

# Getting Started with Python



[https://github.com/ryan-leung/PHYS4650\\_Python\\_Tutorial/](https://github.com/ryan-leung/PHYS4650_Python_Tutorial/)

*Ryan Leung (PhD in Astrophysics)*

*Python Tutorial for PHYS 4650*

*8th Feb, 2017*

Please try to set-up your  
**python** in your computer  
/open **Jupyter**

*While listening, you may goto **Page 14***

# Learning Outcomes

1. Choose your own text-editor.
2. Get a working python for your OS.
3. Use python in ipython notebook.
4. Define different data structures.
5. Make a loop with for and while.
6. Defining functions.
7. Reading files and plotting graphs.
8. And practising with examples.

# Part 1

## Installing/Running Python

# Why python ?

*Python and Programing*

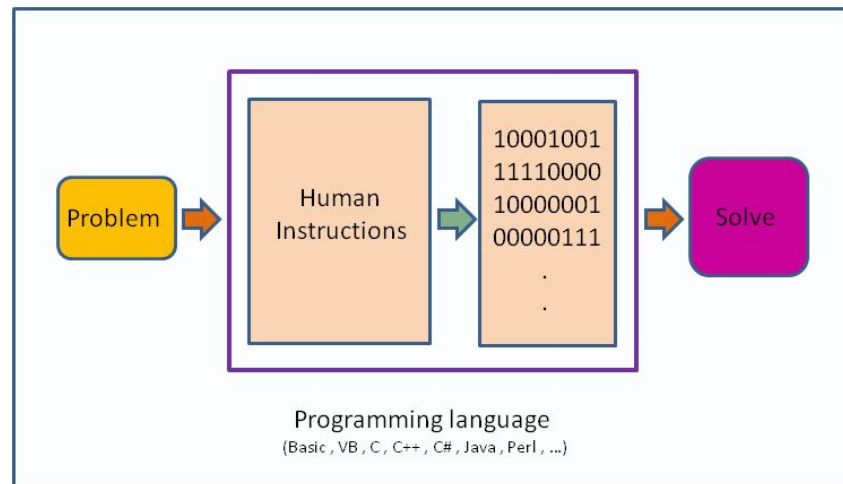
# What is python language?



- High-level programming language.
- Object-oriented, interpreted, interactive.
- Easy write, easy read.
- Dynamic variables & memory management.

# Programming basic

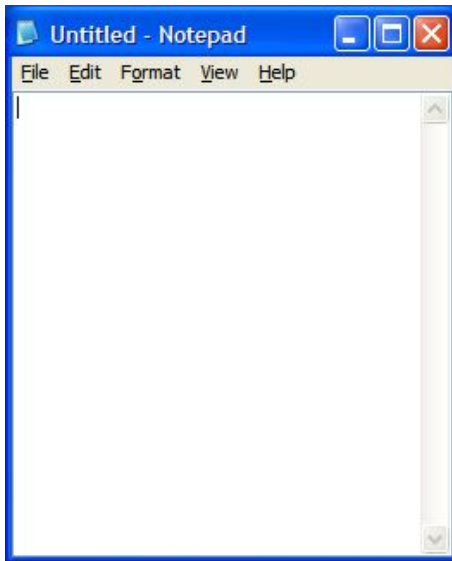
- Sequences of instructions that tell the computer to solve your problem.
- Like cooking,
  - A program is a **receipt**.
  - You prepare raw food, seasoning, etc. (Input)
  - If you follow the receipt, you will get a good food (hopefully).



# Writing a program

First, get yourself a **text editor**.

Avoid using old-school editor:





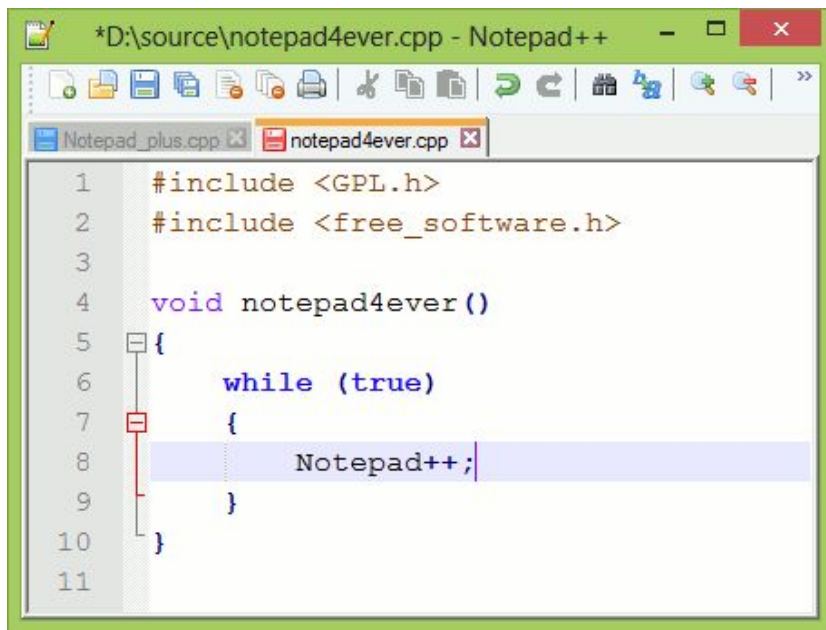
# Writing a program in command line (mac / linux)

1. Vim
2. nano

# Writing a program (GUI editor)

For windows:

## Notepad++

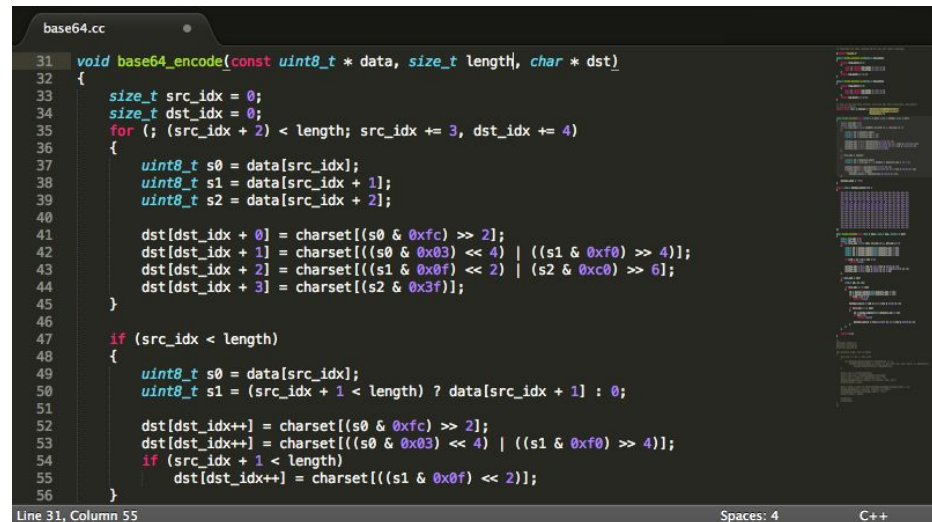


```
1  #include <GPL.h>
2  #include <free_software.h>
3
4  void notepad4ever()
5  {
6      while (true)
7      {
8          Notepad++;
9      }
10 }
11
```

