Introduction to Python (for neuroscience)

NSP bootcamp
Daniel Denman
Assistant Professor, Department of Physiology and Biophysics

Plan for Thursday AM

- Introduction (45 min)
- Set up Python environments (5 - 15 min)
- Break
- Intro Jupyter notebook (with instructor) (X min)]

Plan for Thursday PM

- Independent data analysis notebooks (60 min).
- Intro to cloud-based Jupyter and team set up.
- Break
- Independent data analysis notebooks - collaboration in the cloud

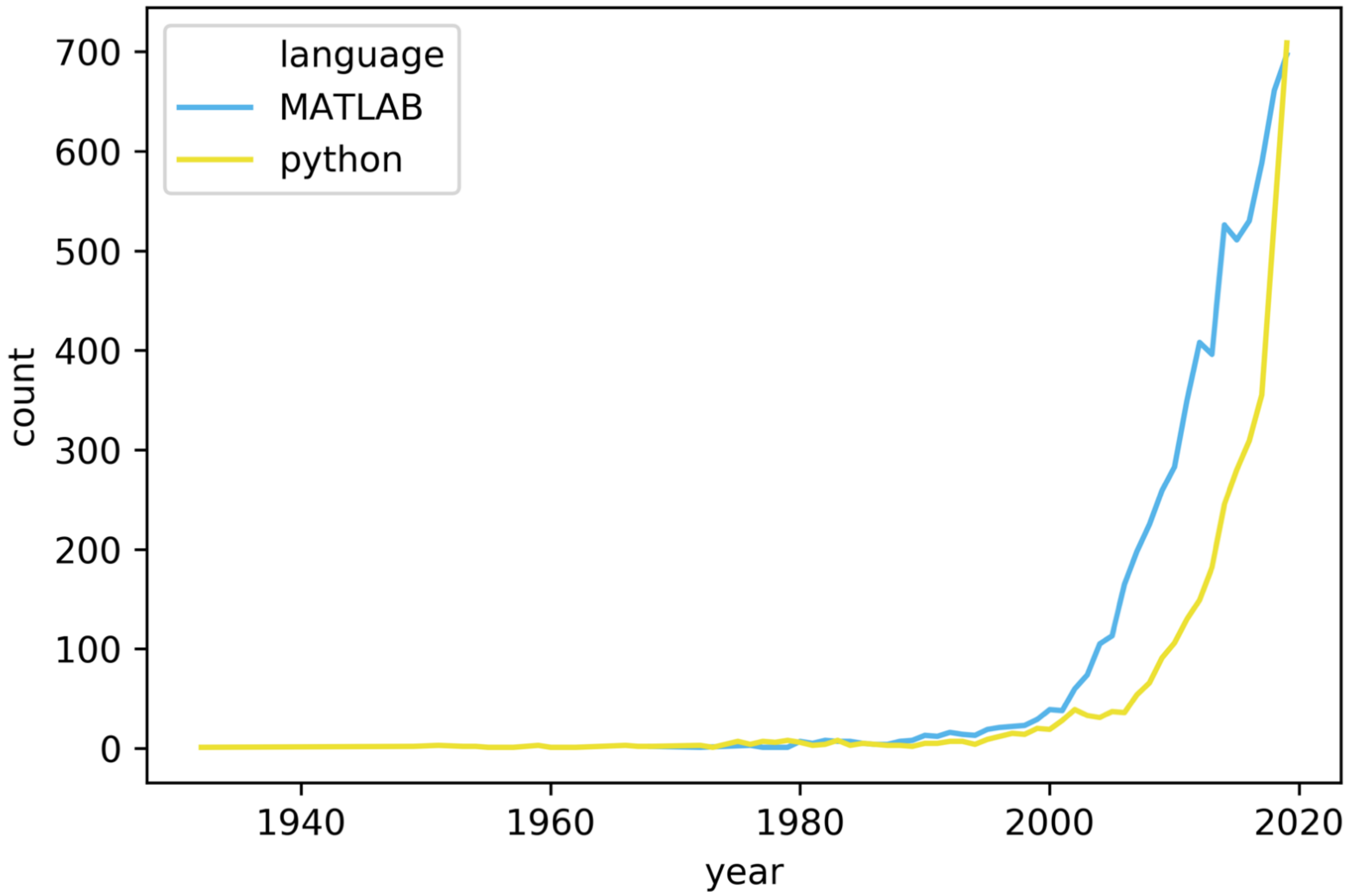
why code?

- Python or MATLAB? yes.
- Python was developed to make it easier for people to automate simple tasks ("scripting")
- You *could* do things by hand, but why not have a computer do it?

history: languages used for neuroscience

TIOBE index

history: languages used for neuroscience



TIOBE index

history: languages used for neuroscience

Volume 16 Issue 12, December 2019

nature methods
FOCUS ON DEEP LEARNING IN MICROSCOPY
Mapping epitranscriptomes without antibodies
Multimodal super-resolution microscopy
Inferring microbe-metabolite interactions
Nanopore sequencing of native RNA

Focus on Deep Learning in Microscopy

Artwork representing the application of deep learning methods in microscopy.

Image: National Institutes of Health/Stocktrek Images/Getty. Cover design: Erin DeWalt

TIOBE index

history: languages used for neuroscience

Volume 16 Issue 12, December 2019

nature methods
FOCUS ON DEEP LEARNING IN MICROSCOPY
Mapping epitranscriptomes without antibodies
Multimodal super-resolution microscopy
Inferring microbe–metabolite interactions
Nanopore sequencing of native RNA

Analysis | Published: 21 October 2019

Nucleus segmentation across imaging experiments: the 2018 Data Science Bowl

Juan C. Caicedo, Allen Goodman, Kyle W. Karhohs, Beth A. Cimini, Jeanelle Ackerman, Marzieh Haghighi, CherKeng Heng, Tim Becker, Minh Doan, Claire McQuin, Mohammad Rohban, Shantanu Singh & Anne E. Carpenter

Nature Methods **16**, 1247–1253(2019) | Cite this article

3257 Accesses | **1** Citations | **41** Altmetric | Metrics

2 out top 3 entries used python

TIOBE index

history: languages used for neuroscience

Volume 16 Issue 12, December 2019

nature methods
FOCUS ON DEEP LEARNING IN MICROSCOPY
Mapping epitranscriptomes without antibodies
Multimodal super-resolution microscopy
Inferring microbe-metabolite interactions
Nanopore sequencing of native RNA

