Astronomical data analysis using Python

Lecture 1

Yogesh Wadadekar

NCRA-TIFR

Nov-Dec 2021

What this course is about?

- prerequisites - no previous programming experience needed, but conceptual understanding will be easier if you have some prior experience in another language. We will be provinding all the slides used in lectures as well as edited videos via YouTube. If you find the pace of the course to be too fast (and you will, if you are an absolute beginner to programming), please review the slides and talk videos periodically. Practice the code snippets provided.

What this course is about?

- prerequisites - no previous programming experience needed, but conceptual understanding will be easier if you have some prior experience in another language. We will be provinding all the slides used in lectures as well as edited videos via YouTube. If you find the pace of the course to be too fast (and you will, if you are an absolute beginner to programming), please review the slides and talk videos periodically. Practice the code snippets provided.
- given the large number of participants in the course, I will only take questions at the end of each lecture. We will also use the Moodle forum for interactions.

What I will teach you in this course (10 lectures of 1 hour each + at least 2 tutorials

- the core Python language (6 lectures)
- how to use Python for numerical computing (1 lecture)
- how to use Python for making plots (1 lecture)
- how to use Python for astronomy specific data analysis using astropy and astroquery. (2 lectures)

A bit about myself

I am a faculty member at the National Centre for Radio Astrophysics, TIFR in Pune, India. I taught myself Python in late 2001, when I was a postdoc. I have used Python extensively for astronomical data analysis since 2003. I have taught an introductory Python programming course four times between 2009-2014.
This course represents my first attempt to teach Python programming online.

NCRA • TIFR

Acknowlegements

Figure: Dr. Kaustubh Vaghmare (left) and Dr. Preetish K. Mishra (right)

NCRA • TIFR

My slice of Python

The ultimate goal of this course is to introduce you to *astronomical data analysis* using Python. Due to the limited contact hours in this course, we will take the most direct path to the ultimate goal. This means I will consciously *ignore most aspects of the language that are not relevant to data analysis*

Only the final few lectures will focus on specific usage of Python in astronomy data analysis. If you are planning to use Python for data analysis in other domain areas, much of the course will still be useful for you.

Usage of Python has exploded in data science (AI/ML) applications. *This course will not be covering those aspects.*

Programming is a craft - it requires doing

Computer programming requires practice. If you know a programming language without using it, your knowledge will soon rust away.
Given the number of participants in this course, it becomes difficult to implement this idea. So the onus is on you!
From the second lecture onwards, I will share with you all the code we use in class. Please use it to clarify your understanding and to practice your skills. And after the course is over, please continue to study and modify other people's code and then start to write your own. There is no shortcut to writing good programs!

Installing Python on your computer

https://realpython.com/installing-python/
The above webpage provides detailed instructions on installing Python on Linux, Windows, and Mac-OSX. We do not have the resources to troubleshoot installation problems. Use online resources or consult a local expert if you face difficulties.
If you don't have Python installed, please spend the next few days to get it working on your machine.

Jupyter notebooks from JupyterLab

JupyterLab is an open-source web application that allows you to create and share documents that contain code, plots and narrative text. It works with many programming languages including, of course, Python. We will use Jupyter notebooks (like in Mathematica!) starting with the second lecture of this course. The notebooks for each lecture will be shared with you right after the lecture. To install JupyterLab on your computer, please follow the instructions at:
https://jupyterlab.readthedocs.io/en/stable/getting_started/installation.html

Google Colab - an online Cloud based Python notebook

Installing Python on your computer will give you the maximum and power and flexibility. However, if you are unable to install it for some reason, you can create online Jupyter notebooks using Google Colab at:
https://colab.research.google.com/
You can use Google Drive to save any notebooks you create with Google Collab. You can even load the notebooks for this course into Google Colab and then modify and run them there.
For data analysis of large datasets, you will need a local installation.

Why Python?