Notepad++: <https://notepad-plus-plus.org/>

Sublime Text: <https://www.sublimetext.com/>

## Sublime Text



```
31 void base64_encode(const uint8_t * data, size_t length, char * dst)
32 {
33     size_t src_idx = 0;
34     size_t dst_idx = 0;
35     for (; (src_idx + 2) < length; src_idx += 3, dst_idx += 4)
36     {
37         uint8_t s0 = data[src_idx];
38         uint8_t s1 = data[src_idx + 1];
39         uint8_t s2 = data[src_idx + 2];
40
41         dst[dst_idx + 0] = charset[(s0 & 0xfc) >> 2];
42         dst[dst_idx + 1] = charset[((s0 & 0x03) << 4) | ((s1 & 0xf0) >> 4)];
43         dst[dst_idx + 2] = charset[(s1 & 0x0f) << 2 | (s2 & 0xc0) >> 6];
44         dst[dst_idx + 3] = charset[(s2 & 0x3f)];
45     }
46
47     if (src_idx < length)
48     {
49         uint8_t s0 = data[src_idx];
50         uint8_t s1 = (src_idx + 1 < length) ? data[src_idx + 1] : 0;
51
52         dst[dst_idx++] = charset[(s0 & 0xfc) >> 2];
53         dst[dst_idx++] = charset[(s0 & 0x03) << 4 | ((s1 & 0xf0) >> 4)];
54         if (src_idx + 1 < length)
55             dst[dst_idx++] = charset[(s1 & 0x0f) << 2];
56     }
57 }
```

# Writing a program (GUI editor)

For mac:

1. Brackets
2. Textmate
3. Sublime Text

Brackets: <http://brackets.io/>

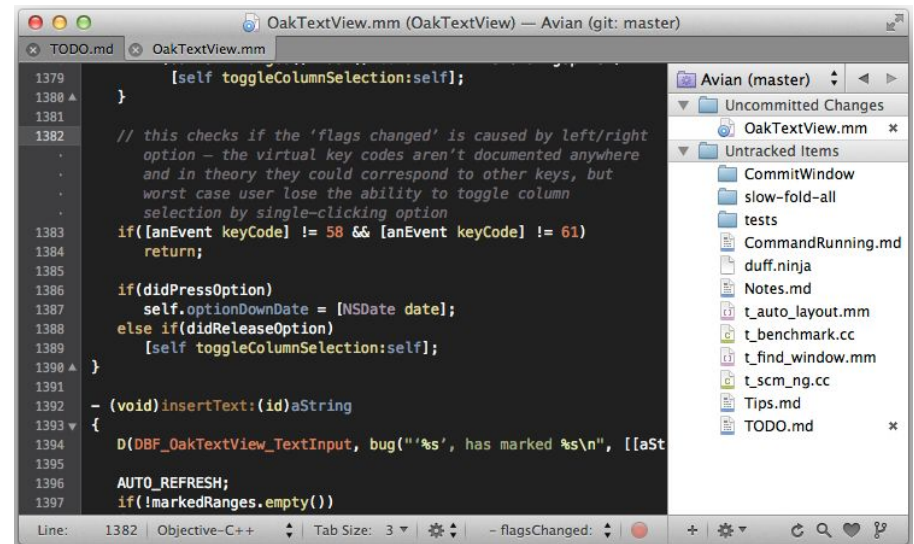
Textmate: <http://macromates.com/>



```
103
104 <div id="hero-wrapper">
105   <div id="hero" class="row">
106     <div class="large-12 columns">
107       <h1>Brackets is an open source code editor for web designers and front-end developers. <a class="not
108       <div id="download">
109         <a id="hero-cta-button" href="https://github.com/adobe/brackets/releases/latest" class="large rd

378 #hero-cta-button {
379   font: 400 18px source-sans-pro, sans-serif;
380   text-align: center;
381   padding-left: 1.5em;
382   padding-right: 1.5em;
383 }

110   <span id="download-version" class="nowrap" data-i18n="index.page.hero.latest-release">Latest
111   </a>
112   <div id="os-alert" class="alert-box radius" data-alert>
113
114   </div>
```



```
1379 [self toggleColumnSelection:self];
1380 }
1381
1382 // this checks if the 'flags changed' is caused by left/right
1383 // option - the virtual key codes aren't documented anywhere
1384 // and in theory they could correspond to other keys, but
1385 // worst case user lose the ability to toggle column
1386 // selection by single-clicking option
1387 if([anEvent keyCode] != 58 && [anEvent keyCode] != 61)
1388   return;
1389
1390 if(didPressOption)
1391   self.optionDownDate = [NSDate date];
1392 else if(didReleaseOption)
1393   [self toggleColumnSelection:self];
1394 }
1395
1396 - (void)insertText:(id)aString
1397 {
1398   D(DBF_OakTextView_TextInput, bug("'s', has marked %s\n", [[aSt
1399
1400   AUTO_REFRESH;
1401   if(!markedRanges.empty())
```

# Writing a program (GUI editor)

For linux:

1. **gedit**
2. **Geany**
3. **atom**
4. **Sublime Text**

# Writing a program

- Second, think what you are going to do
- Define some initial constant as “**variables**”
- Write some “**operations**” ==> final products.

But problems usually comes out,  
and that's call “**bugs**”.

# Bug is hard to kill!

- Coding and debugging is a tough task
- **Spend less time debugging, Code more!**
- **Lots of packages written in Python for speed deployment!**



# How can I get python?

*Install python and get it works*

# Python distribution

## **Anaconda / Miniconda (Win, mac, Linux)**

<https://www.continuum.io/downloads>

<http://conda.pydata.org/miniconda.html>

## **Official Site (Win, mac, Linux)**

<https://www.python.org/downloads/>

## **Linux: pre-installed, or get it from repository:**

Fedora: `sudo dnf install python`

Ubuntu: `sudo apt-get install python`

## **Mac: macports / homebrew**

If you have macports on your mac, you can follow this instruction in [macports\\_installation.md](#)



# Get Anaconda

Download for Windows

Download for OSX

Download for Linux

## Anaconda 4.2.0

### For Windows

Anaconda is BSD licensed which gives you permission to use Anaconda commercially and for redistribution.

#### [Changelog](#)

1. Download the installer
2. Optional: Verify data integrity with [MD5 or SHA-256](#) [More info](#)
3. Double-click the **.exe** file to install Anaconda and follow the instructions on the screen

Behind a firewall? Use these [zipped Windows installers](#)

### Python 3.5 version

64-BIT INSTALLER (391M)

