QuantEcon–RSE DIIS Workshop 2019

Computational Economics with Python

Introduction

April 2019

Introduction

Personnel

- John Stachurski
- Natasha Watkins
- Aakash Choudhury

Thanks

- Alfred P. Sloan Foundation
- Research School of Economics
- DIIS and Azadeh Abbasi Shavazi in particular

Timeline

1. **9:30-10:30** Introduction and Overview (JS)
2. 10:30-11:00 tea break
3. **11:00-12:00** Scientific Computing in Python (JS)
4. 12:00-1:00 Lunch
5. **1:00-2:00** Data Analysis in Python Part I (NW)
6. 2:00-2:30 tea break
7. **2:30–3:30** Data Analysis in Python Part II (NW)

Prereqs / Aims / Outcomes

Assumptions = You are

- computer/maths/stats literate
- not familiar with Python

Aims =

- Overview of scientific computing and Python
- Show some examples
- Discuss / argue
- Resources for further study

Overview: Why Python?

"The average python programming in USA is paid with $117000 per year. It's very much awesome."

– Kovid Raj Panthy, The Youngest Programmer of Nepal is an algorithm-ist, programmer, artificial intelligence developer, key-note speaker, blogger, youtuber and Glocal's 20under20.

Source: https://www.codementor.io/coderkovid/why-should-you-learn-python-programming-in-2019-tru8or6yy

Background — Language Types

Proprietary

- Excel
- MATLAB
- STATA, etc.

Open Source

- Python
- Julia
- R

closed and stable vs open and fast moving

Background — Language Types

Low level

- C/C++
- Fortran
- Java

High level

- Python
- Ruby
- Javascript

Low level languages give us fine grained control

Example. $1 + 1$ in assembly

```
pushq   %rbp
movq    %rsp, %rbp
movl    $1, -12(%rbp)
movl    $1, -8(%rbp)
movl    -12(%rbp), %edx
movl    -8(%rbp), %eax
addl    %edx, %eax
movl    %eax, -4(%rbp)
movl    -4(%rbp), %eax
popq    %rbp
```

High level languages give us abstraction, automation, etc.

Example. Reading from a file in Python

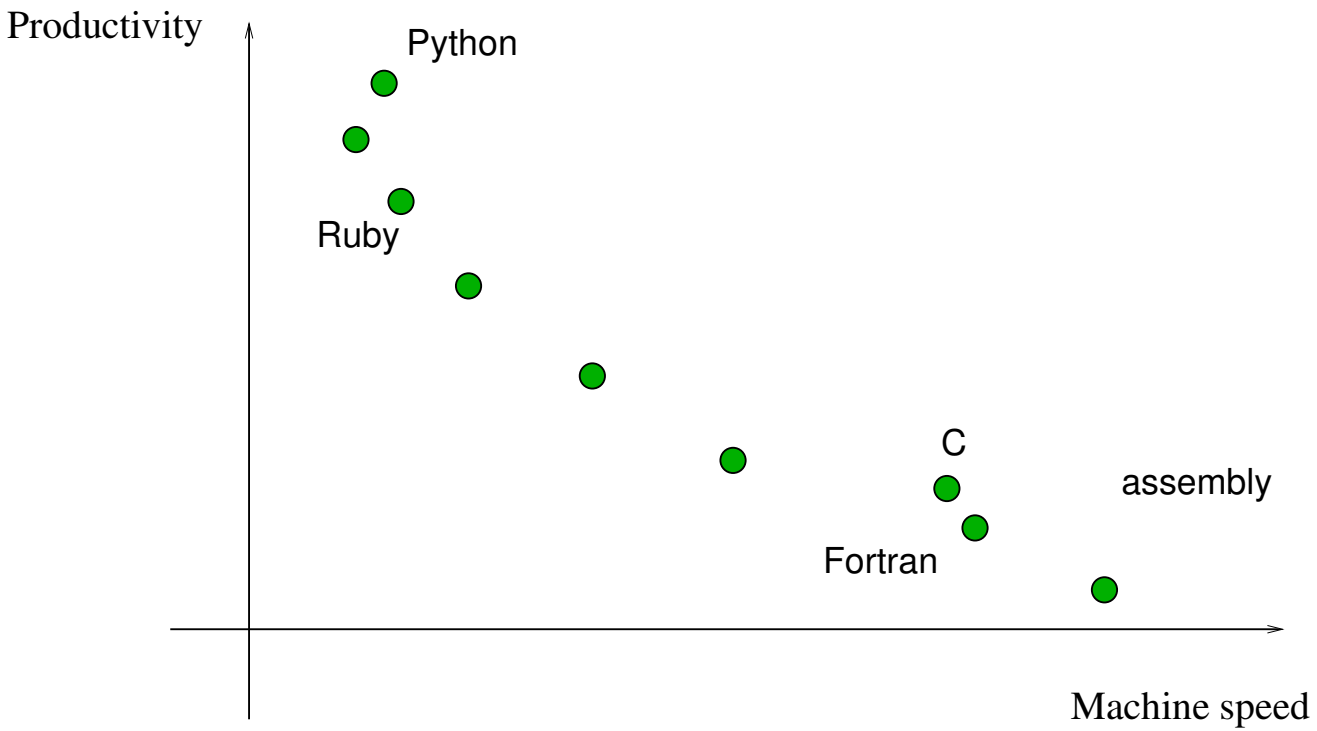
```
data_file = open("data.txt")
for line in data_file:
    print(line.capitalize())
data_file.close()
```