Focus on Deep Learning in Microscopy

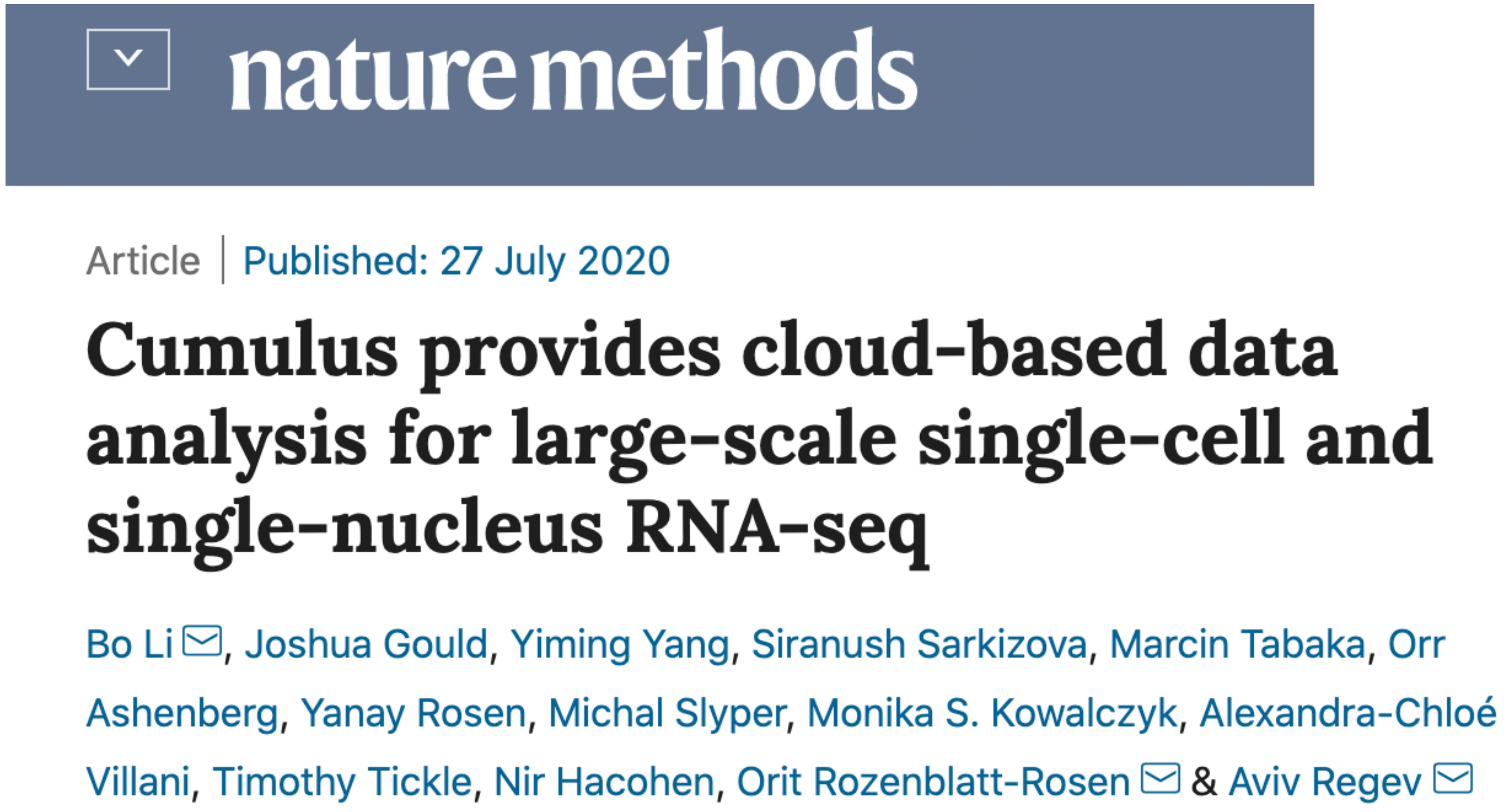
Artwork representing the application of deep learning methods in microscopy.

Image: National Institutes of Health/Stocktrek Images/Getty. Cover design: Erin DeWalt

5/7 software papers used python
1 used R
1 used ImageJ (Java)

TIOBE index

history: languages used for neuroscience


nature methods

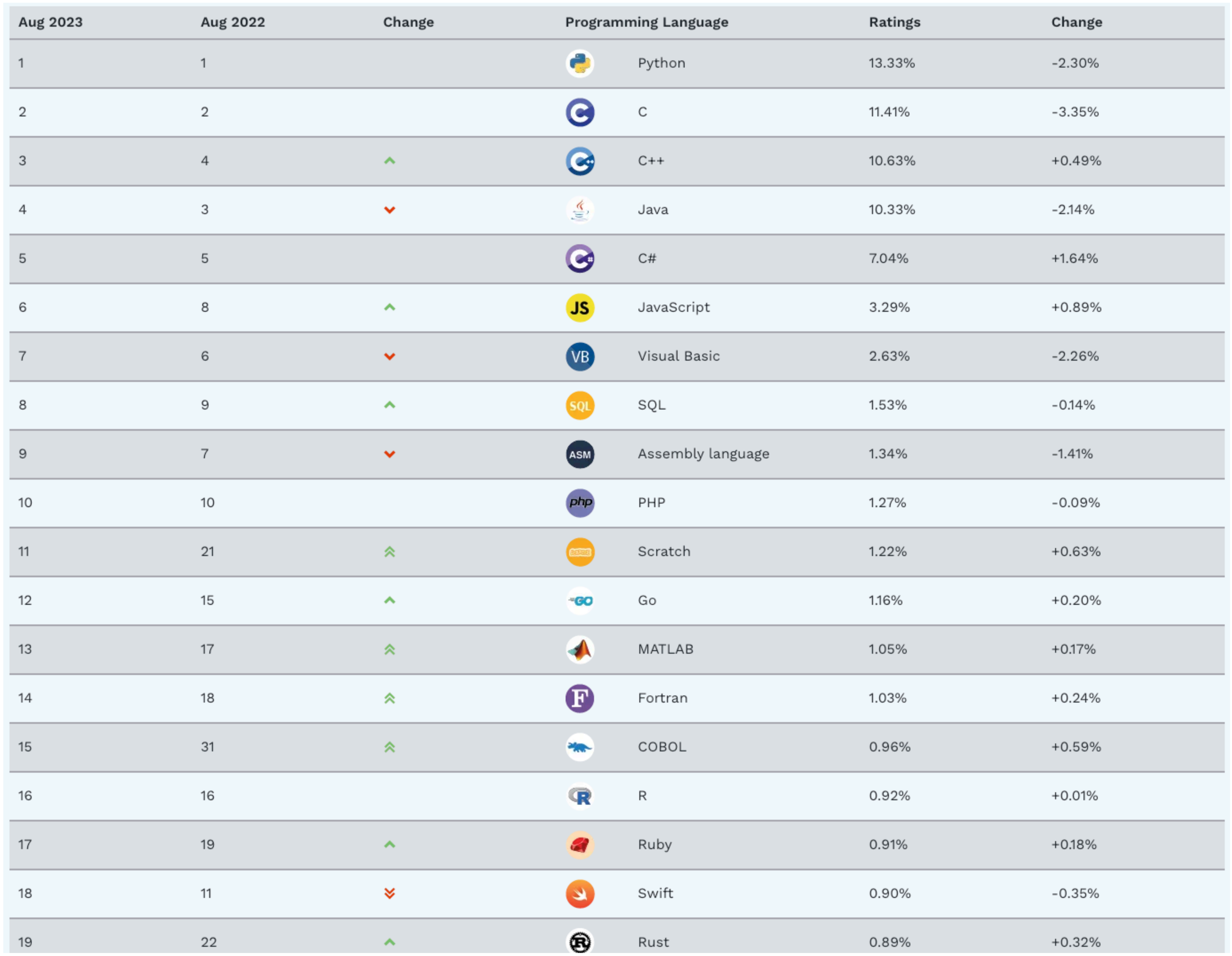
Article | Published: 27 July 2020

Cumulus provides cloud-based data analysis for large-scale single-cell and single-nucleus RNA-seq