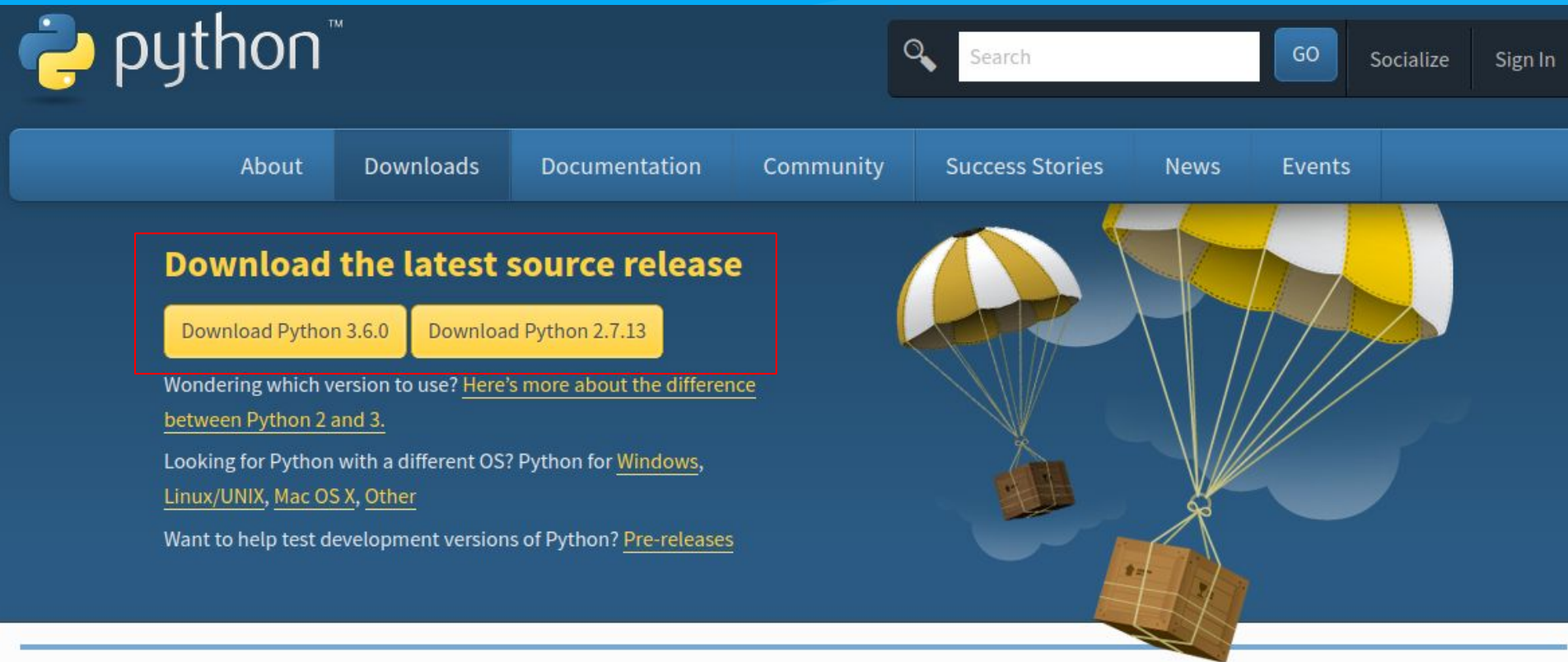
32-BIT INSTALLER (333M)

### Python 2.7 version

64-BIT INSTALLER (381M)

32-BIT INSTALLER (324M)

# From official site



The screenshot shows the Python.org website. At the top is the Python logo and the word "python" with a trademark symbol. To the right is a search bar with a magnifying glass icon, a "GO" button, and links for "Socialize" and "Sign In". Below the header is a navigation bar with links: "About", "Downloads", "Documentation", "Community", "Success Stories", "News", and "Events". The main content area features a red-bordered box with the heading "Download the latest source release". Inside this box are two yellow buttons: "Download Python 3.6.0" and "Download Python 2.7.13". Below the buttons, there is text: "Wondering which version to use? [Here's more about the difference between Python 2 and 3.](#)" followed by "Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)" and "Want to help test development versions of Python? [Pre-releases](#)". To the right of the text is an illustration of two parachutes with yellow and white stripes, each carrying a stack of cardboard boxes.

python™

Search GO Socialize Sign In

About Downloads Documentation Community Success Stories News Events

**Download the latest source release**

Download Python 3.6.0 Download Python 2.7.13

Wondering which version to use? [Here's more about the difference between Python 2 and 3.](#)

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)

Want to help test development versions of Python? [Pre-releases](#)

Looking for a specific release?

Python releases by version number:

Release version	Release date	Click for more	
<a href="#">Python 3.4.6</a>	2017-01-17	<a href="#">Download</a>	<a href="#">Release Notes</a>
<a href="#">Python 3.5.3</a>	2017-01-17	<a href="#">Download</a>	<a href="#">Release Notes</a>

# Python version, 2 vs 3

- **Python 2** vs **Python 3**:
  - **Python 2** still has a huge number of users.
  - Officially, they suggest people to use **python 3**
  - **Python 3** is the future, it reduces nasty way to code.
  - But there stills a lot of well-written package for **python 2**.
  - So, learn both is the best option, you can use any of it, but keep it consistent.
  - Learn to use “\_\_future\_\_” package in **python2**.

# Check your python version

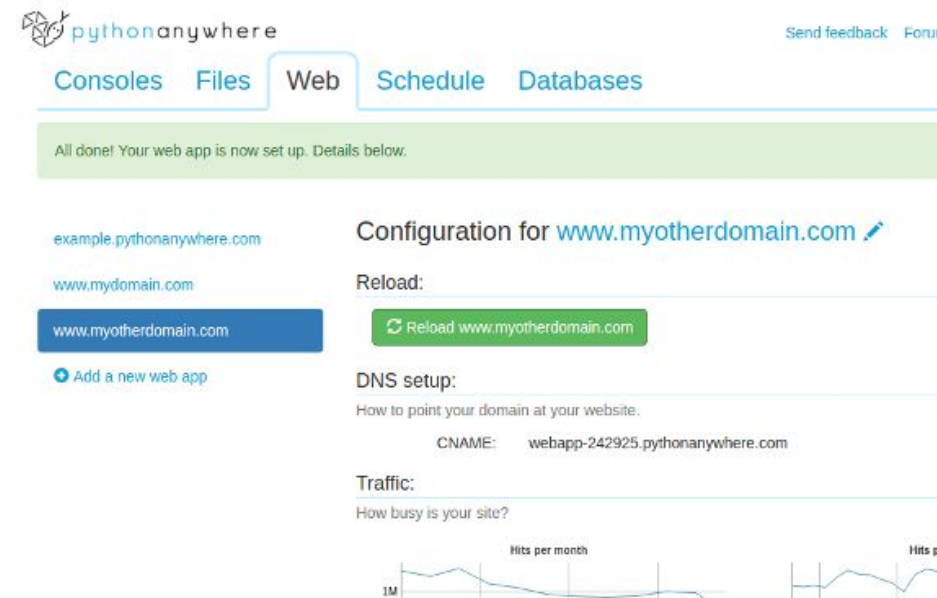
- You can always check the python version by running its interpreter.
- We use **python 2** in this workshop.

```
3: python2 ▼  
  
# yanyan @ vela in ~ [17:28:58]  
$ python2  
Python 2.7.13 (default, Dec 21 2016, 07:16:46)  
[GCC 6.2.1 20160830] on linux2  
Type "help", "copyright", "credits" or "license" for more information.  
>>> 
```

```
3: python3 ▼  
  
# yanyan @ vela in ~ [17:30:20]  
$ python3  
Python 3.6.0 (default, Jan 16 2017, 12:12:55)  
[GCC 6.3.1 20170109] on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> 
```