Python for Productivity

> From local infrastructures to cloud-based systems to building websites to interfacing with SQL databases, Python has nearly limitless applications. Despite its wide-ranging impact, it remains gloriously clean and easy to learn.
>
> – *mashable.com*

- core language is small
- **lots** of third party libraries

Trade-Offs

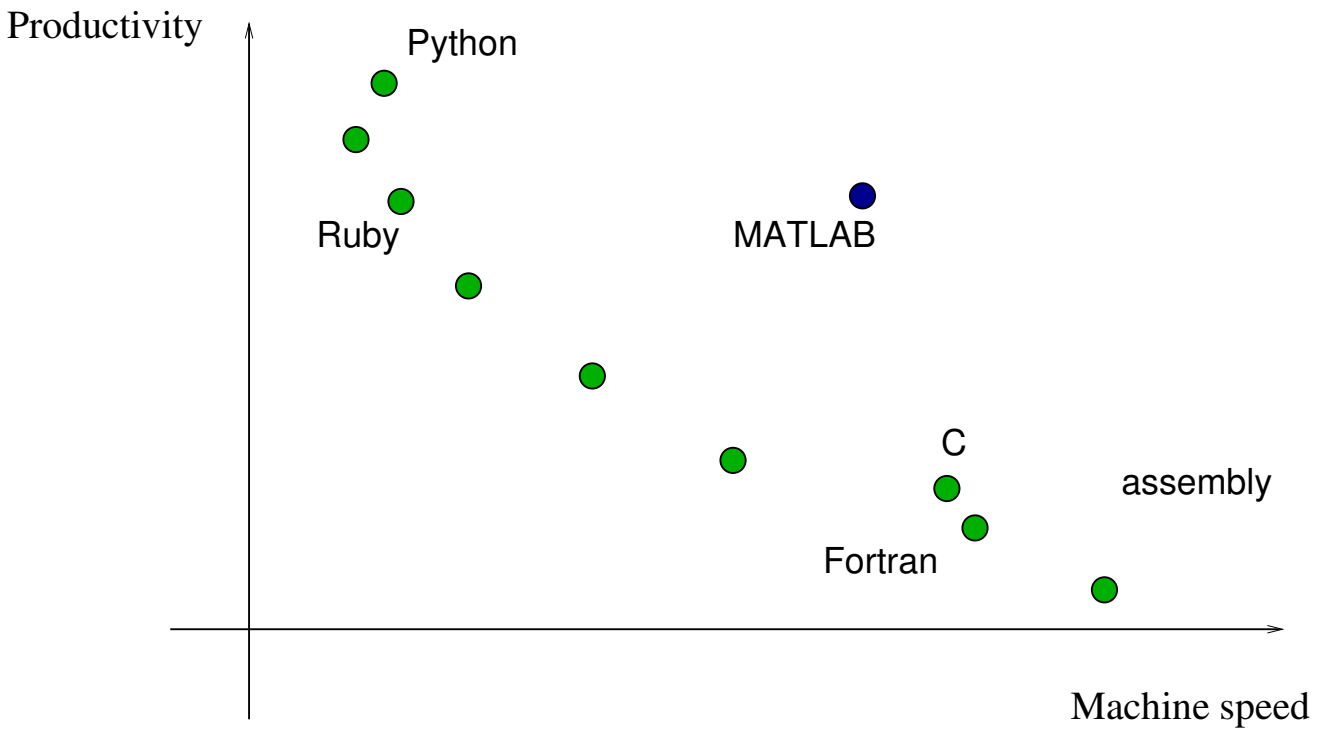

Productivity
Python
Ruby
C
assembly
Fortran
Machine speed

But what about scientific computing?