Bo Li, Joshua Gould, Yiming Yang, Siranush Sarkizova, Marcin Tabaka, Orr Ashenberg, Yanay Rosen, Michal Slyper, Monika S. Kowalczyk, Alexandra-Chloé Villani, Timothy Tickle, Nir Hacohen, Orit Rozenblatt-Rosen & Aviv Regev

TIOBE index

history: languages used for neuroscience



Aug 2023	Aug 2022	Change	Programming Language	Ratings	Change
1	1		Python	13.33%	-2.30%
2	2		C	11.41%	-3.35%
3	4	▲	C++	10.63%	+0.49%
4	3	▼	Java	10.33%	-2.14%
5	5		C#	7.04%	+1.64%
6	8	▲	JavaScript	3.29%	+0.89%
7	6	▼	Visual Basic	2.63%	-2.26%
8	9	▲	SQL	1.53%	-0.14%
9	7	▼	Assembly language	1.34%	-1.41%
10	10		PHP	1.27%	-0.09%
11	21	▲▲	Scratch	1.22%	+0.63%
12	15	▲	Go	1.16%	+0.20%
13	17	▲▲	MATLAB	1.05%	+0.17%
14	18	▲▲	Fortran	1.03%	+0.24%
15	31	▲▲	COBOL	0.96%	+0.59%
16	16		R	0.92%	+0.01%
17	19	▲	Ruby	0.91%	+0.18%
18	11	▼▼	Swift	0.90%	-0.35%
19	22	▲	Rust	0.89%	+0.32%

TIOBE index

history

- First released in 1991
- A “scripting” or “high-level” language, designed for readability and productivity
  - simple syntax, use of white space

- Major release: Python 2.7, July 2010
- Use increases
- “data science” after era of “Big Data”
- Support for Python2.7 ended Jan 1, 2020

Guido van Rossum

history

- First released in 1991
- A “scripting” or “high-level” language, designed for readability and productivity
    - simple syntax, use of white space

- Major release: Python 2.7, July 2010
- Use increases
- “data science” after era of “Big Data”
- Support for Python2.7 ended Jan 1, 2020

Guido van Rossum

history

- First released in 1991
- A “scripting” or “high-level” language, designed for readability and productivity
  - simple syntax, use of white space

- Major release: Python 2.7, July 2010
- Use increases
- “data science” after era of “Big Data”
- Support for Python2.7 ended Jan 1, 2020

Guido van Rossum

history

- First released in 1991
- A “scripting” or “high-level” language, designed for readability and productivity
  - simple syntax, use of white space

- Major release: Python 2.7, July 2010
- Use increases
- “data science” after era of “Big Data”
- Support for Python2.7 ended Jan 1, 2020

Guido van Rossum
“Benevolent Dictator for Life”

history

- First released in 1991
- A “scripting” or “high-level” language, designed for readability and productivity
  - simple syntax, use of white space

- Major release: Python 2.7, July 2010
- Use increases
- “data science” after era of “Big Data”
- Support for Python2.7 ended Jan 1, 2020

Guido van Rossum
“Benevolent Dictator for Life”

but, but, biology…why code at all?

history

- First released in 1991
- A “scripting” or “high-level” language, designed for readability and productivity
  - simple syntax, use of white space

- Major release: Python 2.7, July 2010
- Use increases
- “data science” after era of “Big Data”
- Support for Python2.7 ended Jan 1, 2020

Guido van Rossum
“Benevolent Dictator for Life”

but, but, biology…why code at all?

- Python was developed to make it easier for people to automate simple tasks (“scripting”)

overview

Some plusses

Some minuses

overview

Some plusses

- **Free**
- **Readable syntax**
- **Cross platform**
- **Huge community**
- **Used across science ****and***** outside of science**

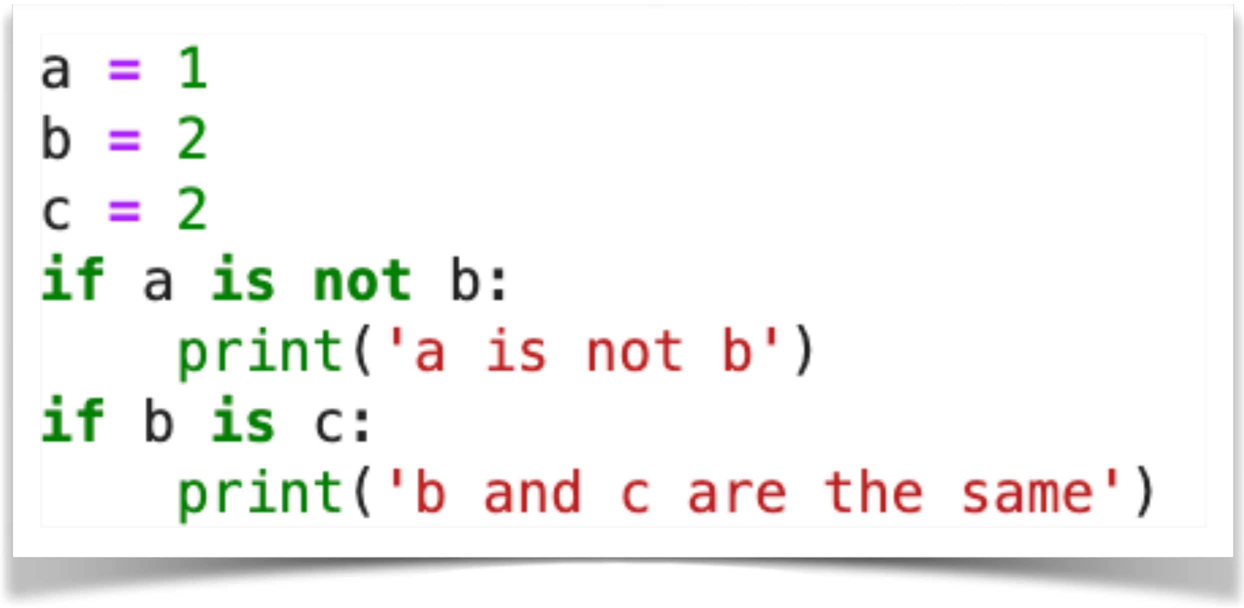
Some minuses

overview

Some plusses

- Free
- Readable syntax
- Cross platform
- Huge community
- Used across science **_and_** outside of science