- A powerful, general purpose programming language, yet easy to learn. Strong, but optional, *object Oriented programming* support
- Very large user and developer community, very extensive and broad library base
- Very extensible with C, C++, or Fortran, portable distribution mechanisms available
- Free; non-restrictive license; open source
- it is now the standard scripting language for data analysis
- very powerful array processing capabilities (*numpy*)
- extensive documentation - Many books and on-line documentation resources available (for the core language and its packages)

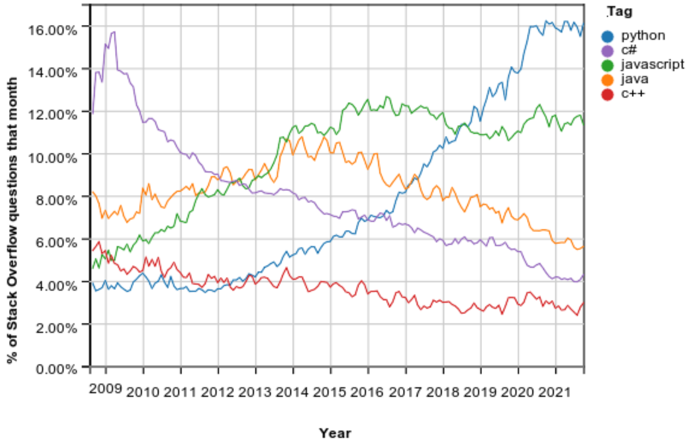
Why python?

- superb database interfaces to all popular databases.
- Clean code (very few non-alpha–numerics)
- **forced indentation** (back to old Fortran?)
- concise
- great for large teams
- Plotting is easy (and very easy if you know Matlab) using *matplotlib*
- Support for many widget systems for GUI development
- many other advantages which I have not listed

Disadvantages of Python

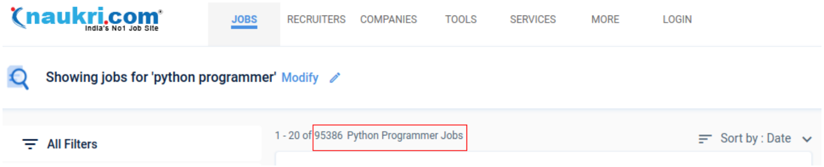
- More items to install separately (eased by prepackaged distributions like Anaconda and package management tools)
- Some specialised scientific libraries not as stable or fast as in Fortran
- but many Fortran libraries are wrapped: e.g. The NAG Library for Python is available through the *naginterfaces* package, with full access to the mathematical and statistical routines. The IMSL Python library allows access to IMSLroutines.
- Array indexing convention backwards, compared to Fortran
- Small array performance slower (eased greatly by *numpy*)

Python's popularity is exploding



Credit: Stack Overflow

Many job openings for Python programmers



Showing jobs for 'python programmer'

1 - 20 of 95386 Python Programmer Jobs

Credit: naukri.com

Python usage in optical astronomy

- STScI PyRAF (IRAF) + additional Python only routines
- ESO PyMIDAS (MIDAS)
- Astro-WISE (widefield imaging system)
- Pyephem - solar system ephemeris
- Rubin/LSST is using Python/C++ for their software stack

Python usage in Radio astronomy

- CasaPy (CASA) - AIPS++ based, default system for EVLA and ALMA data analysis. Many Python based data reduction pipelines for different telescopes use CASA.
- ParselTongue - call AIPS tasks from Python. SPAM uses ParselTongue.
- PYGILDAS (GILDAS) - IRAM data analysis software ported to Python
- APECS (APEX control software)
- KAT-7 Control and Monitoring System is in Python
- Presto - pulsar search and analysis suite; recent routines in Python

NCRA • TIFR

Python for scientists and engineers

- full featured, high level programming language
- very easy to learn – National Mission on Education through ICT sponsored a large program to develop computer education materials in Python for school and college students (http://python.fossee.in/).
- powerful text processing capabilities - many sysadmins have adopted it.
- powerful interfaces to almost any database
- web-friendly language - many packages available for controlling and accessing content on websites.
- good numerical computation capabilities
- good plotting capabilities
- most popular language for AI/ML researchers

Our course focused on using Python

for data analysis with special emphasis on astronomy.

Learning resources: Books

A search for “Python programming” on the books section of Amazon.in (as of Nov 2021) shows more than 7000 books.
Many of these are beginner level books. Which one you choose depends on your learning style and personal preferences. So, I will desist from making specific recommendations.
Many of the popular books have gone through multiple editions. Be sure to get the most recent one.

Online Material

- www.python.org

Online Material

- www.python.org
- Start with the Python tutorial - http://docs.python.org/tutorial/ we will cover all of it and more in this course.