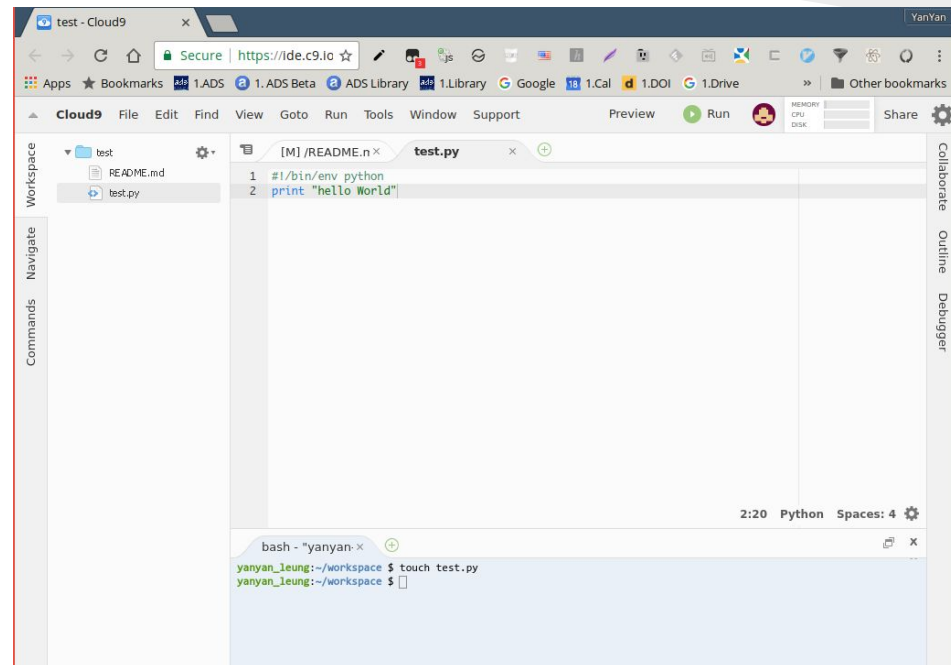
# Python in the cloud

<https://www.pythonanywhere.com/>



The screenshot shows the PythonAnywhere web interface. At the top, there's a navigation bar with 'Consoles', 'Files', 'Web', 'Schedule', and 'Databases'. Below this, a green banner states 'All done! Your web app is now set up. Details below.' The main content area is titled 'Configuration for www.myotherdomain.com'. It includes a 'Reload' section with a green button to 'Reload www.myotherdomain.com'. Below that is the 'DNS setup' section, which shows 'CNAME: webapp-242925.pythonanywhere.com'. The 'Traffic' section shows two line graphs for 'Hits per month' and 'Hits per day'.

<https://c9.io/>



The screenshot shows the Cloud9 IDE interface. The browser address bar shows 'https://ide.c9.io'. The interface includes a file explorer on the left showing a 'test' directory with 'README.md' and 'test.py'. The main editor area shows the content of 'test.py', which contains a Python script: 

```
#!/bin/env python
print "hello World"
```

. The bottom panel shows a terminal window with the following commands and output: 

```
bash - "yanyan"
yanyan_leung:~/workspace $ touch test.py
yanyan_leung:~/workspace $
```

# Very subtle things about package!

*How to add packages to python?*

# Symbolic / numerical / statistical / machine learning

- For symbolic

- sympy



- For numerical

- numpy
- scipy



SymPy



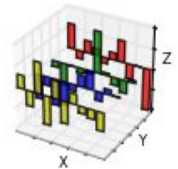
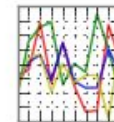
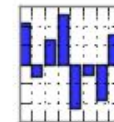
machine learning in Python

- For statistical and machine learning

- scikit-learn
- pandas

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



# Python plotting / visualising

- **matplotlib**
  - all-round, major plotting in python
- **aplpy**
  - FITS image plotting in high quality
- **yt**
  - large data / volumetric data visualising
- **bokeh**
  - interactive plots in html & javascript

**matplotlib**





# Install package (Anaconda)

- To search/install packages, use “**conda**”
- Command:
  - `conda search xxxxxx`
  - `conda install xxxxxx`
- Other commands:

[http://conda.pydata.org/docs/\\_downloads/conda-cheatsheet.pdf](http://conda.pydata.org/docs/_downloads/conda-cheatsheet.pdf)

# Install package (pip)

- To search/install packages, use “pip”
- Package list: <https://pypi.python.org/pypi>
- Command:
  - `pip search xxxxxx`
  - `pip install xxxxxx`

## Useful Commands:

<code>install</code>	Install packages.
<code>uninstall</code>	Uninstall packages.
<code>freeze</code>	Output installed packages in requirements format.
<code>list</code>	List installed packages.
<code>show</code>	Show information about installed packages.
<code>search</code>	Search PyPI for packages.
<code>zip</code>	Zip individual packages.
<code>unzip</code>	Unzip individual packages.
<code>bundle</code>	Create pybundles.
<code>help</code>	Show help for commands.

# I don't have pip :(

- <https://pip.pypa.io/en/stable/installing/>
- Linux user:

[https://packaging.python.org/install\\_requirements\\_linux/#installing-pip-setuptools-wheel-with-linux-package-managers](https://packaging.python.org/install_requirements_linux/#installing-pip-setuptools-wheel-with-linux-package-managers)

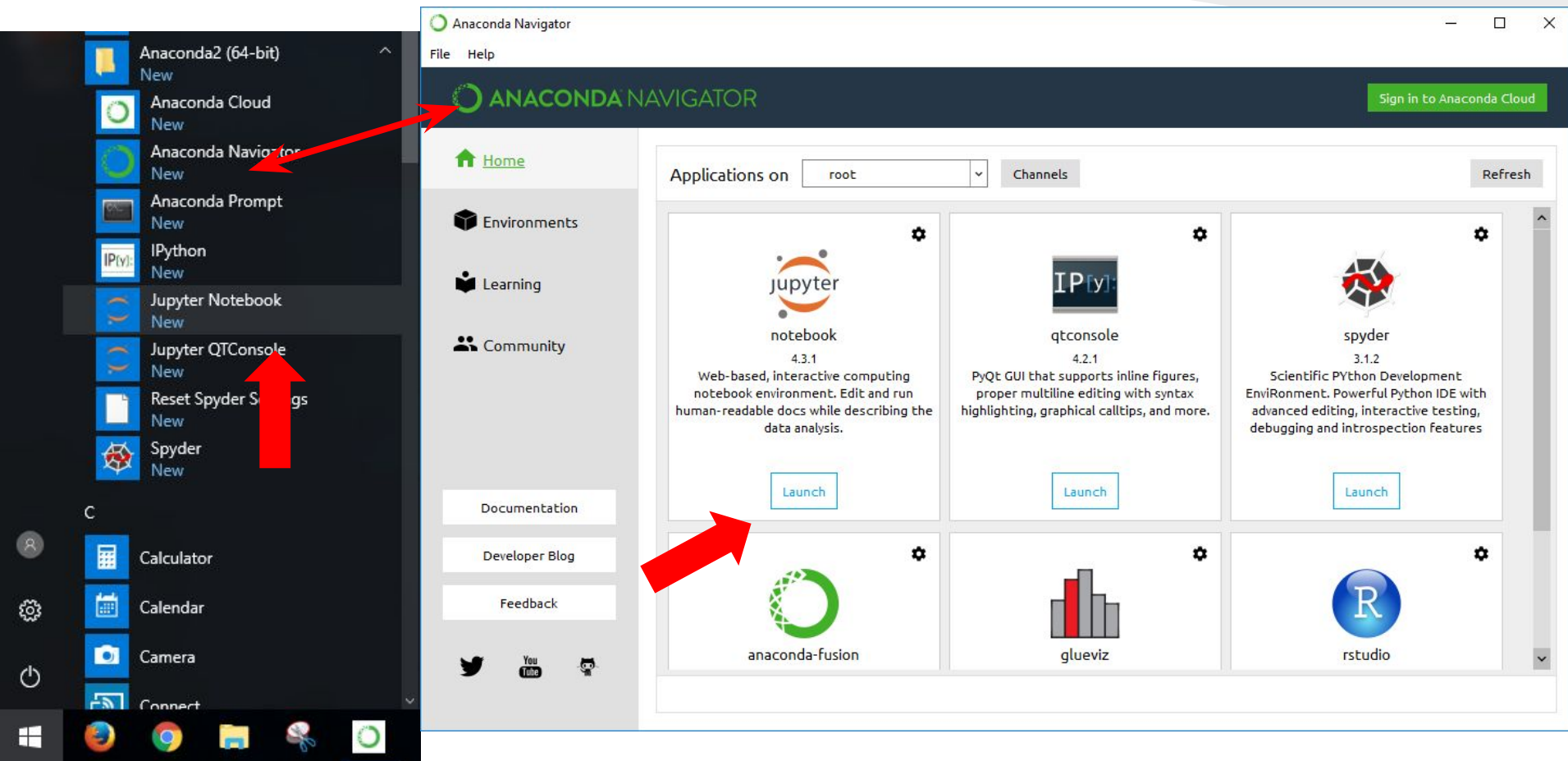
## Fedora

- Fedora 21:
  - Python 2:
    - **sudo yum upgrade python-setuptools**
    - **sudo yum install python-pip python-wheel**
  - Python 3:
    - **sudo yum install python3 python3-wheel**
- Fedora 22:
  - Python 2:
    - **sudo dnf upgrade python-setuptools**
    - **sudo dnf install python-pip python-wheel**
  - Python 3:
    - **sudo dnf install python3 python3-wheel**

