QuantEcon–RSE Honours Workshop 2018

Computational Economics with Python

Introduction

March 2018

Team

- Fedor Iskhakov – ANU
- Matthew McKay – ANU
- Thomas Sargent – NYU
- Sophie Stern – ANU
- John Stachurski – ANU
- Natasha Watkins – ANU

Thanks

- Research School of Economics
- Alfred P. Sloan Foundation

Morning Timeline

1. **9:30-10:30** Introduction and Overview
   - John Stachurski
2. 10:30-11:00 break
3. **11:00-12:00** Introduction to Python
   - Natasha Watkins
4. **12:00-12:30** Scientific Libraries
   - John Stachurski

Afternoon Timeline

1. **13:30–13:45** Quantum Computing
   - Sophie Stern
2. **13:45-15:00** Introduction to Dynamic Programming
   - Fedor Iskhakov
3. 15:00-15:30 break
4. **15:30-17:00** An Optimal Stopping Problem
   - Thomas J. Sargent

Aims / Outcomes / Expectations

Aims =

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

Background — Open Source

Proprietary

- MATLAB
- STATA, etc.

Open Source

- Python
- Julia
- R

closed and stable vs open and fast moving

Background — Language Types

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

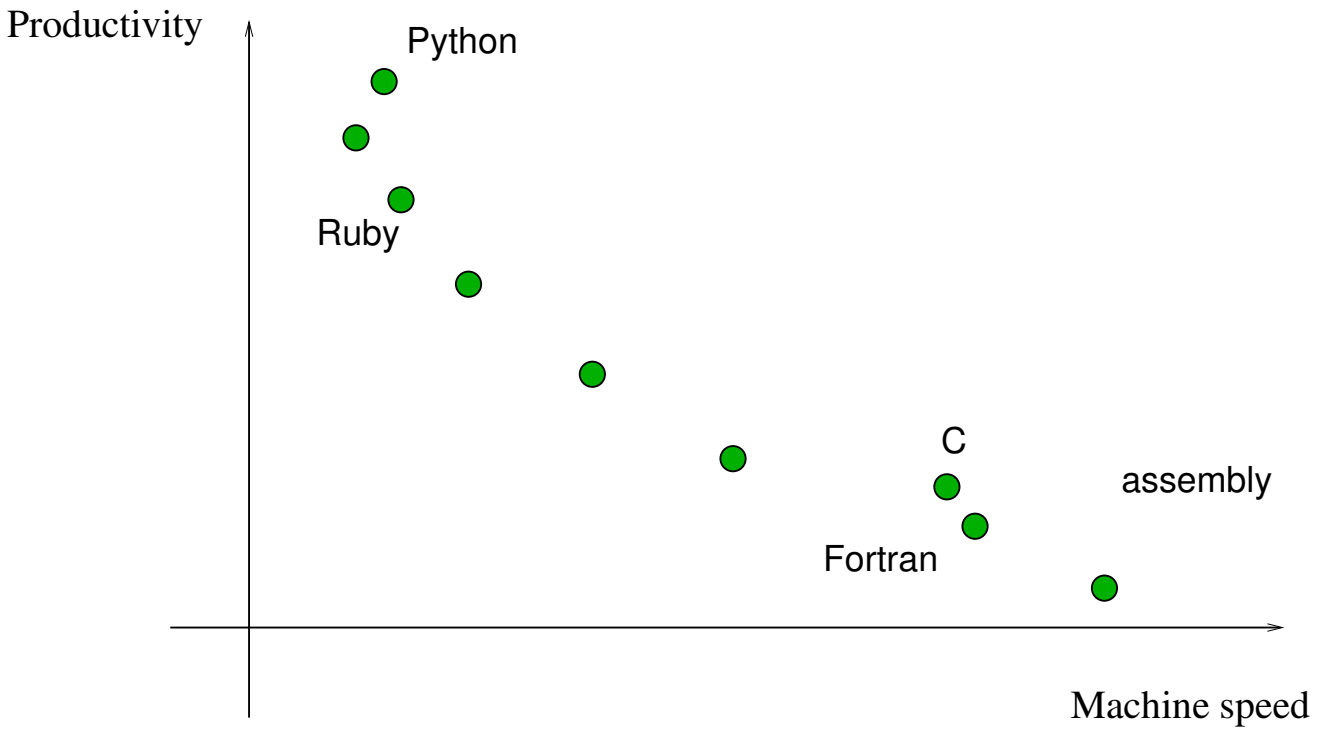
```
data_file = open("data.txt")
for line in data_file:
    print(line.capitalize())
data_file.close()
```