Online Material

- www.python.org
- Start with the Python tutorial - http://docs.python.org/tutorial/ we will cover all of it and more in this course.
- SciPy conferences - http://conference.scipy.org - lots of interesting talks (many with video versions)

Online Material

- www.python.org
- Start with the Python tutorial - http://docs.python.org/tutorial/ we will cover all of it and more in this course.
- SciPy conferences - http://conference.scipy.org - lots of interesting talks (many with video versions)
- Also check out the variety of excellent Python programming courses on coursera and eDx. Lots of good videos on YouTube

NCRA • TIFR

Online Material

- www.python.org
- Start with the Python tutorial - http://docs.python.org/tutorial/ we will cover all of it and more in this course.
- SciPy conferences - http://conference.scipy.org - lots of interesting talks (many with video versions)
- Also check out the variety of excellent Python programming courses on coursera and eDx. Lots of good videos on YouTube
- Stack Overflow is a very useful question and answer website for all your Python questions
- http://python.fossee.in also has a number of tutorials.

NCRA • TIFR

Python Version 2 or 3?

Two major versions of Python are in widespread use. Python 2 and Python 3. Each has many sub-versions in use. **We will exclusively use Python 3 in this course**, since all major data analysis related packages have been ported to Python 3 and are now very stable.

Hello World program

```
$ python -c 'print ("Hello World")'
Hello World
```

Starting Python

simply type *python* at the command prompt

```
$ python3
Python 3.9.7 (default, Sep 10 2021, 14:59:43)
[GCC 11.2.0] on linux
Type "help", "copyright", "credits" or "license" for
more information.
>>>
```

Ipython - an enhanced python shell

simply type *ipython* at the command prompt.

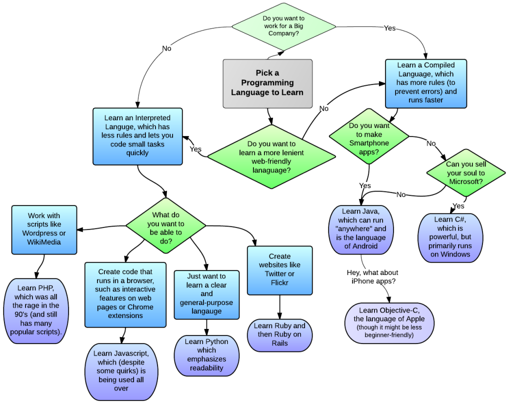
```
$ ipython3
Python 3.9.7 (default, Sep 10 2021, 14:59:43)
Type 'copyright', 'credits' or 'license' for more
information
IPython 7.20.0 - An enhanced Interactive Python.
Type '?'  for help.
In [1]:
```

Ipython is the shell for astronomy analysis packages - casapy and Pyraf.

Jupyter notebook - we will use this in our course!

Demo

Programming language flowchart


Do you want to work for a Big Company?
No
Yes
Pick a Programming Language to Learn
Learn a Compiled Language, which has more rules (to prevent errors) and runs faster
Learn an Interpreted Languge, which has less rules and lets you code small tasks quickly
Do you want to learn a more lenient web-friendly lanaguage?
Yes
No
Do you want to make Smartphone apps?
No
Can you sell your soul to Microsoft?
Yes
No
Yes
Learn Java, which can run "anywhere" and is the language of Android
Learn C#, which is powerful, but primarily runs on Windows
Hey, what about iPhone apps?
Learn Objective-C, the language of Apple (though it might be less beginner-friendly)
What do you want to be able to do?
Work with scripts like Wordpress or WikiMedia
Learn PHP, which was all the rage in the 90's (and still has many popular scripts).
Create code that runs in a browser, such as interactive features on web pages or Chrome extensions
Learn Javascript, which (despite some quirks) is being used all over
Just want to learn a clear and general-purpose langauge
Learn Python which emphasizes readability
Create websites like Twitter or Flickr
Learn Ruby and then Ruby on Rails

Credit: zappable.com

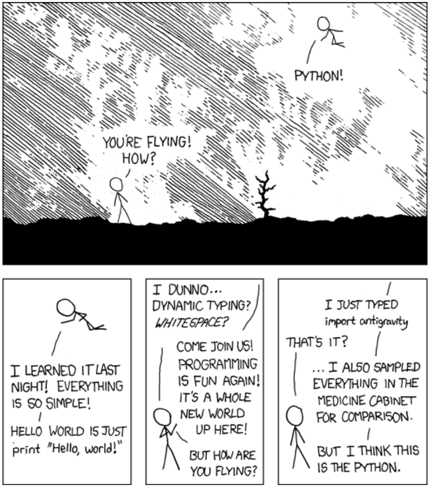
The Zen of Python, by Tim Peters *import this*

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one– and preferably only one –obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.

The Zen of Python, by Tim Peters *import this*

Although never is often better than *right* now.
If the implementation is hard to explain, it’s a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea – let’s do more of those!

That Python feeling


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

Credit: XKCD