Some minuses



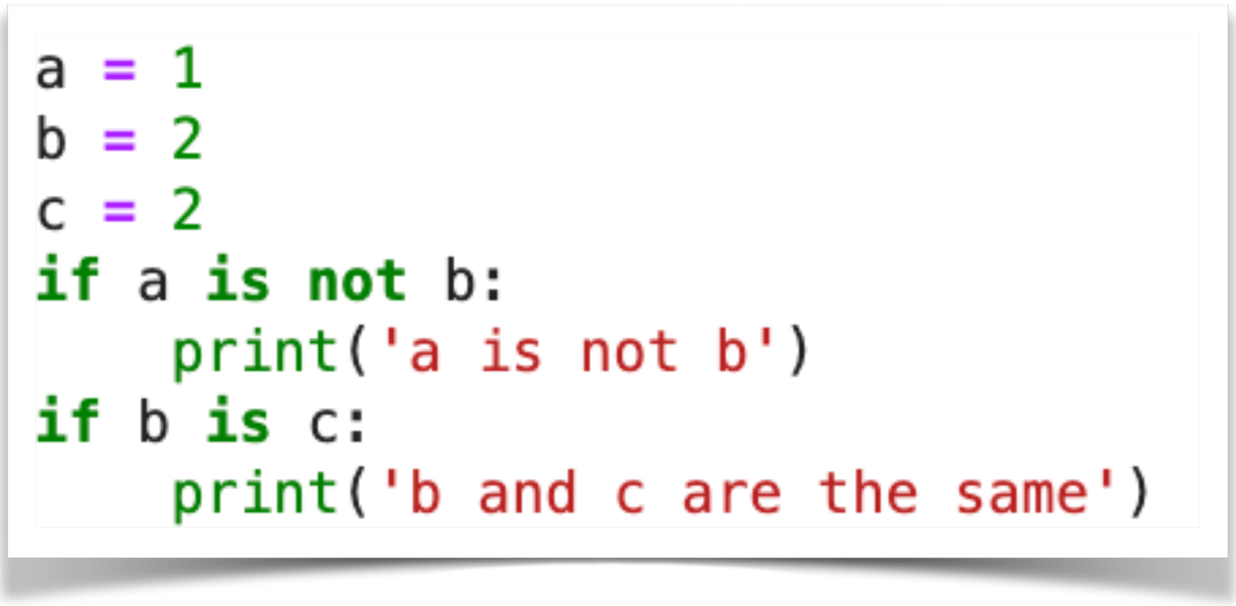
```
a = 1
b = 2
c = 2
if a is not b:
    print('a is not b')
if b is c:
    print('b and c are the same')
```

overview

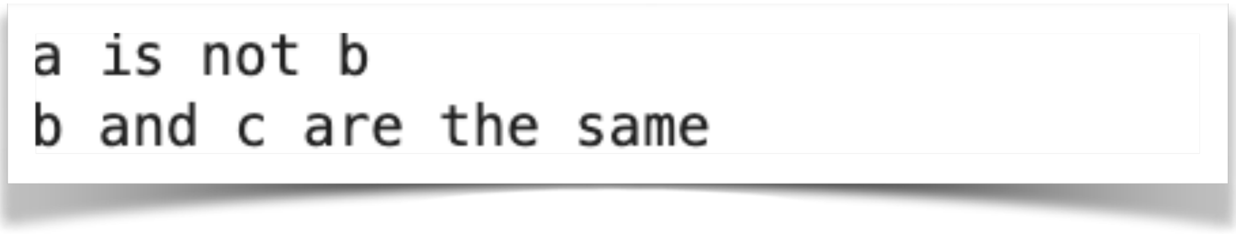
Some plusses

- Free
- Readable syntax
- Cross platform
- Huge community
- Used across science *<u>*and*</u>* outside of science

Some minuses



```python
a = 1
b = 2
c = 2
if a is not b:
    print('a is not b')
if b is c:
    print('b and c are the same')
```



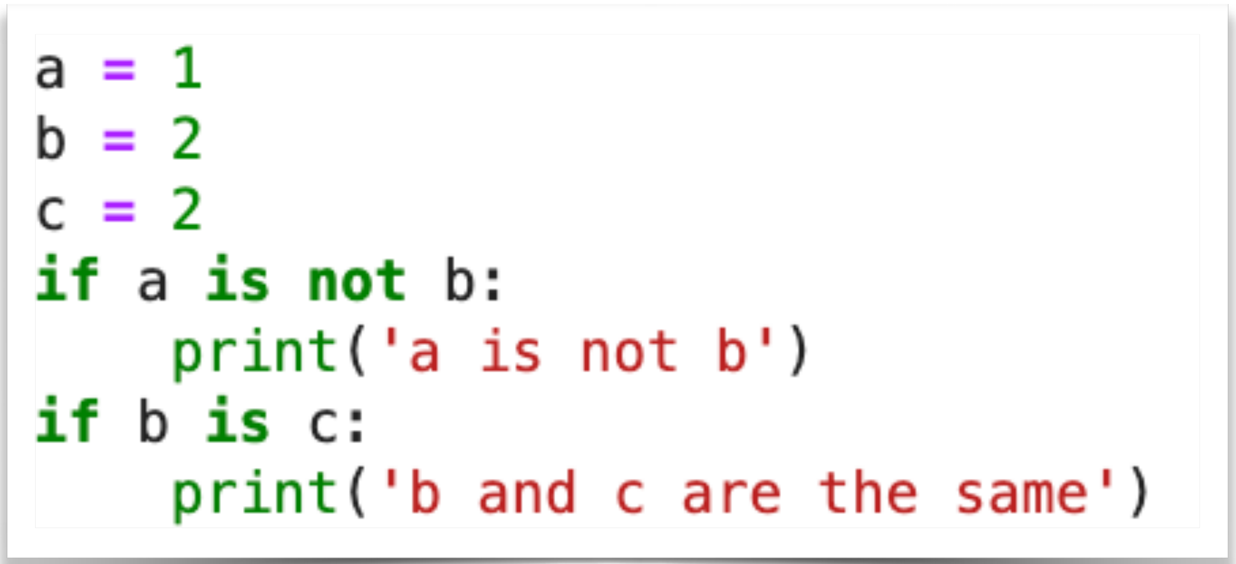
```
a is not b
b and c are the same
```

overview

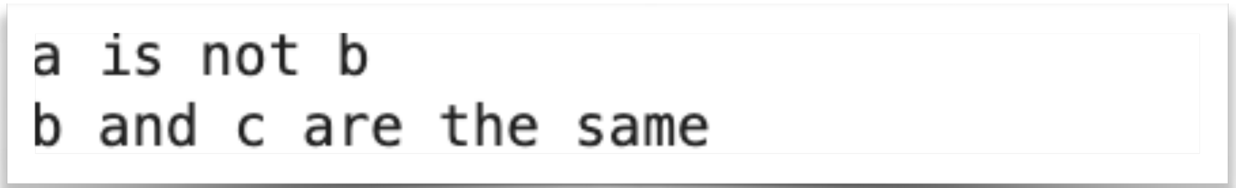
Some plusses

- **Free**
- **Readable syntax**
- **Cross platform**
- **Huge community**
- **Used across science ****and**** outside of science**

