Nitime and IPython: tools for time-series analysis and high-level parallel computing

In this talk we will discuss two different projects, directed at different aspects of data analysis in neuroscience. We will begin with a description of Nitime, a library for the analysis of data from neuroscience experiments. Nitime includes tools for the representation, analysis and visualization of time-series and related quantities. It contains a core of numerical algorithms for time-series analysis both in the time and spectral domains, a set of container objects to represent time-series, and auxiliary objects that expose a high level interface to the numerical machinery and make common analysis tasks easy to express with compact and semantically clear code. The nitime classes can represent time and time-series on several different orders of magnitude (suitable for data from different experimental modalities), as well as events and epochs. In addition, the library contains implementations of algorithms for spectral analysis, event-related analysis and bi-/multi-variate analysis of time-series data. The library contains a high-level, user-friendly API for the scripting of common analysis tasks. This API emphasizes efficiency, in that computations are delayed until they are needed and once they are performed they are cached for further use. Finally, the library contains several visualization functions, based on functionality from Matplotlib and NetworkX.

The talk will focus on the distinction between uni-variate analysis methods, which apply to the characteristics of individual time-series, and multi-/bi-variate methods which apply to the interaction and mutual influence of two or more time-series. In order to demonstrate this distinction, we will present multi-variate analysis of data from functional Magnetic Resonance Imaging (fMRI) experiments. We will contrast and compare three methods for the calculation of functional connectivity between voxels in fMRI experiments: correlation, spectral coherency and multi-variate autoregressive analysis ("Granger causality").

The second part of the presentation will focus on IPython's new API for high-level parallel computing. As of this year, IPython has a completely new backend to provide easy-to-use distributed and parallel computing tools, based on the high-performance ZeroMQ networking library. With the rapid rise of multicore systems and clusters, the flattening of the speed curve for modern microprocessors and the ever-increasing sizes of datasets in neuroscience, most scientists today will need to parallelize many of their analyses in everyday work. Yet, classical parallel computing tools tend to be cumbersome to use, deploy and interact with, as they have been classically tuned for absolute performance at all costs, at the detriment of usability. IPython, instead, tries to provide a very high-level, easy to understand and use model for the parallelization of common tasks, retaining enough performance to be useful in production contexts but with a constant concern for the scientist's productivity rather than absolute computational performance. We will present a description of the main concepts involved, the abstraction model offered by IPython and some simple examples of practical usage.

Nitime can be found at: http://nipy.org/nitime and IPython at: http://ipython.org.