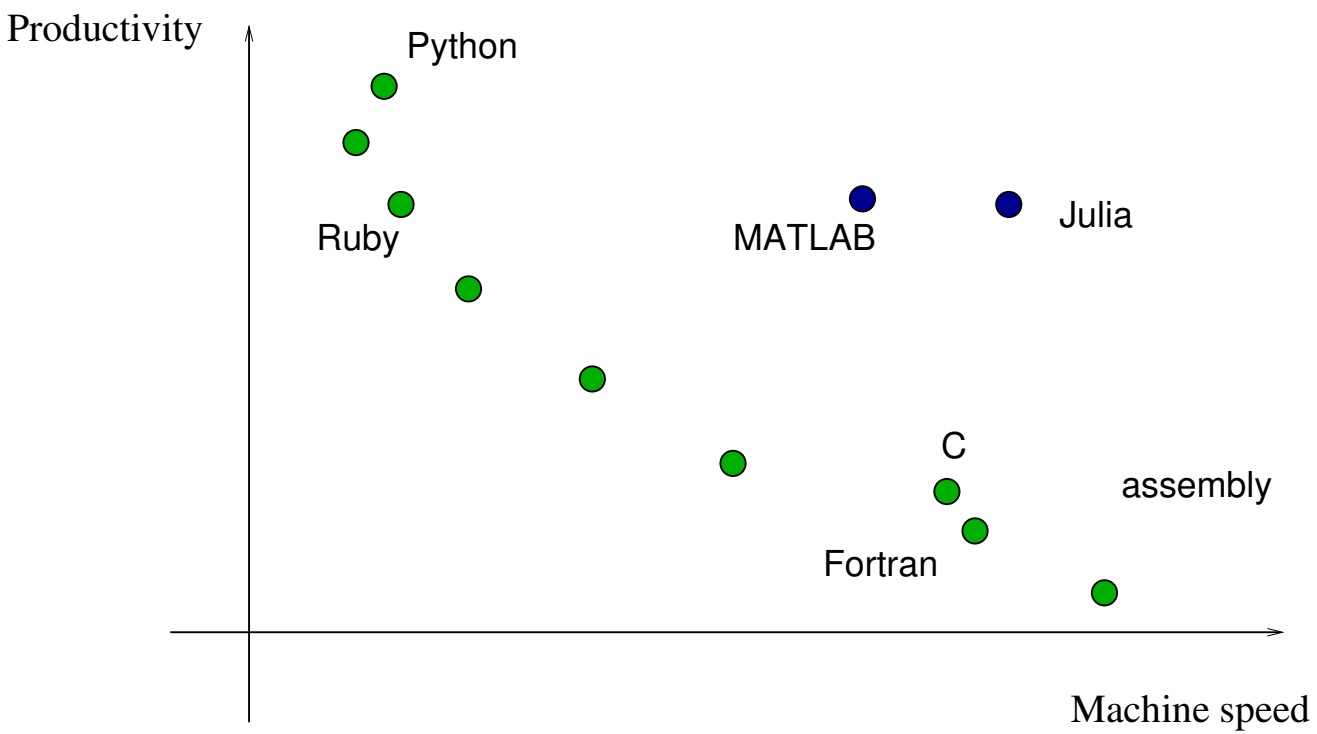
Requirements

- Productive — easy to read, write, debug, explore
- Fast computations

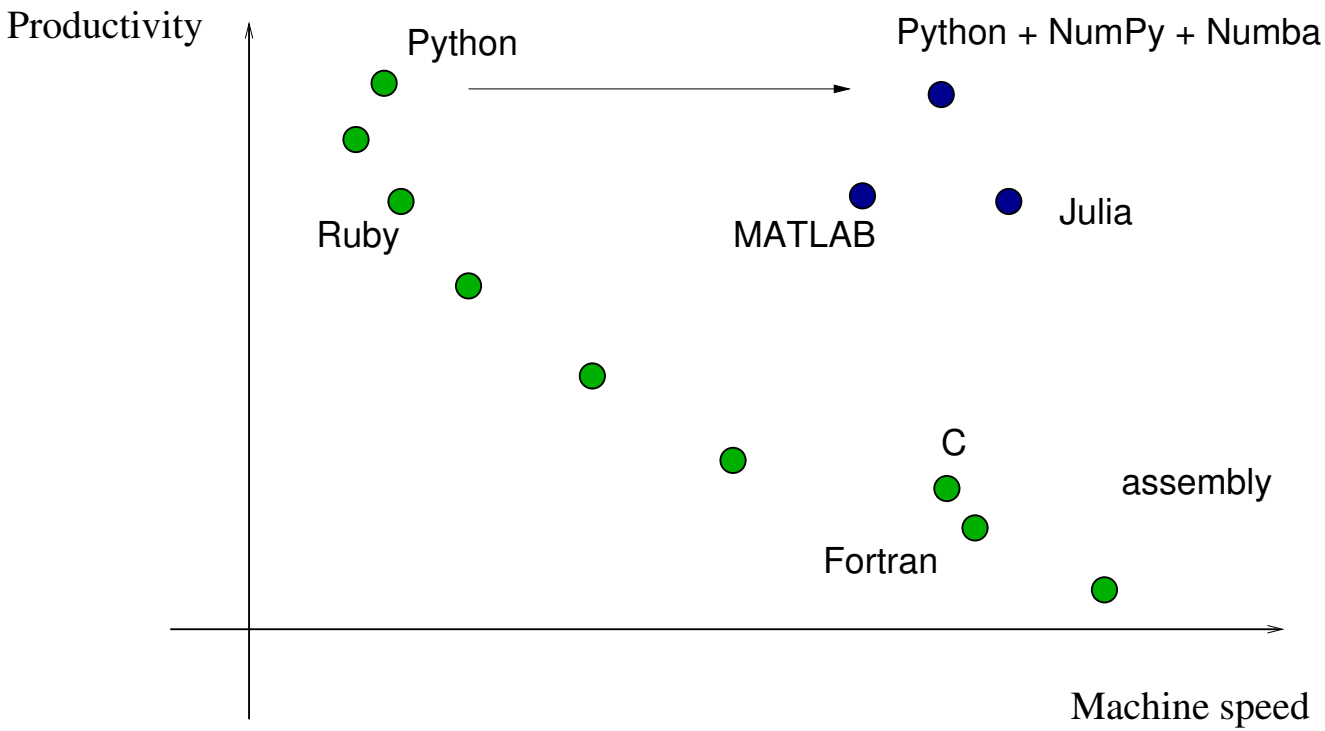
Trade-Offs


Productivity
Python
Ruby
MATLAB
C
assembly
Fortran
Machine speed

Trade-Offs