Some minuses



```
a = 1
b = 2
c = 2
if a is not b:
    print('a is not b')
if b is c:
    print('b and c are the same')
```



```
a is not b
b and c are the same
```

overview

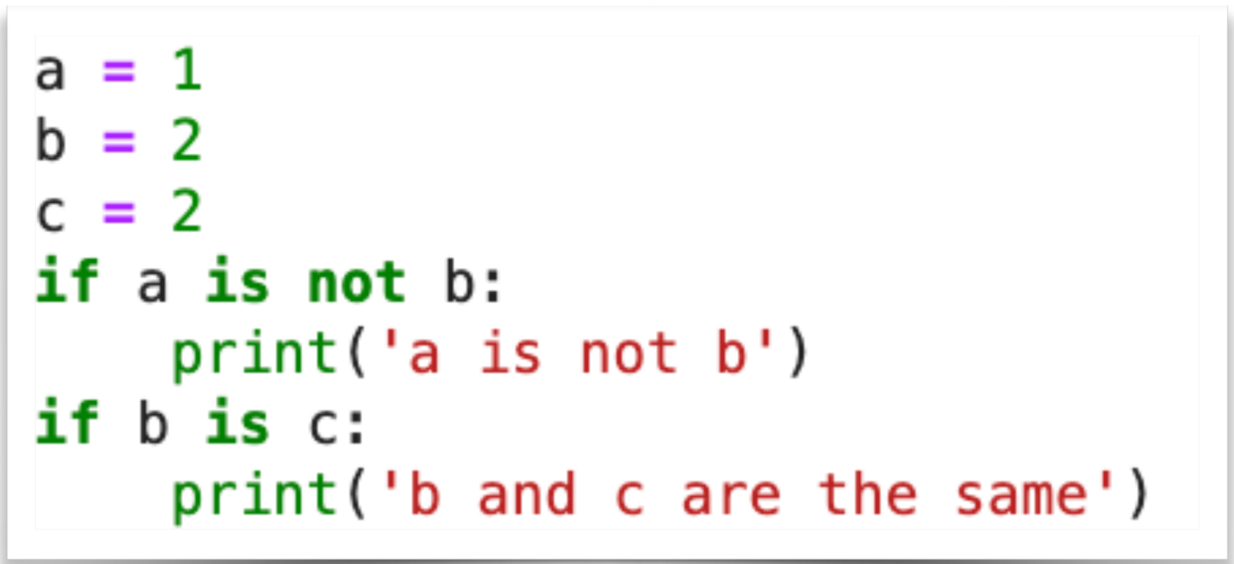
Some plusses

- **Free**
- **Readable syntax**
- **Cross platform**
- **Huge community**
- **Used across science ****and**** outside of science**

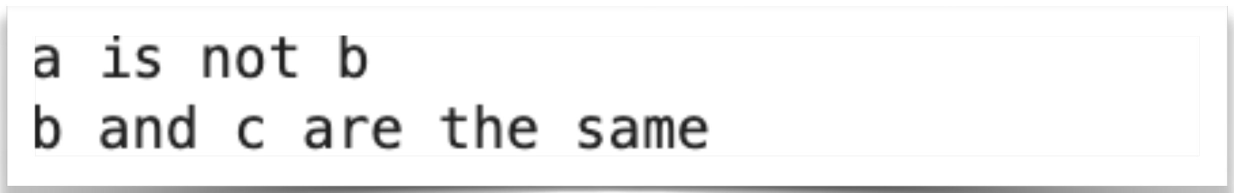
Some minuses

- **Slow[er] to execute (than C)**
- **White space matters (which some find to be a pain)**
- ***Sometimes problematic + insular culture; see "BDFL"**

***editorial opinion**



```
a = 1
b = 2
c = 2
if a is not b:
    print('a is not b')
if b is c:
    print('b and c are the same')
```



```
a is not b
b and c are the same
```

overview: Python language basics

overview: Python language basics

"high-level" :

you don't need to know how a computer *actually* works to use Python. Memory addressing, pointers, call stacks - blah blah computer science

garbage collected:

you, the programmer do not usually need to worry about cleaning up your "garbage" - memory pointers etc. More stability, fewer resources.

overview: Python language basics

“high-level” :

you don’t need to know how a computer *actually* works to use Python. Memory addressing, pointers, call stacks - blah blah computer science

garbage collected:

you, the programmer do not usually need to worry about cleaning up your “garbage” - memory pointers etc. More stability, fewer resources.

Interpreted

Translated into steps for the computer to execute one-by-one, as opposed to translating all of the steps at the beginning, and then (e.g, Igor is a compiled language).

overview: Python language basics

“high-level” :

you don’t need to know how a computer *actually* works to use Python. Memory addressing, pointers, call stacks - blah blah computer science

garbage collected:

you, the programmer do not usually need to worry about cleaning up your “garbage” - memory pointers etc. More stability, fewer resources.

Interpreted

Translated into steps for the computer to execute one-by-one, as opposed to translating all of the steps at the beginning, and then (e.g, Igor is a compiled language).

Several programming paradigms:

Procedural programming (like C), Object-oriented programming (like Java), functional programming (Wolfram)

overview: Python language basics

“high-level” :

you don’t need to know how a computer *actually* works to use Python. Memory addressing, pointers, call stacks - blah blah computer science

garbage collected:

you, the programmer do not usually need to worry about cleaning up your “garbage” - memory pointers etc. More stability, fewer resources.

Interpreted

Translated into steps for the computer to execute one-by-one, as opposed to translating all of the steps at the beginning, and then (e.g, Igor is a compiled language).

