Introduction to scientific computing

OpenSourceEconomics

March 16, 2020

We teach basic software engineering, numerical methods, and computational engineering skills. They allow you to leverage tools from computational science and increase the transparency and extensibility of our implementations. In doing so, you expand the set of possible economic questions that you can address and improve the quality of your answers.

We organize the course around the two flagship codes of our group. We use two codes to explore selected issues in software engineering, numerical methods, and computational engineering.

- `respy`

  We maintain a Python package for the simulation and estimation of a prototypical finite-horizon dynamic discrete choice model based on (Keane and Wolpin, 1997). Additional information about the project is available in our online documentation at `respy.readthedocs.org`.

- `estimagic`

  We maintain a Python package that helps to build high-quality and user-friendly implementations of (structural) econometric models. `estimagic` provides a consistent interface to a large set of global and local optimizers. All optimizers can handle linear equality and inequality constraints, as well as many other types of constraints. Complicated optimizations can be monitored using an interactive browser-based dashboard. Moreover, `estimagic` provides functions for numerical differentiation and inference for maximum likelihood and method of simulated moments estimation. More details can be found in the online documentation.

Throughout the course, we will make heavy use of Python and its SciPy ecosystem and Jupyter Notebooks. Basic knowledge of this toolchain is a prerequisite. There exist numerous introductory resources, and we provide a curated list here. We will use Zulip for all course communications. Above all, we will post a host of supplementary material there. Please be sure to join our workspace.

- **Introduction**

  We provide a general introduction to computational modeling in economics and our OpenSourceEconomics initiative. We outline opportunities for students to get involved.

- **Testing**

  We discuss different types of automated tests and how they can be applied to scientific software projects. After presenting the basics, we show some examples in `estimagic` and `respy`.

- **Collaboration**

  Collaboration We introduce continuous integration features of GitHub and discuss workflows for different team sizes.

- **Numerical optimization**

  After a brief theoretical introduction to local and global optimization, students will solve straightforward optimization problems with scipy. We then discuss the limitations of scipy.optimize in typical econometric workflows and introduce the students to the powerful optimization tools of the estimagic package.

- **Numerical derivatives**

  After a brief introduction to numerical differentiation, we show how to calculate gradients, Jacobian and Hessian matrices using estimagic.

- **Numerical integration**

  We first introduce the participants to quadrature, Monte-Carlo, and Quasi-Monte-Carlo methods and provide them with (over)simplified guidelines to choose between the different methods. We then show the convergence speed of different approaches in a discrete choice dynamic programming model.

- **Discrete choice dynamic programming models**

  We first outline the underlying economic, mathematical, and computational model. We then provide an overview of its applications in economics. Finally, we use respy to showcase the application of earlier lessons for numerical integration, parallelization strategies, version control, software testing, and continuous integration.

- **Execution speed**

  Execution speed Python is often called a slow language. Nevertheless, it is one of the most widely used languages for high-performance computing. We teach students how to make their Python code fast with numpy and numba and how to parallelize it using multiprocessing and joblib. After the students solve simple examples, we look at some implementations is respy.

References

Gabler, J. (2019). A python tool for the estimation of (structural) econometric models.

Keane, M. P. and Wolpin, K. I. (1997). The career decisions of young men. *Journal of Political Economy*, 105(3):473–522.