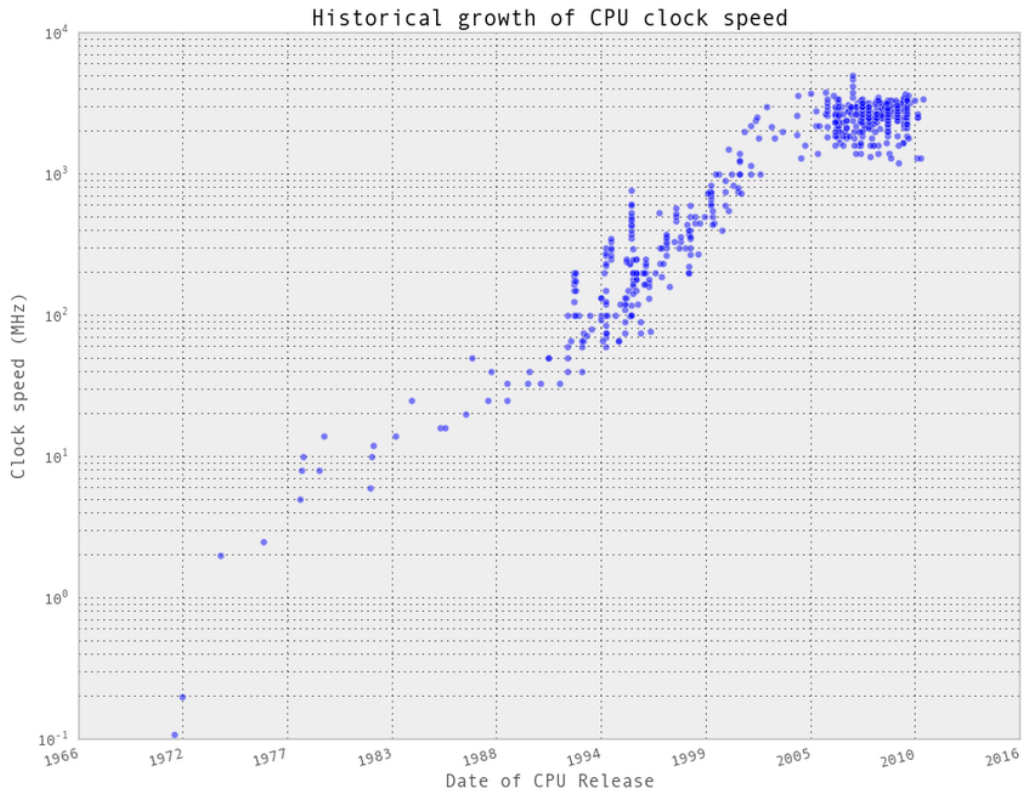

Productivity
Python
Ruby
MATLAB
Julia
C
assembly
Fortran
Machine speed

Trade-Offs

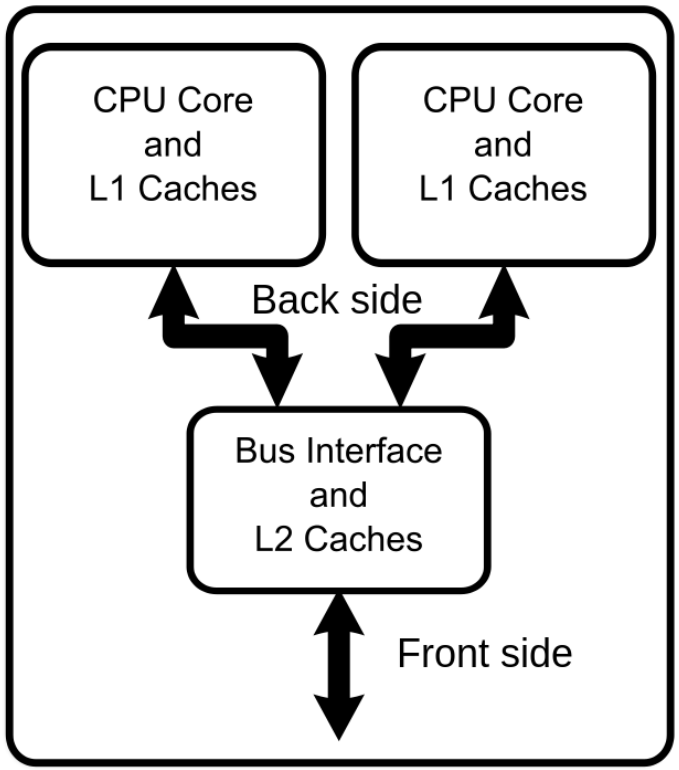

Productivity
Python
Python + NumPy + Numba
Ruby
MATLAB
Julia
C
assembly
Fortran
Machine speed