# Finally, it's time to run a python script

*How to run your scripts*

# Open jupyter in Windows



# Open jupyter in linux / mac

```
# yanyan @ vela in ~/workspace [12:37:07]
$ jupyter notebook

# yanyan @ vela in ~/workspace [12:37:07]
$ jupyter notebook
[I 12:38:14.082 NotebookApp] Serving notebooks from local directory: /home/yanya
n/workspace
[I 12:38:14.082 NotebookApp] 0 active kernels
[I 12:38:14.082 NotebookApp] The Jupyter Notebook is running at: http://localhos
t:8888/?token=40a1e1aa7783bb15e5178ec870a0f8bb07470e94d0a02da0
[I 12:38:14.082 NotebookApp] Use Control-C to stop this server and shut down all
kernels (twice to skip confirmation).
[C 12:38:14.083 NotebookApp]

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
    http://localhost:8888/?token=40a1e1aa7783bb15e5178ec870a0f8bb07470e94d0a
02da0
Gtk-Message: Failed to load module "pk-gtk-module"
Created new window in existing browser session.
[I 12:38:15.159 NotebookApp] Accepting one-time-token-authenticated connection f
rom ::1
```

# Part 2

## Basic Python

# Running a python script

- You can always check the python version by running its interpreter.
- We use **python 2** in this workshop.
- A common *shebang* line used for the Python interpreter is as follows:
  - `#!/usr/bin/env python`
- You must then make the script executable, using the following command:
  - `chmod +x xxxxxxxxxx.py`



# Python interpreter

- For Linux / OSX, type “python” in terminal.
- For windows, open “Anaconda” folder in Start menu.

```
python
# yanyan at vela in ~ [17:34:44]
$ python
Python 2.7.11 |Anaconda 2.3.0 (64-bit)| (default, Dec 6 2015, 18:08:32)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>> 
```

```
python
Deactivating environment "D:\Apps\Anaconda"...
Activating environment "D:\Apps\Anaconda"...

[Anaconda] C:\Users\yanyan>python
Python 2.7.11 |Anaconda 2.4.1 (32-bit)| (default, Dec 7 2015, 14:13:17) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>>
```

# Spyder IDE



Like Matlab

Spyder (Python 3.4)

File Edit Search Source Run Debug Consoles Tools View Help

Editor - /tmp/interpolation.py

```
4 From the SciPy Cookbook
5 """
6
7 from numpy import arange, cos, linspace, pi, sin, random
8 from scipy.interpolate import splprep, splev
9
10 # make ascending spiral in 3-space
11 t=linspace(0,1.75*2*pi,100)
12
13 x = sin(t)
14 y = cos(t)
15 z = t
16
17 # %% add noise
18 x+= random.normal(scale=0.1, size=x.shape)
19 y+= random.normal(scale=0.1, size=y.shape)
20 z+= random.normal(scale=0.1, size=z.shape)
21
22 # %% spline parameters
23 s=3.0 # smoothness parameter
24 k=2 # spline order
25 nest=-1 # estimate of number of knots needed (-1 = maximal,
26
27 # %% find the knot points
28 tckp,u = splprep([x,y,z],s=s,k=k,nest=-1)
29
30 # %% evaluate spline, including interpolated points
31 xnew,ynew,znew = splev(linspace(0,1,400),tckp)
32
33 import pylab
```

Object inspector

Source Console Object numpy.mean

mean

**Definition:** mean(a, axis=None, dtype=None, out=None, keepdims=False)

**Type:** Function of numpy.core.fromnumeric module

Compute the arithmetic mean along the specified axis.

Returns the average of the array elements. The average is

Object inspector Variable explorer File explorer Static code analysis

IPython console

IP: Console 1/A

Python 3.4.0 on linux -- IPython 4.0.0

In [1]: runfile('/tmp/interpolation.py', wdir='/tmp')

Internal console Console History log IPython console

Permissions: RW End-of-lines: LF Encoding: UTF-8 Line: 18 Column: 43 Memory: 86 %

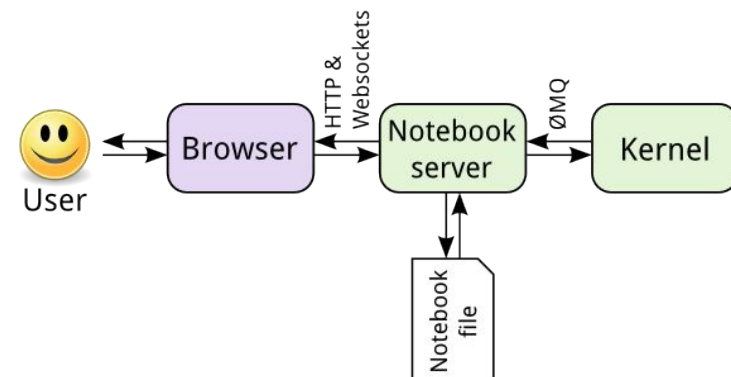
# ipython / jupyter

Like Maple /  
Mathematica!

For more interaction and fun, we use *ipython* to run our python code. *ipython* is old name, new name is call jupyter.



[INSTALL](#) [PROJECT](#) [COMMUNITY](#) [DOCUMENTATION](#) [NBVIEWER](#) [BLOG](#) [DONATE](#)



Open source, interactive data science and scientific computing across over 40 programming languages.

# Launch ipython for testing purpose

Lazy? <https://try.jupyter.org/>

But missing some packages! Better run your program in your own computer

 jupyter

Hosted by Rackspace 

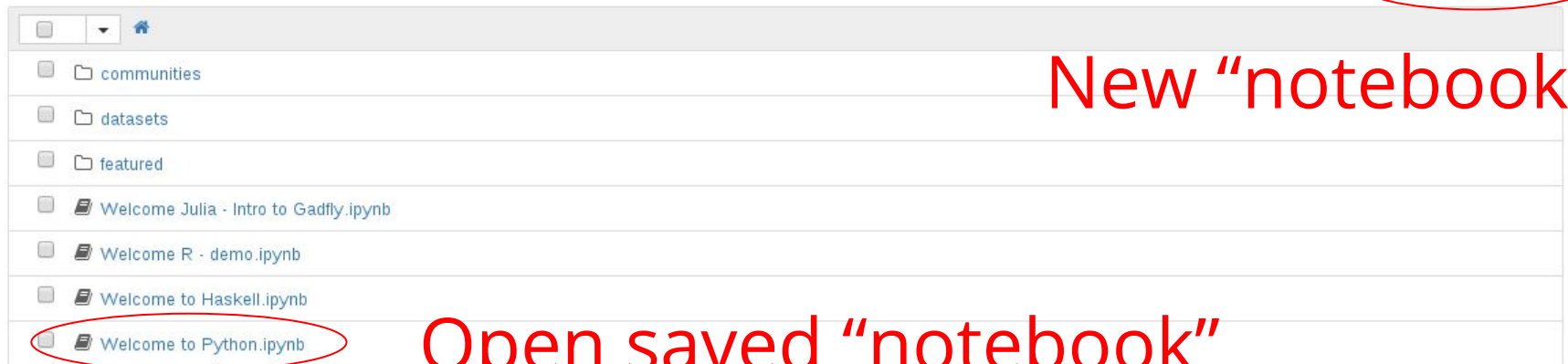
Files Running Clusters

Select items to perform actions on them.

