Models of Neural Systems I, WS 2008/09
Computer Practical 2
Discussed on 27 Oct 2008

IPython

In order to make programming in Python more efficient we recommend you to use `ipython`. IPython is an extended interactive shell (a replacement for `python`) which supports advanced line editting (history, tab completion) and special commands (magic commands). The list of the most commonly used magic commands is below:

`%magic` – a full list of magic functions
`%hist` – show input history
`%run` – run python script inside IPython
`%edit` – open an external editor (default `vim`)
`%pdb` – run a debugger when an error occurs
`cmd?` – print a documentation of a `cmd`
`cmd??` – print a documentation of a `cmd` together with the source code

Exercises

1. **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 *
```

Do you know what the differences are?

(d) (*Optional*) 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()
```

2. **Scientific Python** (SciPy)

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 manipulation of numerical arrays, linear algebra, solving ordinary differential equations and many others.

(a) Import `scipy` into your interactive shell: `from scipy import *`

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

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

(e) (*Optional*) Solve the following system of linear equations:

$$x + 2y = 1$$
$$3x + 4y = 0$$

Hint: use `linalg.solve`

3. **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):

```
from pylab import *
figure()
plot([1,2,3,4])
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 (`xlabel`, `ylabel`, `title`).

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

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

CONTACT

ROBERT SCHMIDT (ITB, R. 2316)   PHONE: 2093-8926   EMAIL: R.SCHMIDT@BIOLOGIE.HU-BERLIN.DE

BARTOSZ TELENCZUK (ITB, R. 1309) PHONE: 2093-8838 EMAIL: B.TELENCZUK@BIOLOGIE.HU-BERLIN.DE