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*

Trade-Offs

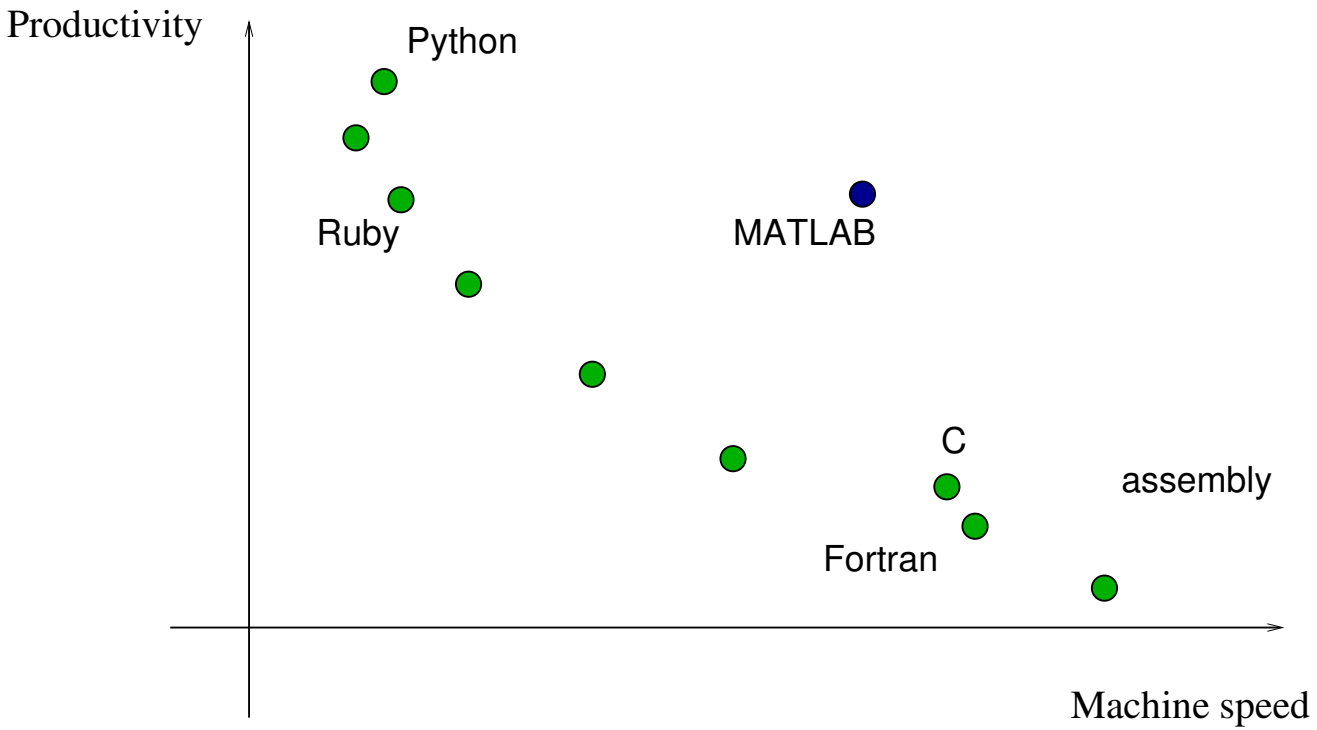

Productivity
Python
Ruby
C
assembly
Fortran
Machine speed

But what about scientific computing?