Upload New 

New “notebook”

Open saved “notebook”



The screenshot shows the Jupyter web interface. At the top, there are tabs for 'Files', 'Running', and 'Clusters'. Below the tabs, there is a header bar with 'jupyter' on the left and 'Hosted by Rackspace' on the right. The main area is a file browser. It has a search bar and a home icon. Below that, there is a list of files and folders. The files are: 'communities', 'datasets', 'featured', 'Welcome Julia - Intro to Gadfly.ipynb', 'Welcome R - demo.ipynb', 'Welcome to Haskell.ipynb', and 'Welcome to Python.ipynb'. The 'Welcome to Python.ipynb' file is circled in red. In the top right corner of the file browser, there are three buttons: 'Upload', 'New', and a refresh icon. The 'New' button is also circled in red.

# Launch ipython in your computer

In command prompt/terminal, type  
`ipython notebook`

Go to <http://localhost:8888/> in browser

jupyter

Files Running IPython Clusters

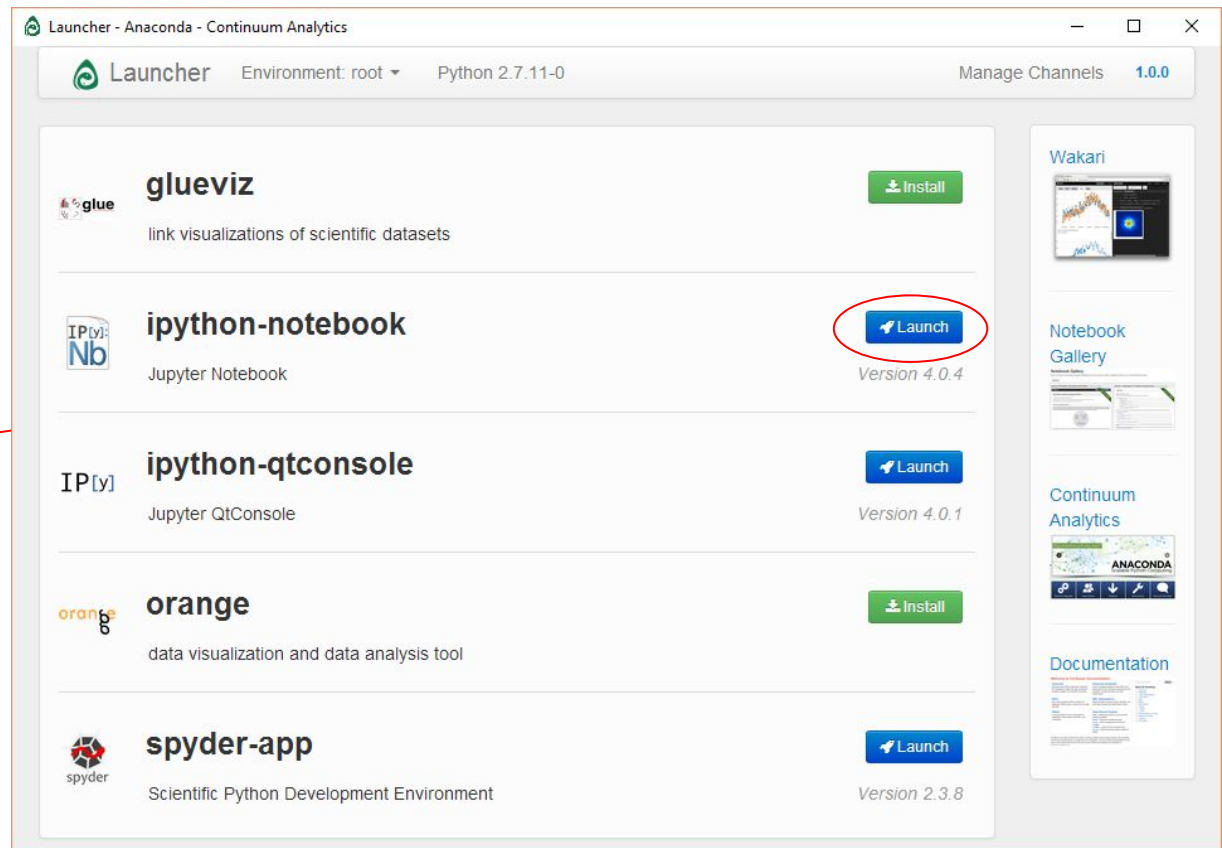
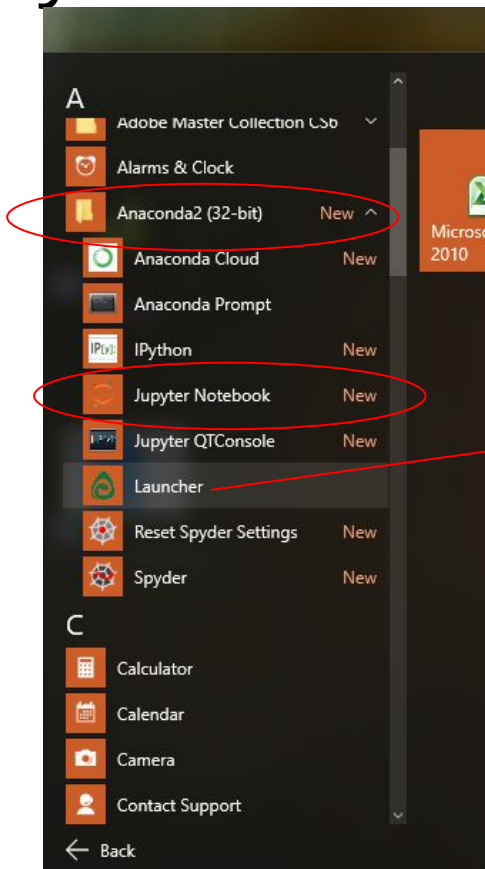
Select items to perform actions on them.



The screenshot shows the Jupyter Notebook web interface. At the top, there's a header with the Jupyter logo and the word 'jupyter'. Below it, there are tabs for 'Files', 'Running', and 'IPython Clusters'. The 'Files' tab is active, showing a file browser. The breadcrumb path is '/ research / python / 2016-JAN\_python\_workshop'. The file list includes: '..', 'Data Structures and Loops.ipynb', 'Files and Plot.ipynb', 'Functions.ipynb', 'Syntax.ipynb', 'Basic python.pdf', 'LICENSE.md', and 'README.md'. On the right side, there's a toolbar with 'Upload', 'New', and a refresh icon. The 'New' dropdown menu is open, showing options: 'Text File', 'Folder', 'Terminal', 'Notebooks', 'Julia 0.4.3', and 'Python 2'. The 'Python 2' option is circled in red.