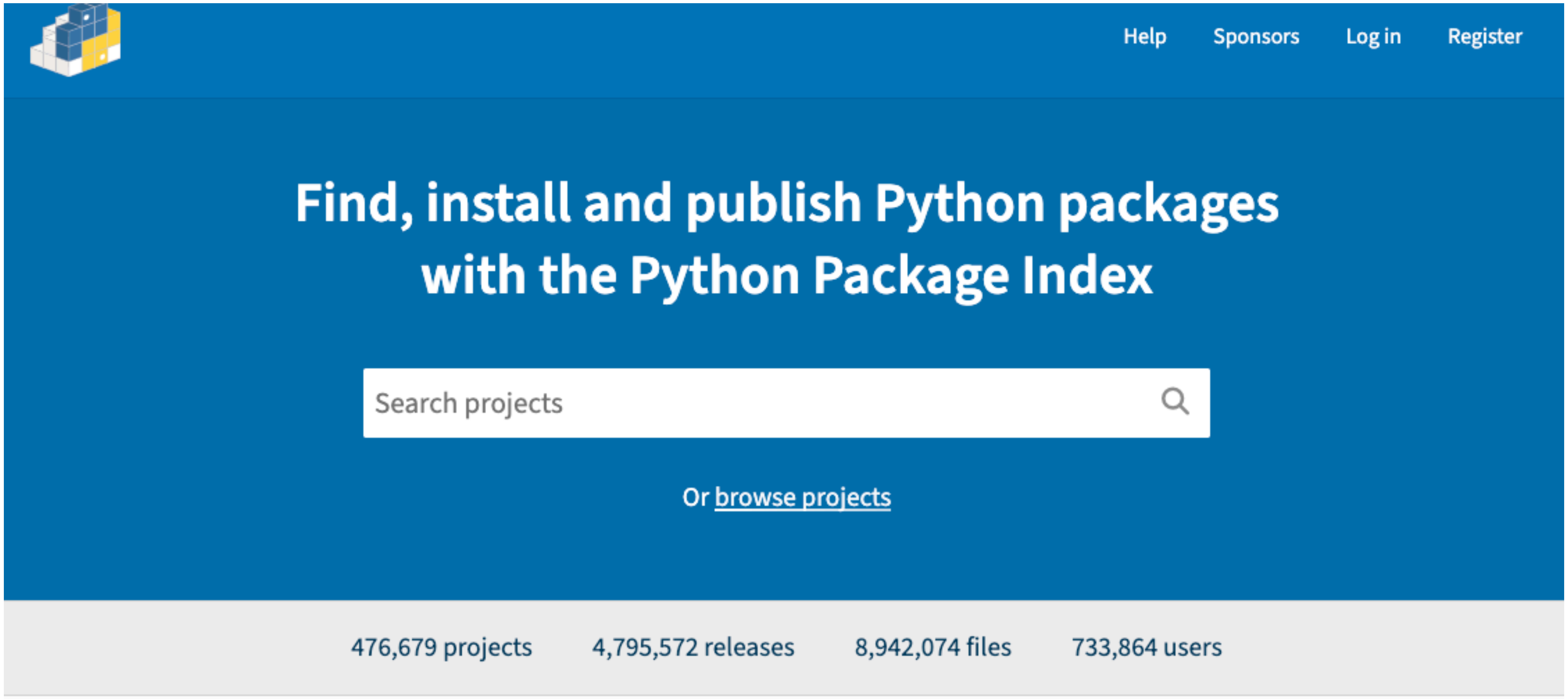
Several programming paradigms:

Procedural programming (like C), Object-oriented programming (like Java), functional programming (Wolfram)

Standard library

Often written in C (again, you, the programmer, don’t need to care about this), these are functions and tools that ship with Python. Widely useful, and already vetted. Backwards compatible.

overview: packages


Help Sponsors Log in Register

Find, install and publish Python packages with the Python Package Index

Search projects

Or browse projects

476,679 projects 4,795,572 releases 8,942,074 files 733,864 users

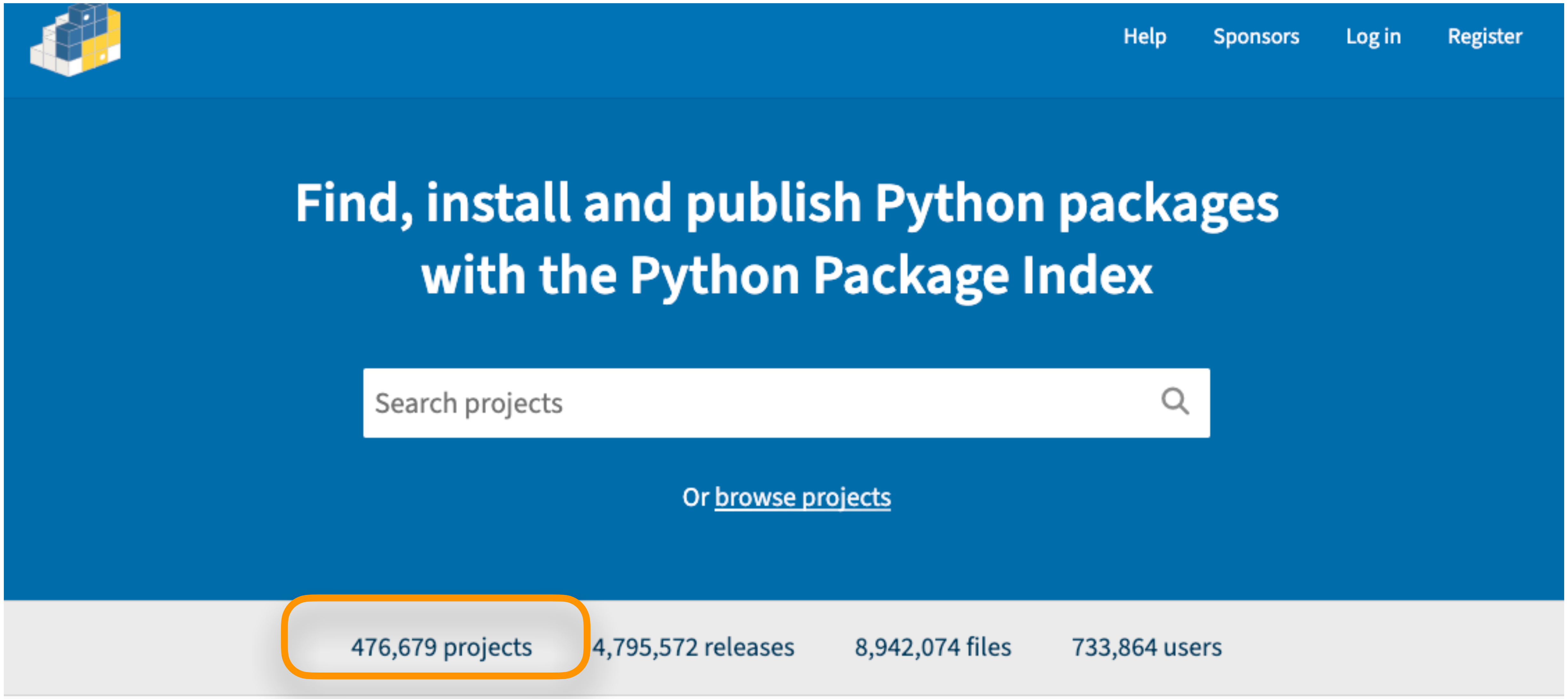
Packages

In addition to the standard library, the true power of python is the extensive world of packages available. These are sets of tools you can use with Python to do just about anything!

Some are general tools, the hammers or screwdrivers of using python for science: **numpy**, **matplotlib, pandas, seaborn**

Others are specialized: **scikit-learn, PIL, scanpy, Suite2P, DeepLabCut, PyTom**

overview: packages



Find, install and publish Python packages with the Python Package Index

476,679 projects 4,795,572 releases 8,942,074 files 733,864 users

Packages

In addition to the standard library, the true power of python is the extensive world of packages available. These are sets of tools you can use with Python to do just about anything!

