Short Introduction to Programming for Economists

Albert Rodriguez Sala

IDEA-UAB

September 24, 2020

1st: Choose a program that you like

What to take into account:

- Compilation and execution speed. The harder you go on quantitative economics (heterogeneous agents) the more you'll need.
- Easiness of syntax/coding: Learning cost. Code-writing time.
- Commercial licences vs open software.
- Resources and community behind?
- General programming language?
- Choose a program that you like the interface/display! You might spend most of your days in front of it. Better you like it. Choosing a language depends on what you want to do in research/career and your background.

Comparing some languages

- Matlab.
- Python.
- Julia.
- C++, Fortran.
- For a proper comparison of the first 3, check: Coleman, Maliar, Maliar (2018)

Matlab

Pros

- **The program of choice for most economists**.
- Easy syntax and learning.
- a vast catalog of pre-written numerical routines. Great to work with matrices, linear algebra, optimization methods...

Con

- low speed for big problems (NOT JIT).
- Not open software, commercial license. (this could be a pro for some people).
- Not a general programming language. Ex: Not good for data cleaning and analysis.

Python

Pros

- Most trending coding language. The economist
- Great documentation with a huge community behind. Many developers, collaborators and ample learning sources.
- Open software.
- Versatile language (GPL).

Cons

- Slower than other languages for certain types of algorithms (not JIT).
- I experienced optimization routines are less stable than in Matlab.

Julia

Pros

- All of Julia is based around **just-in-time (JIT) compilation**: all code is compiled before it is executed: potential to run at the same speed as languages like C or Fortran.
- Similar syntax as matlab. Symilar structures as python (object oriented programming).

Cons

- Small but growing community. Self-learning.
- Lack of stability in updates, problems with interface (my colleagues use Atom).
- Packages still emerging.

C++, Fortran

Pros

- Just-in-Time compilation. Highest execution times.
- Macroeconomists in heterogeneous agents work with it: Violante, Rios-Rull, Raul, Alex ...

Cons

- High entry/learning cost.
- Code-writing can be much slower which reduces the gains of higher execution speed.

I do not recommend them for people that do not have some experience at programming.

- The more languages you know, the easier it is to learn a new one. Especially true with the cheat-sheets and other resources online.
    1. If you go to academics: Your co-authors might use a different language.
    2. If you go to the job market: Not all firms demand the same program skills. Diversification can be optimal.

Resources

- Google. Most important one.
- QuantEcon website (T. Sargent and J. Stachurski). The way I went. I'd strongly recommend for those who want to use Python or Julia.
- And not forget the quantecon notes: Algorithms for Neokeynesian models, Krusell-Smith, Simulated Method of Moments, etc. also in Matlab.
- Stackoverflow Almost all code doubts that you might have, are already asked (and answered) there. Specially good for python.
- Sergei Maliar website. If you work with Matlab and want to solve complex heterogeneous agents models, Maliar is a great reference.

Writing Good Code: Coding Style

- Keep your code simple, tidy, and easy to read. Readability > Cleverness.
- Break complexe problems in small taskes. Divide and Conquer.
- Make frequent use of functions (and classes in python).
- Don't use magic numbers. No numbers scattered around. Assign them in a constant.
- Don't Repeat Yourself. See on DRY
- Be lazy, automate: First think and plan-ahead to avoid you overwriting.

Good Practices at doing your homeworks

- Google will be your best friend. If someone has already done it, take it... But first understand it!
- Always keep lecture notes open. Do not deviate from theory and understand what you do/need to do.
- Communicate/work with classmates.
- PS are long plus good coding takes time. Thus, you should expect to spend time (days) working in the PS.