Programming Background — Hardware

CPU frequency (clock speed) growth is slowing

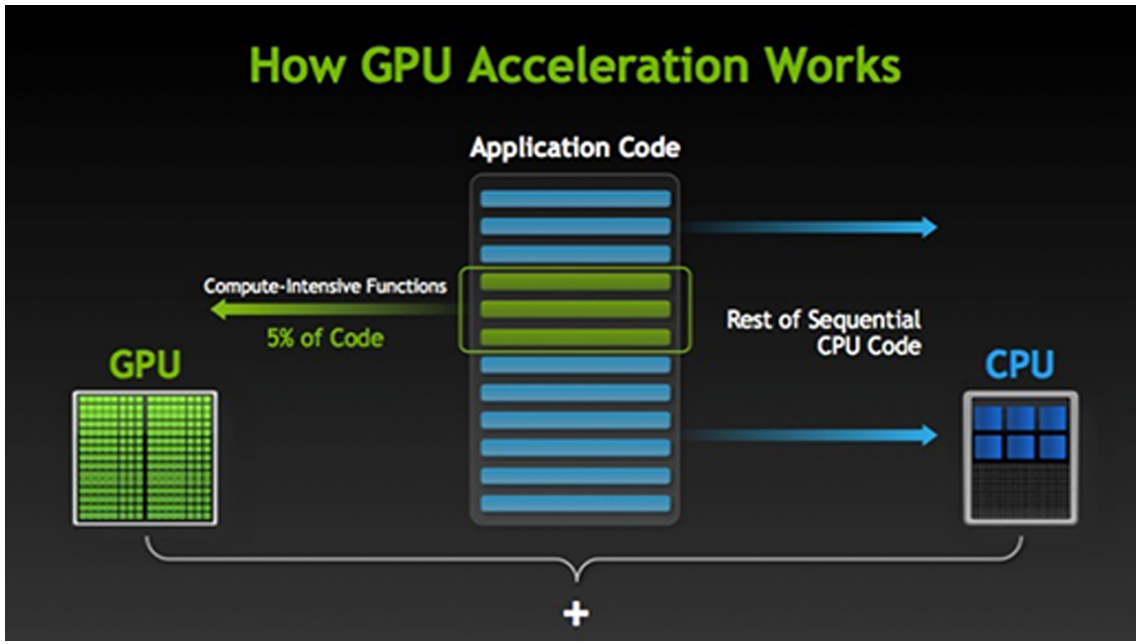


Chip makers have responded by developing multi-core processors



Source: Wikipedia

GPUs / ASICs are also becoming increasingly important



Applications: machine learning, deep learning, etc.

But exploiting multiple cores / threads is nontrivial

Python has strong tools in this area:

- Numpy (implicit multithreading)
- Numba + `@vectorize` + `@jit` + `prange`
- Tensorflow
- PyTorch
- JAX, etc.

