

# Effective Computing in Research

## Introduction to programming and coding setup

Michael

Konstanz, March 2019

# Roadmap

- 1 Introduction
- 2 Geek Stuff
- 3 Productivity Tools
- 4 IDE
- 5 Which Language Should I pick up?
- 6 Reference

# Introduction

This lecture aims to help people like me to setup the programming and coding environment for their own purposes. With the existence of asymmetric information, We might think programming or coding is super difficult or even have no ideas what's going on there. Therefore, I would try to do:

- Give a brief on what is computing
- Introduce some productivity tools for getting into the programming world
- Show how to setup IDEs
- Explain for what kind of purposes, we should learn R, Python, Matlab or even Julia, Java...etc

# What is computer Science

I used to think that computer science is kind of engineer subject. Now, I realized that this is wrong in most cases. A major portion of computer science, perhaps the most important portion, is not concerned with writing computer code. Instead, it deals with *developing as well as analyzing and evaluating the algorithms* that are then implemented in computer code.

- Option 1: Solve a benchmark set of linear programming problems with today's algorithms on a machine from 1991
- Option 2: Solve a benchmark set of linear programming problems with 1991 algorithms on today's machine

Which option would you choose supposed we had a competition ?

Option 1	Option 2
Algorithm(2000s) and Laptop(1991)	Algorithms(1991) and Laptop(2000s)

Table: Optimization progress comparison from Bixby (2012)

# What is computer Science

- **Answer:** Option 1 would be faster by a factor of approximately 300.
- Compared to 1980s, solving a benchmark set of linear programming problems has gain 43 million-fold speedup. Of this speedup, however, only about **1,000-fold** obtained because of Moore's law; the other **43,000-fold** obtained because of improvements in algorithms.
- According to Bixby (2012), between 1991 and 2008 the improvements arising from better algorithms and implementations have resulted in about a 29,000-fold speedup holding the hardware at constant.
- Your smartphone is millions of times more powerful than all of NASA's combined computing in 1969. But can we send our friends to the Moon with our smartphone?

# Get used to Terminal

Please, don't be terrified by this black terminal window:

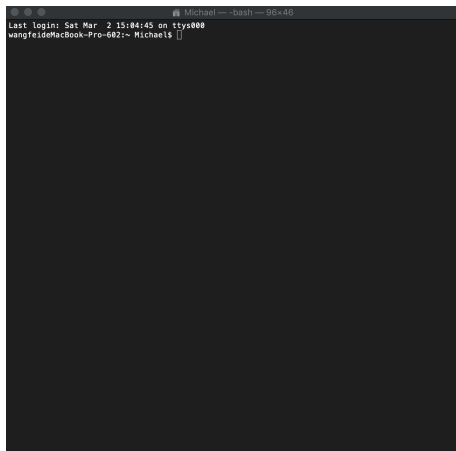


Figure: Terminal on Mac

# Get used to Terminal

Command	Function	Tips
<code>pwd</code>	print the working directory	
<code>man</code>	manual pages for the command	
<code>Q</code>	quit the current section, e.g. manual	
<code>ls</code>	list directory contents	<code>ls -a</code>
<code>~</code>	shorthand for the home directory	
<code>cd</code>	change directory	type one space after <code>cd</code>
<code>mkdir</code>	create directories (or make a file)	with path or without path
<code>rm</code>	delete(or remove) files	with path or without path
<code>cp</code>	copy files	

Table: Essential Terminal Command

# Get used to Terminal

You can customize your terminal like this:

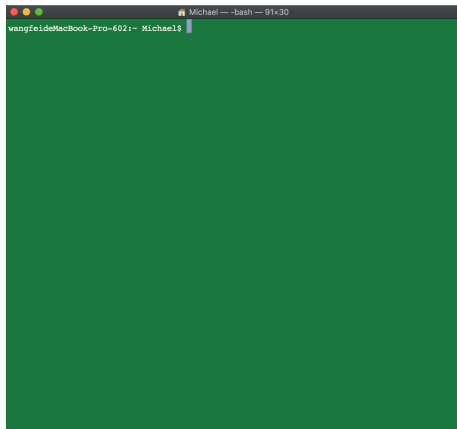



Figure: Terminal on Mac



## Sooner or later, you will be on Github

- GitHub is a web-based hosting service for version control using Git. It is mostly used for computer code. It offers all of the distributed version control and source code management functionality of Git as well as adding its own features.
- Here is the video link where I learned how to use Github:
  - ▶ A Youtube Video - click me 
- Github can be used to impress some girls when you go to a pub.

# Essential Git Command

Command	Function	Tips
<code>git clone</code>	clone cloud file to local directory	
<code>git add</code>	add local file to cloud	<code>git add . (-A)</code>
<code>git commit -m</code>	commit the message	
<code>git push</code>	begin to upload	
<code>git pull</code>	download from branch or cloud	
<code>git add . -dry-run</code>	simulation adding	

# IDE: Integrated development environment

An integrated development environment is a software application that provides comprehensive facilities to computer programmers for software development. An IDE normally consists of

- a source code editor
- build automation tools
- a debugger

We will focus on building up a IDE for Python. For R, I suggest to download Rstudio. For matlab, you will install IDE directly when you install the programming.

# Python IDE: Atom

Atom is a free and open-source text and source code editor for macOS, Linux, and Microsoft Windows with support for plug-ins written in Node.js, and embedded Git Control, developed by GitHub. Atom is a desktop application built using web technologies. Useful guides:

- A short intro on Atom:click me ☕

## Essential Packages:

- platformio-ide-Terminal
- atom-beautify
- atom-ide-ui
- build-python
- autocomplete-python
- hydrogen-launcher
- ide-python
- kite
- script

# Why Atom ?

- Simple
- Elegant
- Beautiful
- Efficient
- Compact

# Python, R, Julia and Matlab

- R - Clean and tidy the data; Statistical modeling
- Python - Scientific Programming
- Matlab/Julia - Numerical operation

Algorithm	Programming	Running time	Using local memory
No	Matlab	40.0244 seconds	relative high (4.92 GB)
	Python	57 - 61.459 seconds	relative low (1 GB)
	Julia	80 seconds	relative low
	R		relative high
Yes	Matlab	3.7044 seconds	
	Python	5.4389 seconds	
	Julia	around 2 seconds	
	R		

**Table:** Comparison for solving RBC model numerically with different programming

# Which language should I use?

I have been struggling with this question for a while. After gaining some experiences on all those languages, here is my final call:

- I will stick to Rstudio for data analyzing and calling statistical models (e.g. VAR, Bayesian Inference)
- I will use Python to do mathematical programming and numerical operations as it is efficient
- I will try to use Matlab as little as possible for not breaking my laptop
- Give up Julia temporarily and wait for the stable version

# Book Recommendation

To prepare this lecture, I have used this book:

- Paarsch, H. J. and Golyaev, K. (2016). *A Gentle Introduction to Effective Computing in Quantitative Research: What Every Research Assistant Should Know*.  
MIT Press



- Bixby, R. E. (2012). A brief history of linear and mixed-integer programming computation. *Documenta Mathematica*, pages 107–121.
- Paarsch, H. J. and Golyaev, K. (2016). *A Gentle Introduction to Effective Computing in Quantitative Research: What Every Research Assistant Should Know*. MIT Press.