Why  python™ ?

Welcome to the Python Meeting group!

1. Brief introduction to Python for those that are new to it
    - what and why…

2. Plans for future meet ups
    - biweekly?
    - platform to discuss problems/ideas
    - introduce development environments you like
    - distributions you prefer
    - packages that you swear by

3. GDAL for Python (raster and vector processing)
    - raster i/o walk through
    - vector i/o and polygon development

What is  python™

and why should you use it…

- Interpreted, object-oriented, high-level programming language
- **Free!**
- Portable
  - works on OS X, Linux, Windows

- Designed to be easy to program and easy to read
- Because of it's relatively simple syntax, easy to translate ideas to code
- Easy to design functions
  - they don't have to be put anywhere specific if you don't want
- Encourages collaboration

"If you can easily discuss your code with others in your office, the result can be better code and better coders"

This is a ***docstring*** – it comes up if you query a function name on the python command line
e.g. > ?classify

Your function

```
def classify (values, boundary):
        '''Classifies values as being below (False) or above (True) a boundary'''
        return [(True if v > boundary else False) for v in values]
```

Your variable

```
#Create a list
numbers=[1.0, 4.2, 5.0, 0.7, -1.4]
```

Pass variable to function

```
#Call the function
boolean=classify(numbers, boundary=0.5)

####################################
```

Get your output

```
#boolean is then equal to :
[True, True, True, True, False]
```

Lots of different text editors/IDEs are available and automated colouring of syntax sometimes helps!

Taken from: https://www.stat.washington.edu/~hoytak/blog/whypython.html

This is a ***docstring*** – it comes up if you query a function name on the python command line
e.g. > ?classify

Your function

```
def classify (values, boundary):
        '''Classifies values as being below (False) or above (True) a boundary'''
        return [(True if v > boundary else False) for v in values]
```

Your variable

```
#Create a list
numbers=[1.0, 4.2, 5.0, 0.7, -1.4]
```

Pass variable to function

```
#Call the function
boolean=classify(numbers, boundary=0.5)

####################################
```

Get your output

```
#boolean is then equal to :
[True, True, True, True, False]
```

Sometimes coloured syntax over complicates things and/or doesn't work for everybody…

Taken from: https://www.stat.washington.edu/~hoytak/blog/whypython.html

- Best seen as a ***glue*** type language
- Programs written in C, C++ and Java generally run much faster
- Despite having faster run times, Python is **much faster to write**
- Even if you don't know lower level languages like C or Java, Python is **easy to learn**
- Check out **learnpythonthehardway.org**

- Useful for *gluing* other languages together
- Possible to use packages and functions from other languages

  e.g. call in functions from matlab using MLabWrap

  e.g. call in functions from R using RPy2

Shell interaction

- Easy to use shell commands from within the script using subprocess

  e.g. list directories as you would in a shell (linux)

```
import subprocess
subprocess.call(['ls', '-l'], shell=True)
```

  e.g. use gdal tools as normal from the shell from inside a python script

```
import subprocess
subprocess.call('gdalinfo velocity.asc', shell=True)
```

Working with GIS data

Interfacing with Python for GIS…

- QGIS - have a look at the cookbook
- ArcGIS

Third party libraries

A major benefit of using Python – there are many (*read: loads and loads*) of third party libraries available

Key libraries:

`NumPy`	*"The fundamental package for scientific computing"* Very efficient array manipulation
`Scipy`	Additional routines for various scientific problems Functions for signal processing, image processing, linear algebra…
`Matplotlib`	Plotting library creating publication quality figures Just like MATLAB
`OSGeo`	Various geospatial processing tools (discussed further later on)
`Pandas`	Development and use of easy-to-use data structures

Python vs MATLAB

If you don't know either, learn Python – portable and a large community

If you know MATLAB, be aware of Python and consider it for the future

Wrappers are available to use Matlab specific things in Python – MLabWrap

Both have extensive libraries – Python has more!

Python can be used for development Matlab was not designed for
- for example useful for interacting with different databases and http server interfacing

Matplotlib offers much of the plotting capability of MATLAB

Python vs IDL

IDL is very good for what it is made for

BUT it is licensed = **EXPENSIVE**

You have to step out of IDL or write your own functions for advanced analysis

If you don't know either, learn Python – open source therefore super portable!