# Launch ipython in your computer

If you have Anaconda GUI / using Windows, you can also launch here:




## Welcome to the Temporary Notebook (tmpnb) service!

This Notebook Server was **launched just for you**. It's a temporary way for you to try out a recent development version of the IPython/Jupyter notebook.

### WARNING

Don't rely on this server for anything you want to last - your server will be *deleted after 10 minutes of inactivity*.

Your server is hosted thanks to [Rackspace](#), on their on-demand bare metal servers, [OnMetal](#).

Hosted by Rackspace 

Files Running Clusters

Select items to perform actions on them.

Upload New ↕

<input type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>		communities
<input type="checkbox"/>		datasets
<input type="checkbox"/>		featured
<input type="checkbox"/>		Welcome Julia - Intro to Gadfly.ipynb
<input type="checkbox"/>		Welcome R - demo.ipynb
<input type="checkbox"/>		Welcome to Haskell.ipynb
<input type="checkbox"/>		Welcome to Python.ipynb
		Running

Opened/running  
notebook

Hosted by Rackspace 

Files Running Clusters

Currently running Jupyter processes

Terminals ▾

There are no terminals running.

Notebooks ▾

	Welcome to Python.ipynb
	Welcome to Haskell.ipynb
	Welcome Julia - Intro to Gadfly.ipynb

All running  
notebook

Shutdown  
button

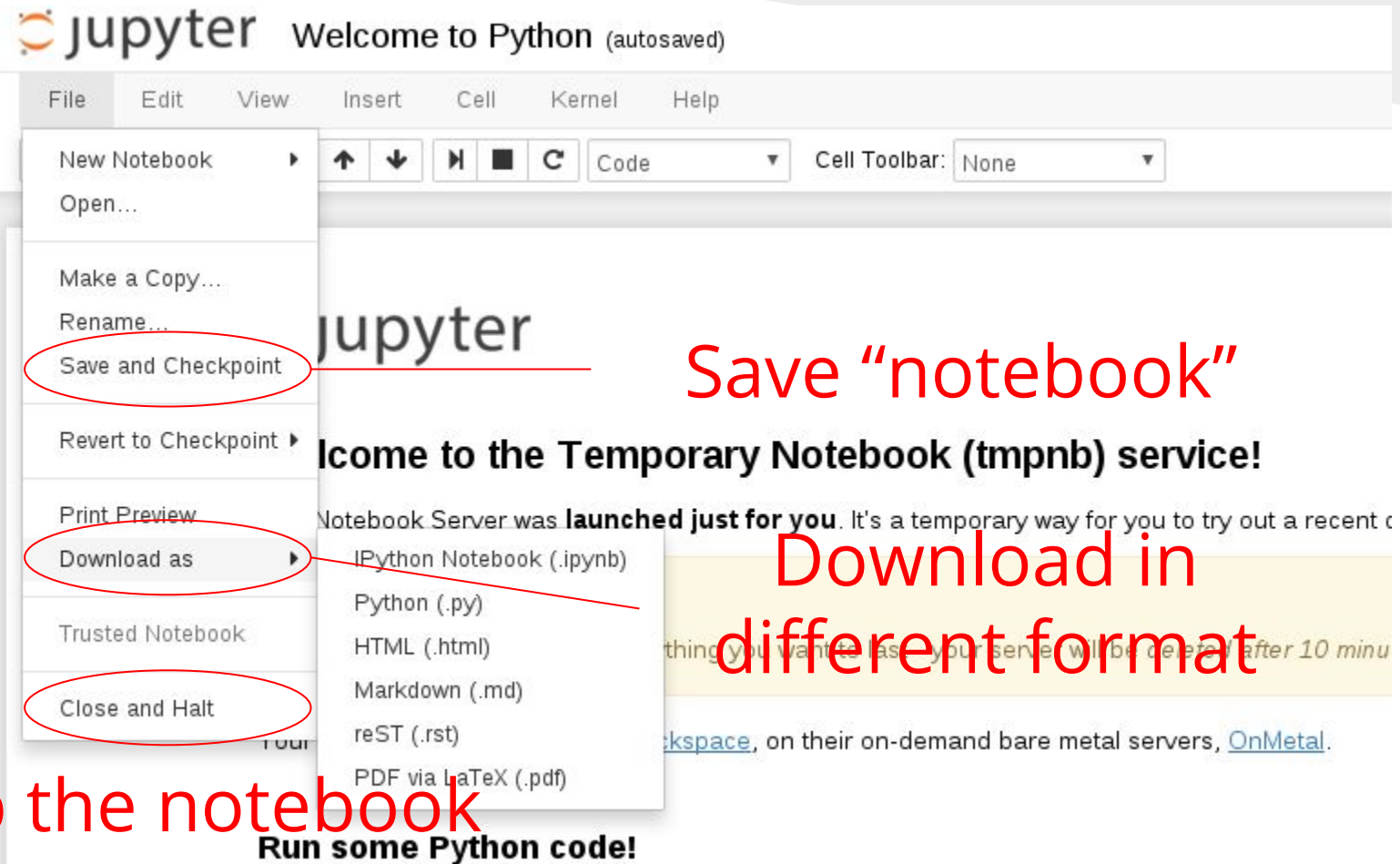
Shutdown

Shutdown

Shutdown

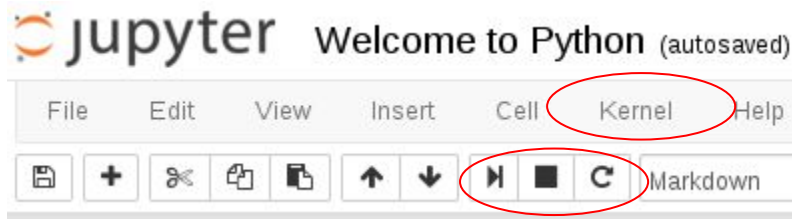


# ipython (save/download notebook)





# Notes on using ipython/jupyter



Something goes wrong?  
Stop the kernel

Stop the current  
cell operation

- To be safe, make sure you open each notebook document in only one tab.
- Data will be lost if the kernel is stopped.
- You can close the browser tab safely after the notebook says “notebook saved”. It will run in background

# Notes on using ipython/jupyter

## 3.1.1. Change Jupyter Notebook startup folder (Windows)

- Copy the *IPython Notebook* launcher from the menu to the desktop.
- Right click on the new launcher and change the “Start in” field by pasting the full path of the folder which will contain all the notebooks.
- Double-click on the *IPython Notebook* desktop launcher (icon shows [IPy]) to start the [Jupyter Notebook App](#), which will open in a new browser window (or tab). Note also that a secondary terminal window (used only for error logging and for shut down) will be also opened. If only the terminal starts, try opening this address with your browser: <http://localhost:8888/>.

## 3.1.2. Change Jupyter Notebook startup folder (OS X)

To launch [Jupyter Notebook App](#):

- Click on spotlight, type `terminal` to open a terminal window.
- Enter the startup folder by typing `cd /some_folder_name`.
- Type `ipython notebook` to launch the [Jupyter Notebook App](#) (it will appear in a new browser window or tab).