Distributed/Cloud Computing

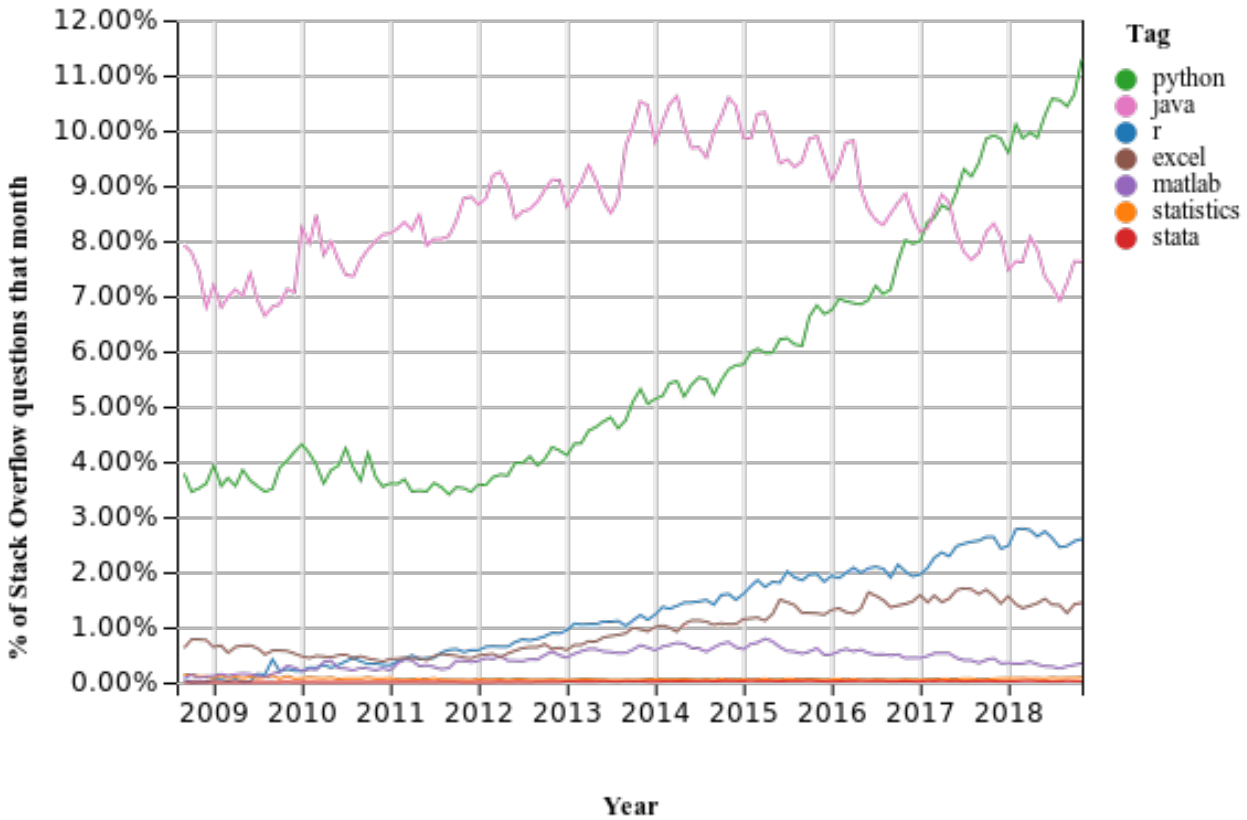
Advantages:

- run code on big machines we don't have to buy
- customized execution environments
- circumvent annoying internal IT departments

Options:

- University machines
- AWS
- Google Colab, etc.

As a result of these advantages:



Workshop Resources

Cheatsheets, downloads, etc. — see

https://quantecon.org/department-of-industry-innovation-and-science

Download workshop files from the GitHub repo

- via `git` or the Download button

Downloads / Installation / Troubleshooting

Install Python + Scientific Libs

- Install Anaconda from `https://www.anaconda.com/downloads`
  - Select latest Python version (3.x)
  - For your OS!
- Not plain vanilla Python

Remote options

- `https://colab.research.google.com`
- `https://notebooks.azure.com/`

Jupyter notebooks

A browser based interface to Python / Julia / R / etc.

Step 1: Open a terminal

- on Windows, use `Anaconda Command Prompt`

Step 2: type `jupyter notebook`

- opening a notebook
- executing code
- edit / command mode
- getting help
- math and rich text
- Jupyter lab