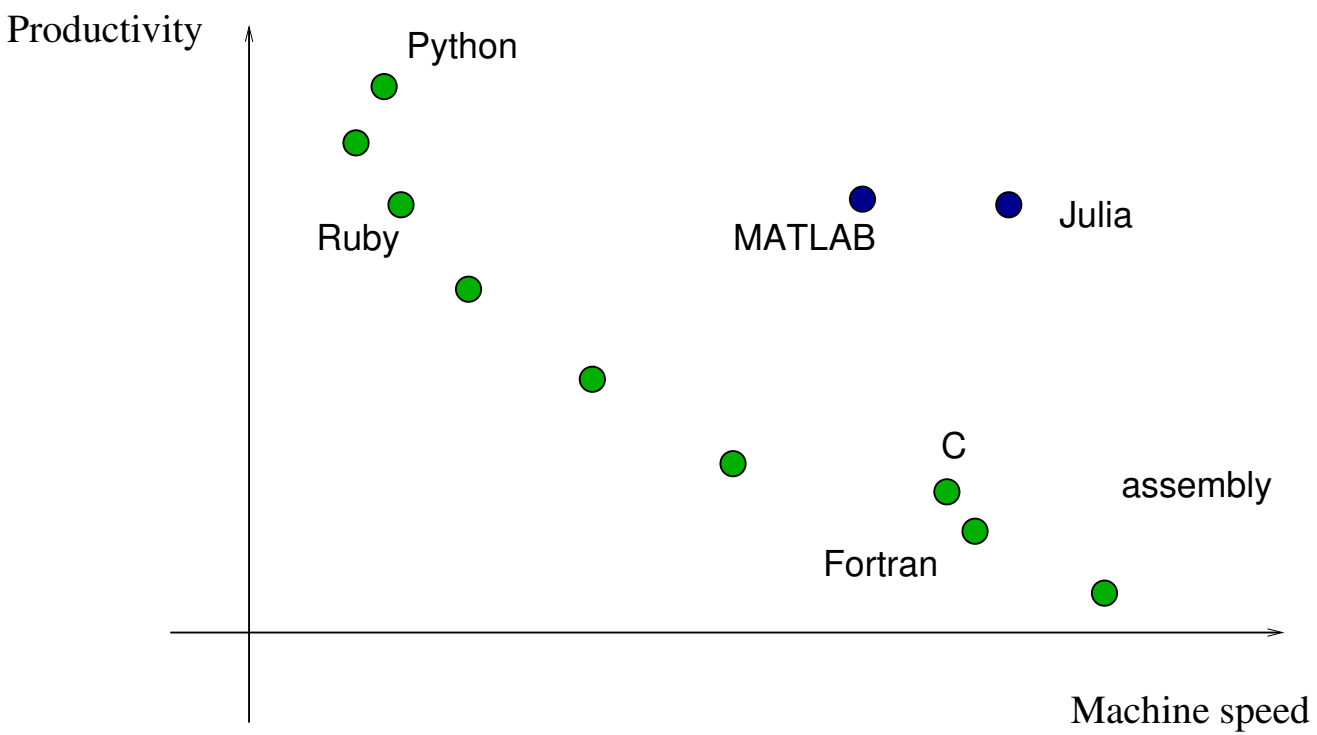
Requirements

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

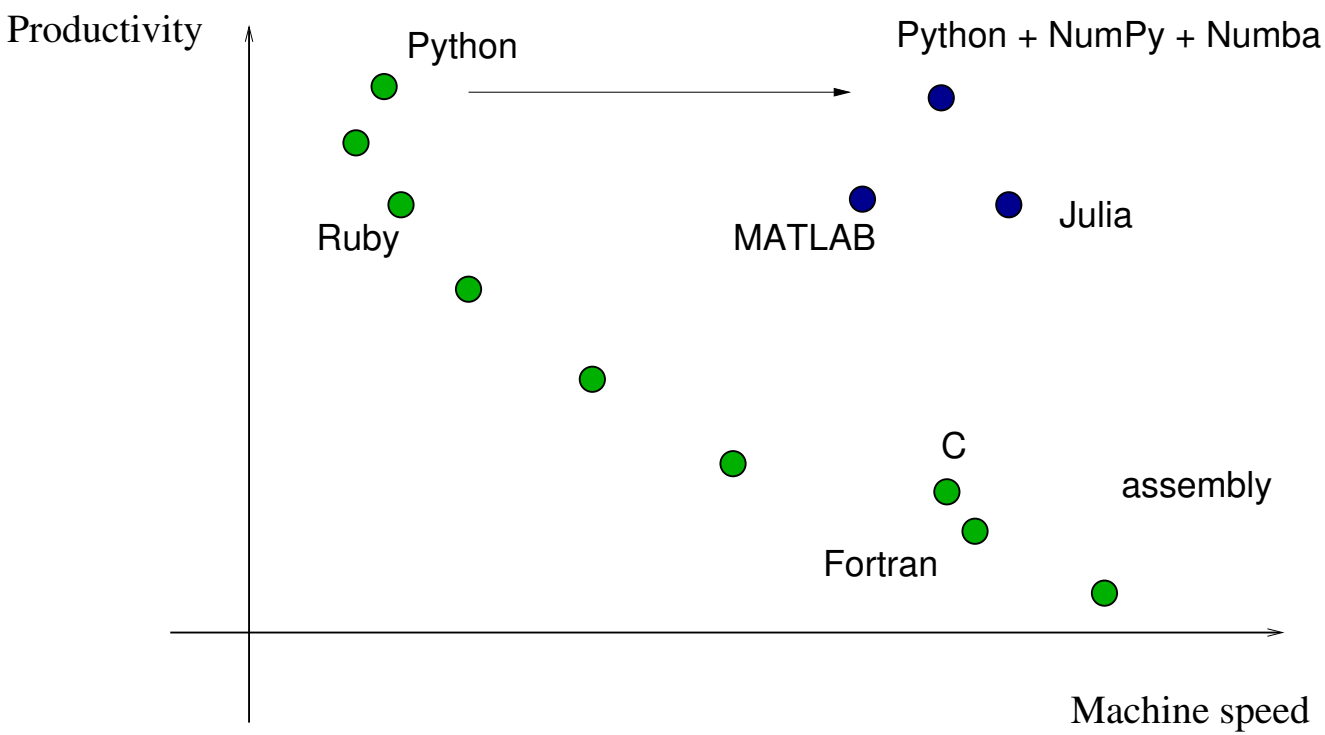
Trade-Offs

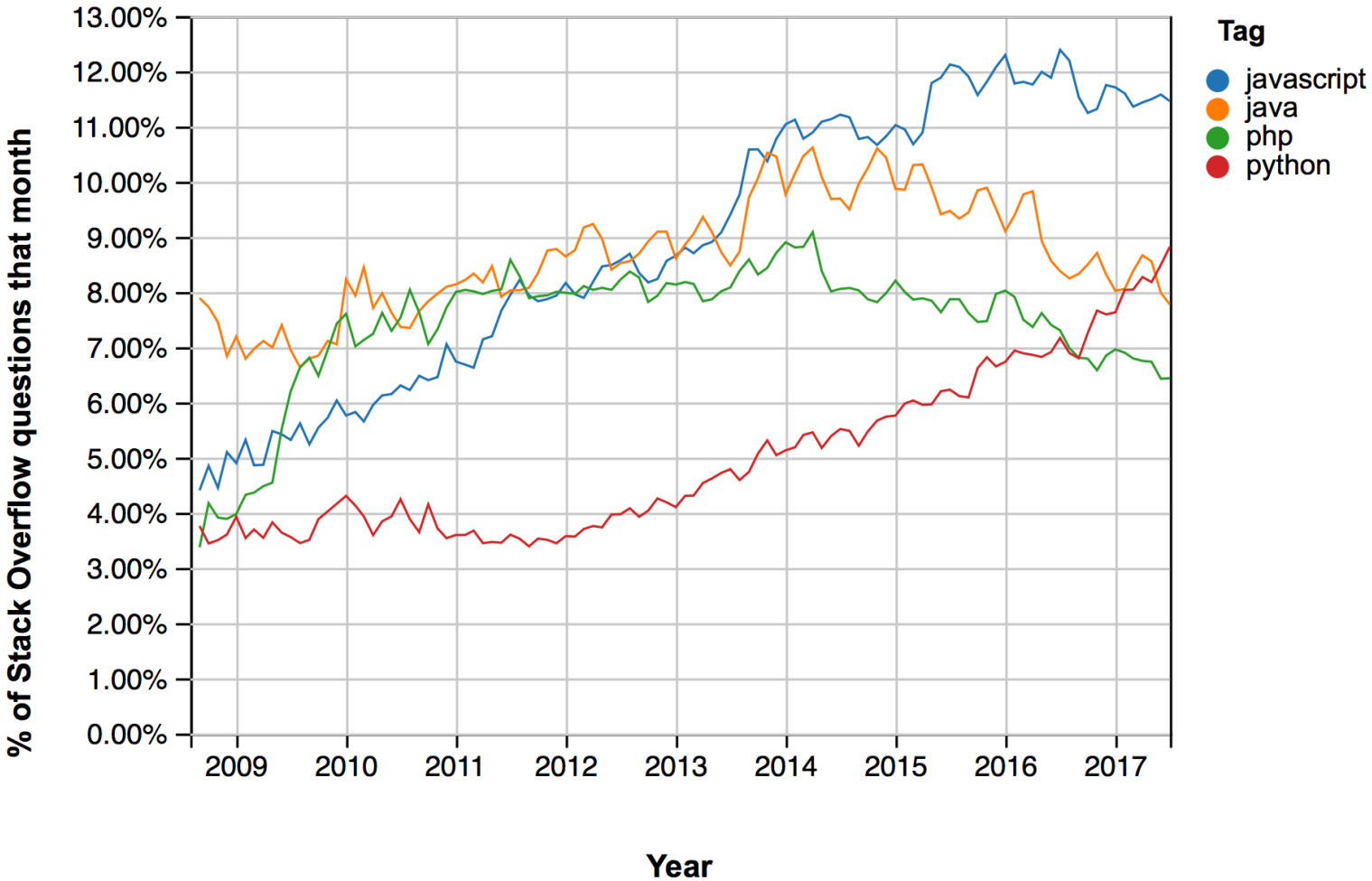

Productivity
Python
Ruby
MATLAB
C
assembly
Fortran
Machine speed

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


% of Stack Overflow questions that month
13.00%
12.00%
11.00%
10.00%
9.00%
8.00%
7.00%
6.00%
5.00%
4.00%
3.00%
2.00%
1.00%
0.00%
2009
2010
2011
2012
2013
2014
2015
2016
2017
Year
Tag
javascript
java
php
python

Workshop Resources

Cheatsheets, downloads, etc. — see

`https://quantecon.org/honours-workshop-2018`

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 Python 3.6
- Not plain vanilla Python

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
- everything's an object (lists, strings)
- installing `quantecon`
- getting help
- introspection
- math and rich text
- Jupyter lab