

# Version control using Git and Plotting Tutorial

Oliver Thomas

Quantum Engineering CDT  
University of Bristol

September 1, 2018

# Overview

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- Version control
- Plotting using gnuplot
- Vim

# Why you should use version control

- Does this seem familiar?

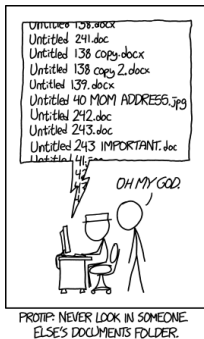


Figure: Bad version control<sup>1</sup>

<sup>1</sup><https://xkcd.com/1459/>

# What is Git?

- Git is one of most used version control software in the world
- Git is cross-platform and easy to use<sup>2</sup>

---

<sup>2</sup><https://try.github.io/levels/1/challenges/1>

# What is GitHub?

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- Github is a cloud service for git which lets you store your repository online
- Why would you store your repository online?
  - Working remotely
  - Collaborative work
  - Hard drive failure!

# Making a repository

- You can do this online on the Github website <sup>3</sup>
- Create a new repository
- Then click clone to get the url, open git on your computer and type:  
`git clone url`

---

<sup>3</sup><https://github.com/>

# Making a repository

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- Go to the folder and right click git with bash
- You are now able to use bash for the rest of the talk!

# Basic Git commands

- There are four<sup>4</sup> important commands you will need for git:
- `git pull`
- `git add *`
- `git commit -a`
- `git push`

---

<sup>4</sup>I cheat here and write a bash script which does these in order so I only have to run a single command.



# A brief note on text editors: Vim

- Vim is a powerful cross-platform text editor, released in 1991 and is still regarded as one of the most popular editors <sup>5</sup>.
- Flexible with thousands of plugins available e.g. I use Vim to compile latex documents, this presentation was written in Vim.
- Computing clusters normally only have CLI so if you are running high performance code you will need to be familiar with Vim, Emacs or Nano.
- You can feel like a Hacker.

---

<sup>5</sup>Along with Emacs

# Vim commands

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- The most important thing to remember is that Vim has two main modes, *NORMAL*, ESC and *INSERT*, i
- All commands are run from *NORMAL* mode using :
- to quit use, ESC:q (meaning go to *NORMAL* mode, : means command and q is quit without saving)
- to save and quit use, ESC:wq (w stands for write)

# Adding your first commit

- Every repository should contain a readme
- Then either run:
- `git add *`
- `git commit -a`
- `git push`

Or use the windows GUI version and commit them to your repository.

# gnuplot

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- Gnuplot is popular, multi-platform and standard software on computing clusters<sup>6</sup>
- <https://sourceforge.net/projects/gnuplot/files/gnuplot/5.2.4>

---

<sup>6</sup>standard on most of the popular linux distributions

# Example 1, Plotting functions

- Go to the src folder
- open gnuplot and type  
`load 'ex1gnu.p'`

# Example 1, Plotting functions

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

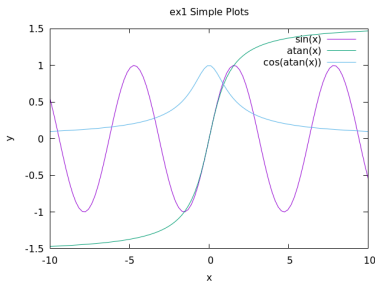


Figure: function plotting

- It could do with some axis labels.
- go into the program and find the line called `plt.ylabel=` and `plt.xlabel=`

# Example 2, Complicated functions!

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

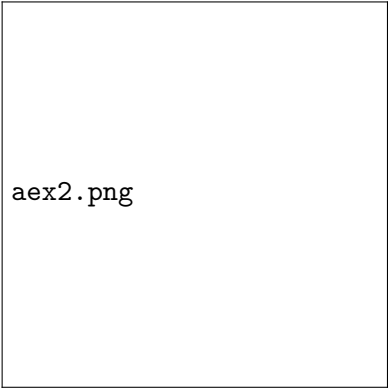
- In the `src` folder open `ex2compfunctions.py`
- Run `all.py` and choose 2

# Example 2, Complicated functions!

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- Figures!



aex2.png

Figure: function plotting



# Example 3, Plotting data!

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- once again, in the `src` folder open `ex3data.py`
- Run `all.py` and choose 3

# Example 3, Plotting data!

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- figure

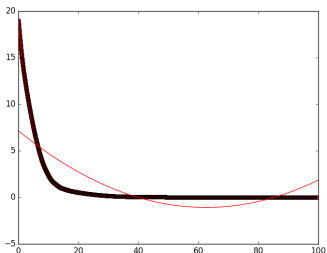


Figure: function plotting

# Example 4, Histograms!

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- once again, in the src folder open `ex4hist.py`
- Run `all.py` and choose 4

# Example 4, Histograms!

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- figure

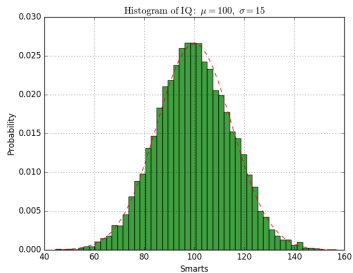


Figure: function plotting

# Example 5, Subplots!

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- In the src folder open `ex5subplots.py`
- Run `all.py` and choose 5

# Example 5, Subplots!

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- Figures!

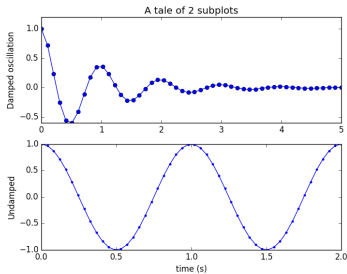


Figure: function plotting

# Example 6, Art!

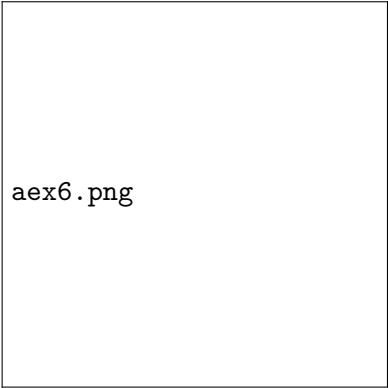
Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- In the `src` folder open `ex6art.py`
- Run `all.py` and choose 6

# Example 6, Art!

- Figures!



aex6.png

Figure: function plotting



# Advanced Git commands

- One of the great things about Git is that you can get by with just the four above commands.
- The git man page is very useful, especially,  
`man gittutorial`  
`man giteveryday`
- `giteveryday` is a super useful collection of the 20 commands you will need regularly.

# Branching

- Branching is useful, it lets you test something out separately to the main branch.
- To make a new branch called test  
`git branch test`
- You can check all of the current branches and which branch you are on with  
`git branch`

# Branching

- To switch to the test branch type:  
`git checkout test`

# Adding Collaborators

Version  
control using  
Git and  
Plotting  
Tutorial

Oliver Thomas

- Go to a repository and on the settings tab click collaborators, you can then search for the github username

# Thanks for listening!



Figure: If it all goes wrong ...<sup>7</sup>

<sup>1</sup><https://xkcd.com/1597/>