If you know IDL, be aware of Python and consider it for the future

- Scipy has tools to read in IDL products - check out scipy.io.readsav
- Projections are well supported
  - Proj4 syntax easily implemented
  - EPSG codes can be used
- Spatial libraries available including GDAL and Basemap

Python vs R

The bread and butter of statistics

R: is widely used – has numerous packages – has efficient structuring – is open source

Python can be run from within R (using the rPython package)

R can be used from within Python (using the RPy2 package)

Python arguably more readable with a less sharp learning curve (if you're brand new to it)

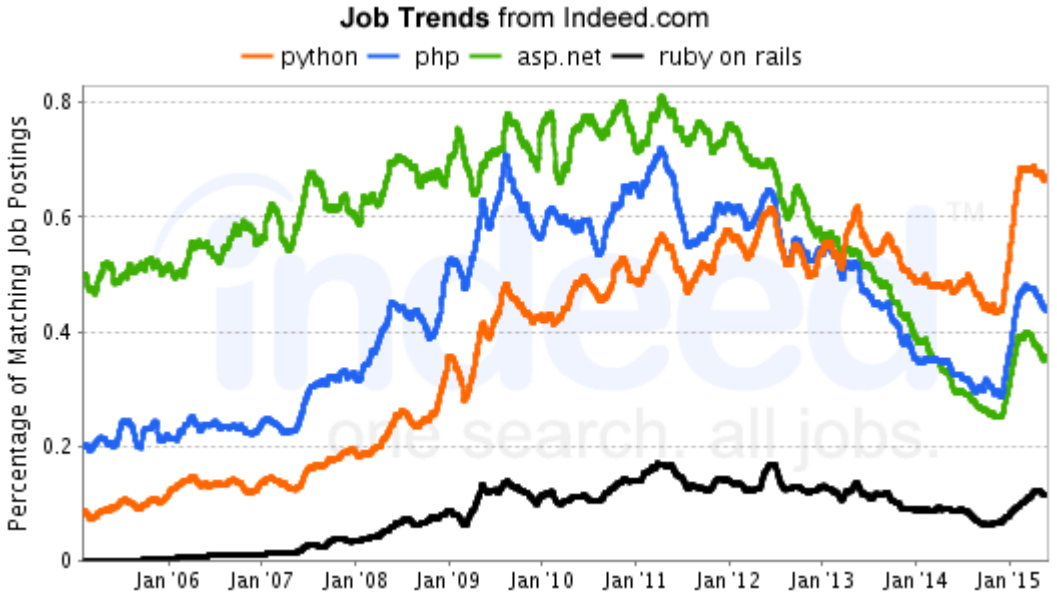
Nicer syntax of Python makes debugging easier

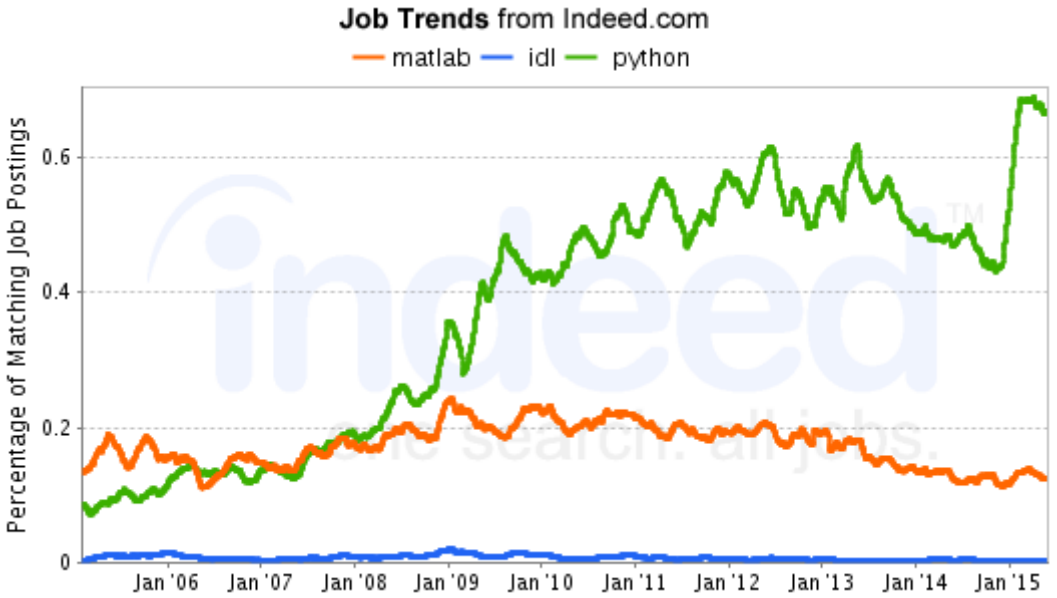
Python is better for general development (websites etc.) < this is not what R was made for

Read more about The Python vs R war

Jobs!

*…because it's **free and portable**, you can take all of your functions away with you without worries about lack of licenses!*





ource: http://www.indeed.com/jobtrends?q=python%2C+php%2C+asp.net%2C+ruby+on+rails&l=