Python for cognitive scientists

Time-stamp: <2012-08-27 21:45 christophe@pallier.org>

Objectives

To get you acquainted with several Python libraries that will be useful in your everyday tasks as a cognitive scientist (analyse corpora, run experiments, display data, ...).

Methodology

This will be hands-on lectures: At the beginning, I will briefly present an overview of the libraries, as well as some examples of use. Then, there will be one or several programming assignments for the whole class (a step by step strategy to solve the problem will be provided. Sometimes, I will also interrupt the students to discuss relevant concepts)

Prerequisites

I assume knowledge of core Python concepts (constants, variables, lists, dictionaries, control flow (conditionals, loops), functions, importing momdules) as explained, say, in the first ten chapters of Invent_Your_Own_Computer_Games_with_Python.

I expect you should know how to:

- use the python interpreter (suggestion: use ipython)
- use an editor to load, write, save Python programs (suggestion: use IDLE if you like a minimalist approch, spyder if you prefer a comprehensive development environment).
- start a python program from the command line (minimal knowledge of command line under either Windows or Unix (cd; mkdir;...)) .

Ideally:

- knowledge of a (distributed) version control system such as Git or Mercurial
- basic knowledge of objects in python

Lecture 1: Introduction to Pygame

Anatomy of a simple pygame program. Basic display functions:c - display geometric shapes (lines, rectangles, circles, ...) - display images - record key press - displaying text

Activities:

1. Program a simple visual detection task
2. Program a diaporama

Lecture 2: Pygame II

Sprites (moving objects), Sound

Activities:

1. Program a Pong program (one paddle)
2. Program an auditory detection task
3. Program an auditory comparison task

Lecture 3: Importing, manipulating and displaying data: Matplotlib, numpy and Pandas

Intro to matplotlib and pandas. Demo of ipython notebook.

Activities:

1. read csv files containing data from different subjects and experimental conditions and create various relevant plots.
2. with numy, perform a permutation test to compare the means of two groups

Lecture 4: python for computational task: numpy

Intro to numpy (See Python4Science)

Activities:

1. Synthesize sounds with numpy
2. Compute spectrograms of sounds
3. Create povel & Essen (1985) stimuli
4. Play the stimuli with pygame, and record the (multiple) key press.

Advanced demo of an oscilloscope in Python

Lecture 5: Text processing in python

Intro to text processing

Activities:

1. Processing text files. Using regular expressions to search and replace.
2. Computing the frequency of words in a text, demonstrate Zipf's law
3. Computing the frequency of letters, diphones, ...

Advanced demos with NLTK