# 1. Syntax

*CH1 Syntax.ipynb*

# 2. Data Structures and Loops

*CH2 Data Structures and Loops.ipynb*

# 3. Functions

*CH3 Functions.ipynb*

# 4. Files operation

*CH4 File operations.ipynb*

# 5. Plots with matplotlib

*CH5 Plots.ipynb*

# 6. Advanced plotting

*CH6 Advanced Plot.ipynb*



# Python in astronomy



**astropy**

A Community Python Library for Astronomy

Until ~2012 python astronomy modules were scattered.  
Several core modules are now unified under astropy:

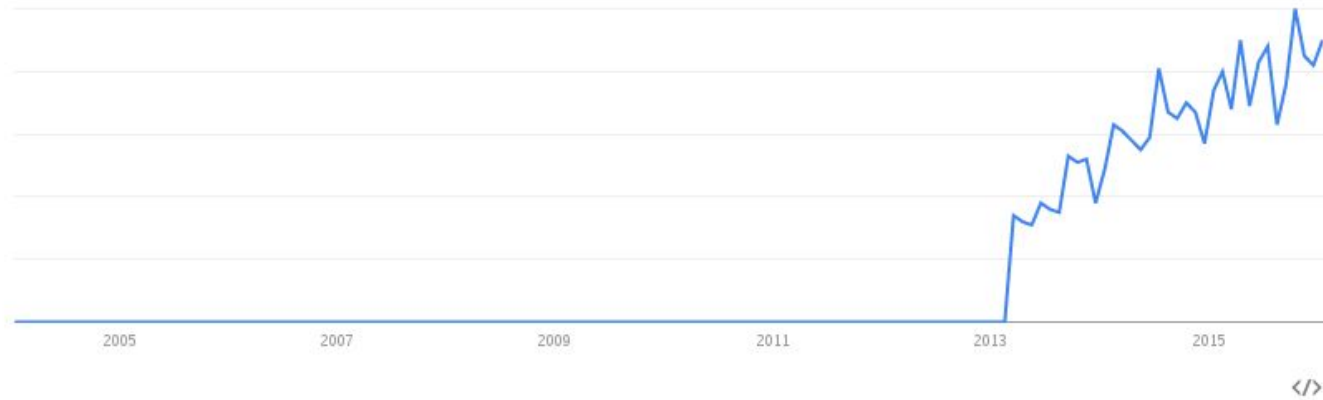
- **astropy.wcs** (World coordinate system (WCS) supported by PyWCS.)
- **astropy.io.fits** (FITS files support supported by PyFITS.)
- **astropy.coordinates** (Celestial coordinate and time transformations.)
- **astropy.units** (Unit and physical quantity conversions, physical constants specific to astronomy.)

**astropy**  
Search term

+ Add term

## Interest over time ?

☐ News headlines ? ☐ forecast ?



## Regional interest ?



# 7. Tools for astronomy

*Astropy - Load fits.ipynb*

*APLpy - Fits image & colour map.ipynb*

*APLpy - Cass A in 3 colours.ipynb*

# Need more performance?

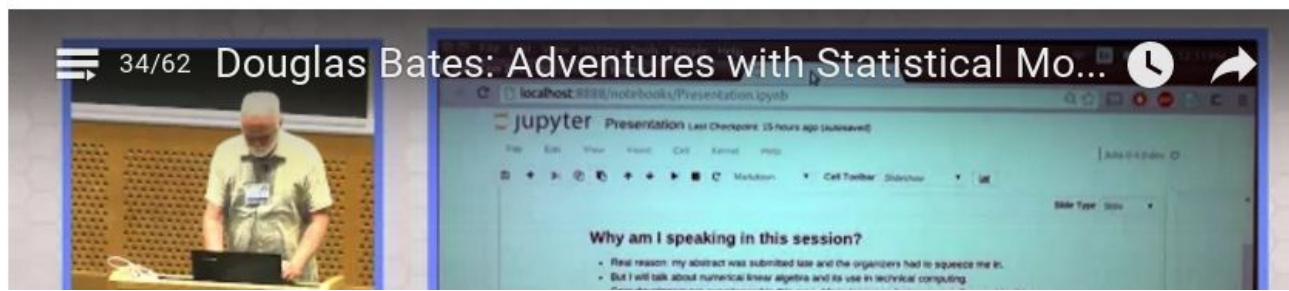
*Any language apart from python and have a great performance?*



[julia](#) | [source](#) | [downloads](#) | [docs](#) | [packages](#) | [blog](#) | [community](#) | [learning](#) | [teaching](#) | [publications](#) | [jsoc](#) | [juliacon](#)

Julia is a high-level, high-performance dynamic programming language for technical computing, with syntax that is familiar to users of other technical computing environments. It provides a sophisticated compiler, [distributed parallel execution](#), numerical accuracy, and an [extensive mathematical function library](#). Julia's Base library, largely written in Julia itself, also integrates mature, best-of-breed open source C and Fortran libraries for [linear algebra](#), [random number generation](#), [signal processing](#), and [string processing](#). In addition, the Julia developer community is contributing a number of [external packages](#) through Julia's built-in package manager at a rapid pace. [IJulia](#), a collaboration between the [IPython](#) and Julia communities, provides a powerful browser-based graphical notebook interface to Julia.

[JuliaCon 2015](#) at MIT was a huge success. The [videos](#) are now online, and a random video from JuliaCon 2015 is presented here.



# Some basic features in julia

- Syntax similarities: python, MATLAB and C
- Utilize matplotlib for plotting, clever and sweet
- Performance compatible to C & Java

	Fortran	Julia	Python	R	Matlab	Octave	Mathe- matica	JavaScript	Go	LuaJIT	Java
	gcc 4.8.2	0.3.7	2.7.9	3.1.3	R2014a	3.8.1	10.0	V8 3.14.5.9	go1.2.1	gsl- shell 2.3.1	1.7.0_75
fib	0.57	2.14	95.45	528.85	4258.12	9211.59	166.64	3.68	2.20	2.02	0.96
parse_int	4.67	1.57	20.48	54.30	1525.88	7568.38	17.70	2.29	3.78	6.09	5.43
quicksort	1.10	1.21	46.70	248.28	55.87	1532.54	48.47	2.91	1.09	2.00	1.65
mandel	0.87	0.87	18.83	58.97	60.09	393.91	6.12	1.86	1.17	0.71	0.68
pi_sum	0.83	1.00	21.07	14.45	1.28	260.28	1.27	2.15	1.23	1.00	1.00
rand_mat_stat	0.99	1.74	22.29	16.88	9.82	30.44	6.20	2.81	8.23	3.71	4.01
rand_mat_mul	4.05	1.09	1.08	1.63	1.12	1.06	1.13	14.58	8.45	1.23	2.35

- IJulia and Jupyter

Home

Untitled1

localhost:8888/notebooks/Untitled1.ipynb?kernel\_name=julia-0.4

John

Julia 0.4.2

jupyter

Untitled1 (unsaved changes)

File

Edit

View

Insert

Cell

Kernel

Help

Code

CellToolbar

Out[2]:

PyObject <matplotlib.collections.PathCollection object at 0x7f6e10c3e9b0>

In [4]:

?repmat

search:

Out[4]:

repmat(A, n, m)

Construct a matrix by repeating the given matrix n times in dimension 1 and m times in dimension 2.

repmat

In [ ]:

If  $A_1, A_2, \dots$  are subsets of  $X$ , then  $\bigcup_n A_n$  is the set  $\{x \in X \mid x \in A_n \text{ for some } n \in \mathbb{N}\}$

In [ ]:

# That's all

*You can stay here for practising, I have collected some programming tasks for you*



# Remember to practise at home

*Questions are obtained from <https://projecteuler.net/>*

# Some sample data sets are included

There are many datasets under the “datasets” folder

Load any one of them and try to plot them out :)

Remember you can download the file by:

```
import urllib  
urllib.urlretrieve("URL", "yourfilename")
```

Or you create a text file and put all the data into it.