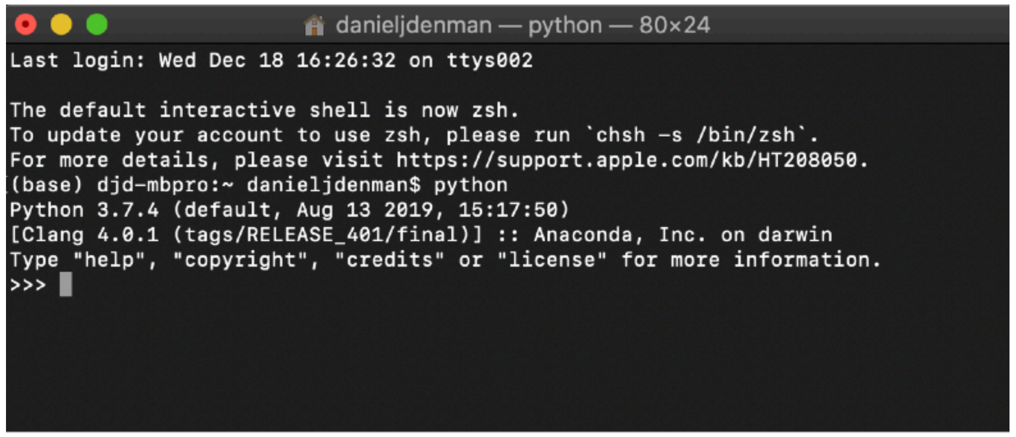
Some are general tools, the hammers or screwdrivers of using python for science: **numpy**, **matplotlib, pandas, seaborn**

Others are specialized: **scikit-learn, PIL, scanpy, Suite2P, DeepLabCut, PyTom**

overview: levels

overview: levels

System



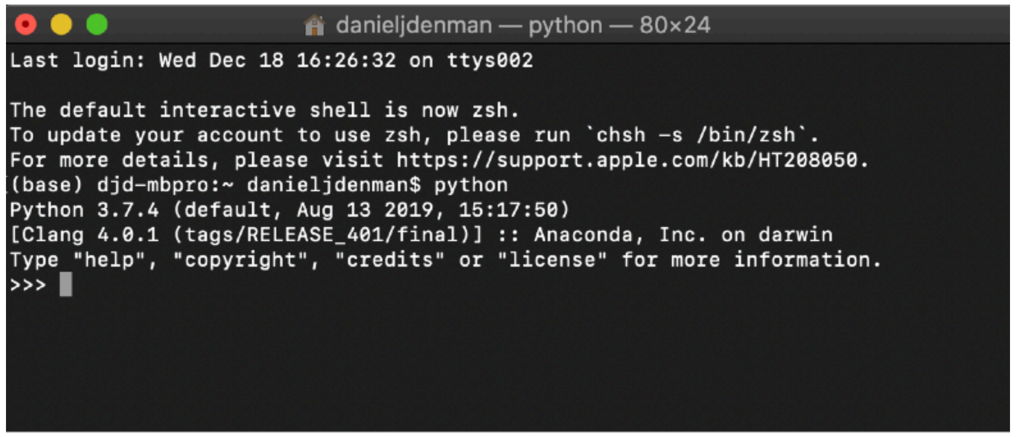
danieljdenman — python — 80×24

```
Last login: Wed Dec 18 16:26:32 on ttys002

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) djd-mbpro:~ danieljdenman$ python
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Native in Mac OS X, Linux; in Windows store (free)

overview: levels

System



```
Last login: Wed Dec 18 16:26:32 on ttys002

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) djd-mbpro:~ danieljdenman$ python
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

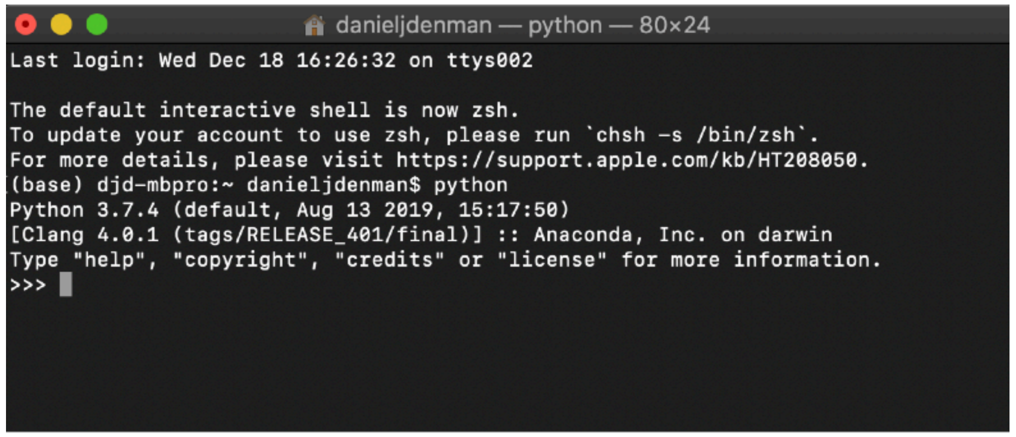
Native in Mac OS X, Linux; in Windows store (free)

Package Managers
Environments

ANACONDA®

overview: levels

System



```
danieljdenman — python — 80×24
Last login: Wed Dec 18 16:26:32 on ttys002

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) djd-mbpro:~ danieljdenman$ python
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Native in Mac OS X, Linux; in Windows store (free)

Package Managers Environments

ANACONDA

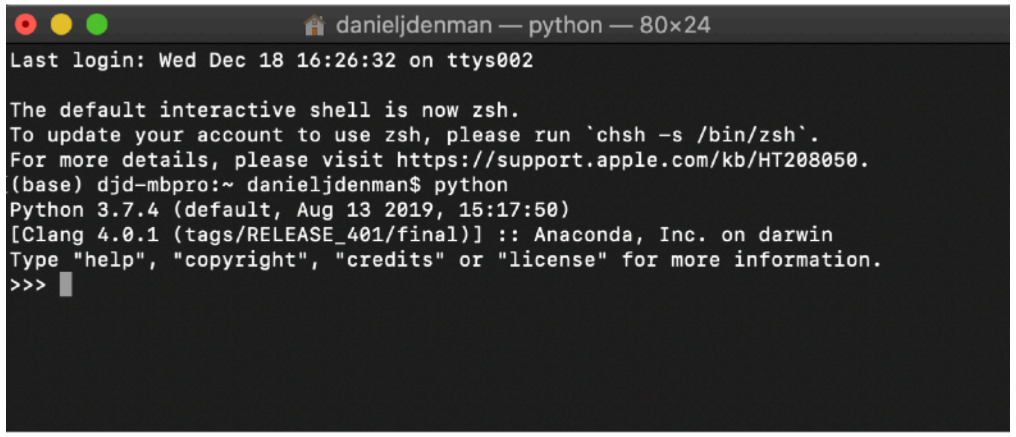
Containerized

docker

COCALC

overview: levels

System



