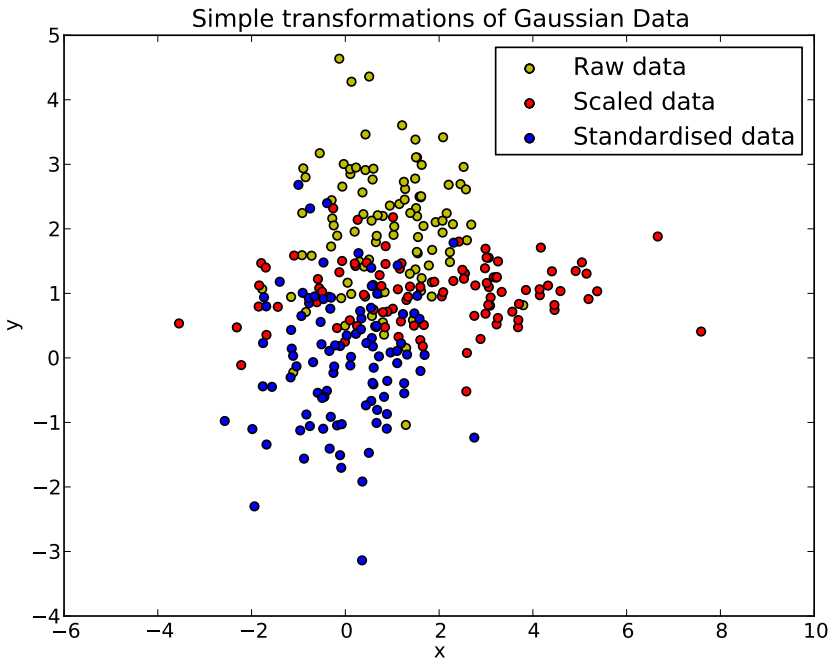

Figure 1: Example plot of Task 1

is zero and its standard deviation σ

$$\sigma[i] = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1}(X[i,n] - \mu[i])^2}$$

is one. Try to implement this function without using loops. (For the use of loops one point will be deducted.)

4. **(3 points)** Complete the function `task1()` that generates the plot in Figure 1. Plot the original data, the scaled data, and the standardized data on top of each other with different colors.

Note that Python is, in general, relatively slow for loops (see Task 2 and, e.g. `http://stackoverflow.com/questions/8097408/why-python-is-so-slow-for-a-simple-loop`). Throughout this course, we will therefore deduct points for the use of unnecessary loops.

Task 2

1. **(3 points)** Implement a for-loop based calculation of the mean μ in the function stub `mean_for(X)` .

2. **(1 point)** Call the function `task2()` which compares the performance of `mean_for` and `scipy.mean`. Which function is faster?