HUMBOLDT-UNIVERSITÄT ZU BERLIN

BERNSTEIN CENTRE FOR COMPUTATIONAL NEUROSCIENCE

Humboldt-Universität zu Berlin
Philippstr. 13 House 6

Phone: 030/2093-9110
Fax: 030/2093-6771
webpage: http://www.bccn-berlin.de/

Models of Neural Systems I, WS 2009/10
Computer Practical 2
Solutions to hand in on: November, 2nd, 2009

1. **List Comprehensions**

   Generate following sequences:

   - odd numbers from 1 to 11,
   - letters from "a" to "k",
   - 10 first powers of 2 (i.e. 2, 4, 8...),

2. **Python modules**

   Python functions are usually organized into modules, which eases a lot the managing of projects. Here you will learn how to write and use simple modules.

   (a) Write your Fibonacci function into a file `mymodule.py` (you can also choose another name).

   (b) Import `mymodule` module into the interactive Python shell and run the functions:

   ```
   import mymodule
   mymodule.fib(10)
   ```

   where `fib` is the name of your function.

   (c) Try alternative forms of `import`:

   ```
   import mymodule as my
   from mymodule import fib
   from mymodule import *
   ```

   Comment on the differences between the three cases.

   (d) It is also possible to write a `main` function which will be called when the module is run as a script (for example like that: `python mymodule.py`). The simplest function looks like that:

```
...
def main():
    fib_series = mymodule.fib(10)
    print "Fibonacci series:", fib_series
if __name__ == '__main__':
    main()
```

3. **Numerical Python** (NumPy)
   Standard version of Python does not provide any functions for efficient numerical computation. However the `scipy` module extends its functionality into this area. It includes optimized algorithms for creation and manipulation of numerical arrays, basic linear algebra operations and others.

   (a) Import `numpy` (for example, `import numpy as np`)

   (b) Create the following arrays: integers from 0 to 99 (`np.arange`), the sequence $\{5.1, 5.2, 5.3, ..., 15.0\}$ (`np.arange`), 5x10 matrix of zeros/ones (`np.zeros`/`np.ones`), 5x5 identity matrix (`np.eye`)

   (c) Define two 2D arrays:

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 0 & 2 & 1 \\ 3 & 0 & 2 \\ 1 & 3 & 0 \end{pmatrix}$$

   Multiply the array A by a scalar, mulitply arrays A and B element-wise, multiply **matrices** A and B (`np.dot`).

   (d) Multiply element-wise column 1 of matrix A with row 2 of matrix B using slice notation.

4. **Basic plotting** (Pylab)

   Pylab (aka matplotlib) is a Python plotting library. Its capabilities and interface are quite similar to the ones offered by Matlab. It can be used to display the results of your calculations, create publication quality figures and can be embedded in graphical user interfaces.

   A simple plot in pylab can be drawn with the following example (taken from matplotlib tutorial):

```
import pylab as plt
plt.figure()
plt.plot([1,2,3,4])
plt.show()
```

   (a) Modify the above code, so that only points are drawn (without connecting lines).

   (b) Label x-axis, y-axis and add the title to the figure (`plt.xlabel`, `plt.ylabel`, `plt.title`).

(c) Split the figure vertically into two panels (use `plt.subplot` command). In the upper panel plot the data with lines and in the lower panel with bars (`plt.bar`).

5. **Activation functions**

   Plot the following functions in one plot. Add the axes labels and a legend to the plot. Give an interpretation for the parameters of the functions.

   (a) Sigmoid function:

   $$f(x) = \frac{1}{1 + \exp(-ax)},$$

   where a is a free parameter.

   (b) Hyperbolic tangent function:

   $$g(x) = \tanh(x).$$

   (c) Threshold function:

   $$h(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0. \end{cases}$$

   (d) Piecewise linear:

   $$i(x) = \begin{cases} 1 & \text{if } x \geq \frac{1}{2} \\ x & \text{if } \frac{1}{2} > x > -\frac{1}{2} \\ 0 & \text{if } x \leq -\frac{1}{2}. \end{cases}$$

6. **Report**

   Generate a single PDF file including your name, code, output of the calculations, comments and plots using `pyreport`. State precisely with each answer which exercise it is related to (you can also copy the text of the exercise at the top of your solution). Attach the code of your own modules imported in the file as a separate PDF document. **Send the report to Bartosz or Robert till Monday, Nov 2nd, 9am!**

Contact

Richard Kempter	Phone: 2093-8925	Email: r.kempter(at)biologie.hu-berlin.de
Robert Schmidt	Phone: 2093-8926	Email: r.schmidt@biologie.hu-berlin.de
Bartosz Telenczuk	Phone: 2093-8838	Email: b.telenczuk@biologie.hu-berlin.de