Efficient Data Science in Python

In recent years Python has exploded onto the data-science scene, and with it has come a great swathe of data-oriented packages. However, as easy as these packages make analysis, using these tools efficiently requires much more know-how. By the end of this course participants will be able to locate and address bottlenecks in their data-science workflows, using a number of different techniques and tools.

Course Outline

- **Profiling and timing code**: The process of identifying bottlenecks in code is the first and most important step to improving efficiency. Here we will see how we can identify our bottlenecks, and quantify future improvements.
- **Vectorising computations**: Here we will discuss *why* vectorisation is efficient, and see how we can vectorise code with NumPy and pandas. We also see how we can use BLAS libraries to speed up our code for free.
- **Just-in-time compilation**: JIT compilation has the potential to drastically improve the performance of your code, whilst requiring minimal alteration. Here we review the PyPy and Numba project's JIT support.
- **Cython**: A discussion of compiled vs. interpreted languages, and their associated performance. An introduction to Cython and its static type declarations.

Learning Outcomes

By the end of the course participants will be able to…

- identify bottlenecks in the performance of their code
- replace costly for-loops with vectorised NumPy and pandas operations
- select a BLAS library appropriate for their system
- use PyPy and Numba to bring just-in-time compilation to their data-science workflow
- understand Cython's static type declarations and be able to write and execute Cython code.

Attendee Feedback NA
