Series 1

Genomics and bioinformatics - Weeks 1 and 2 - September 20, 2012

This first series will introduce you with data visualization, and its treatment using computer programming. Solutions will be provided the next week.

Solutions to programming exercises are scripts showing *one* way to solve them, accompanied with many comments and advice. Since the best way to learn programming is to imitate examples, we strongly advise to look at them carefully.

1 Website of this course

Add the course website to your bookmarks:
`http://moodle.epfl.ch/course/view.php?id=11181`

2 Demo of the UCSC genome browser

Go to the address `http://genome.ucsc.edu/`. Click on "Genome browser", select a species and visualize its genomic content; zoom in and out.

3 Programming exercise

To complete this (and next weeks') exercise, you must have R and Python installed on your computer (see instructions below).

This simple exercise, which consists in playing with a sample text file, should be completed in both languages. You must write your scripts successively in R and Python. Use as much of the documentation given below as needed to obtain the result we desire here.

Testing file: download `genes_expression_100.txt` on the course Moodle website. It is an output file of a typical bioinformatics program in raw text tab-delimited format - 100 lines of the form:

`Gene name <tab> Expression in condition 1 <tab> Expression in condition 2`

Using Python, then R, do:

1. Read the file and extract gene names and associated numbers;
2. Compute the ratios between the two numerical columns; take log_2 of the result;
3. Compute the geometric means between the two numerical columns; take log_{10} of the result;
4. Write these results in a new text file: `"GeneName <tab>` log_2`(ratio) <tab>` log_{10}`(mean)"`
5. Plot ratios vs means (use `matplotlib` for Python).
6. Find an interpretation of the graph you produced.

4 About programming

4.1 A suitable text editor

Programming is about writing scripts, which are plain text, i.e. without any specific style, as opposed to Word documents, for instance. Thus you need an appropriate text editor. We recommend the following free software:

- For Windows users: *Notepad++* (≠ *Notepad*): `http://notepad-plus-plus.org/`
- For Mac users: *TextWrangler*, *Emacs**, *MacVim**
- For Linux users: *Emacs**, *Vim**

* need some learning of their more efficient but maybe non-intuitive usage.

5 R

Cran R is a programming language *dedicated to statistical analysis*, widely used amongst scientists because it is free (GNU license), provides a large variety of efficient statistical libraries, and can produce nice graphics without too much effort. Do not try to use it for any other purpose, though: other programming languages would make it easier.

5.1 Installation

Download the latest version of R at `http://stat.ethz.ch/CRAN/`

During the installation, when you are asked what to install, don't forget to include the help files.

For Windows users: during the installation, choose where you want to install your folder; then you can run it clicking on the `.exe` file located in the `bin/` sub-folder (make a shortcut).

5.2 If you need help

1. A sample R tutorial: `http://cran.r-project.org/doc/manuals/R-intro.pdf`
2. The `?`, `help()` and `example()` commands give you informations about other commands.
3. Use Google for fast access to the reference (type "cran R <command>" to ensure it is about the right 'R'...).
4. Ask you assistants.
5. Whole reference documentation: `http://cran.r-project.org/doc/manuals/refman.pdf`

5.3 Some useful features for this exercise

- `read.table, write.table, c, plot, data.frame, log2, log10, colnames, rownames`.
- How to select rows and columns:
  `data[,1], data$''col1'', data[''col1'']; data[1,]; data[1,1]; data[2:5,3:4]`.
- Data types: data frames, vectors.

6 Python

Python is a very general programming language like C++ or Java. Compared to R, one could say it is a "lower level" language: you can do *everything* with it, but you start from basics and have to build your functions yourself, in more details. Fortunately, often someone did the job for you ("libraries").

6.1 Installing

Download the Enthought distribution of Python from our USB keys. The packages are also available on `UPDATE_URL` (but it is slow to download). I embeds in particular:

- The *Python* interpreter;
- The *IPython* improved interface;
- The *numpy* and *scipy* libraries for scientific calculus;
- The *Biopython* libraries for biological analysis;
- The *matplotlib* library for graphics.

Open a console and type "ipython" (to get a console on Windows, open the "Start" menu and type "cmd" in the search field. On a Mac, use Â®Terminal from your Applications folder).

6.2 If you need help

1. A sample Python tutorial: `http://docs.python.org/tutorial/`
   (read chapters 3.1 to 4.6, 5.5 and 7.2).
2. Use Google for fast access to the reference.
3. Ask you assistants.
4. Whole reference documentation: `http://docs.python.org/library/index.html`

6.3 Some useful features for this exercise

- Print a variable: `print`.
  Run a script (do not copy & paste): `execfile`.
- Files: `open, close, read, readline, readlines, write`.
  Strings: `split`.
  A useful pattern:

  ```
  for line in file:
       <do something with> line.split()  # Comment: 'line' is a string.
  ```

- Data types: `file, int, float, str, list`,
  and conversions: `int('4'), float('4.0'), str(4)`.
- Loops and conditions: `for, if`.
- Libraries and their specific functions:

  ```
  from math import sqrt, log
  from matplotlib.pyplot import plot, show
  ```