```
Last login: Wed Dec 18 16:26:32 on ttys002

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) djd-mbpro:~ danieljdenman$ python
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Native in Mac OS X, Linux; in Windows store (free)

Package Managers
Environments

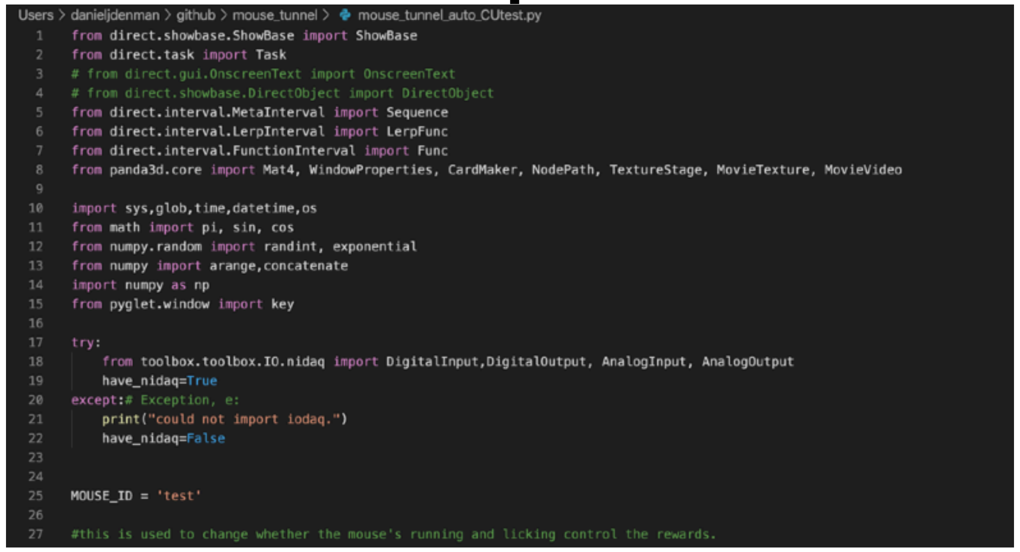
ANACONDA®

Containerized

docker

COCALC

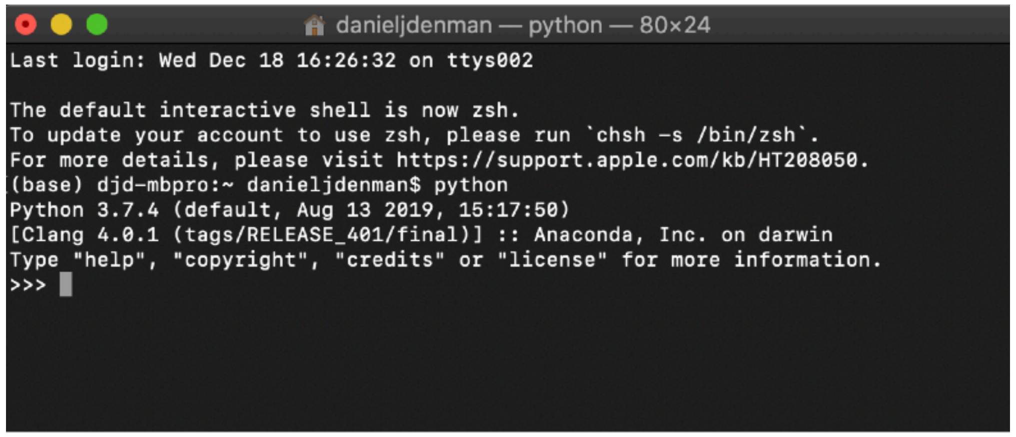
Scripts


Users > danieljdenman > github > mouse_tunnel > mouse_tunnel_auto_CUtest.py

```python
from direct.showbase.ShowBase import ShowBase
from direct.task import Task
# from direct.gui.OnscreenText import OnscreenText
# from direct.showbase.DirectObject import DirectObject
from direct.interval.MetaInterval import Sequence
from direct.interval.LerpInterval import LerpFunc
from direct.interval.FunctionInterval import Func
from panda3d.core import Mat4, WindowProperties, CardMaker, NodePath, TextureStage, MovieTexture, MovieVideo

import sys,glob,time,datetime,os
from math import pi, sin, cos
from numpy.random import randint, exponential
from numpy import arange,concatenate
import numpy as np
from pyglet.window import key

try:
    from toolbox.toolbox.IO.nidaq import DigitalInput,DigitalOutput, AnalogInput, AnalogOutput
    have_nidaq=True
except:# Exception, e:
    print("could not import iodaq.")
    have_nidaq=False


MOUSE_ID = 'test'

#this is used to change whether the mouse's running and licking control the rewards.
```

overview: levels

System


danieljdenman — python — 80×24

```
Last login: Wed Dec 18 16:26:32 on ttys002

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) djd-mbpro:~ danieljdenman$ python
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Native in Mac OS X, Linux; in Windows store (free)

Package Managers Environments

ANACONDA®

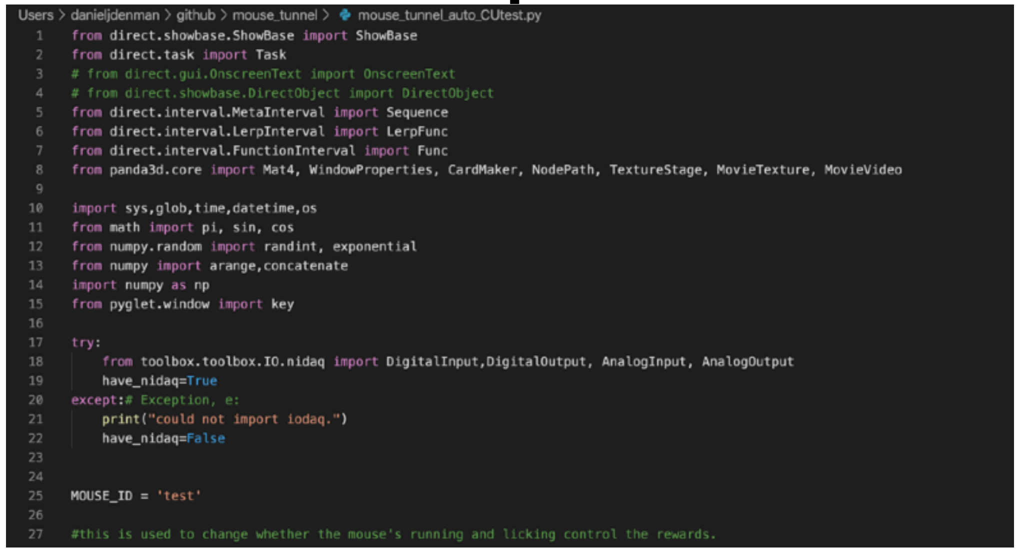
Containerized

docker

COCALC

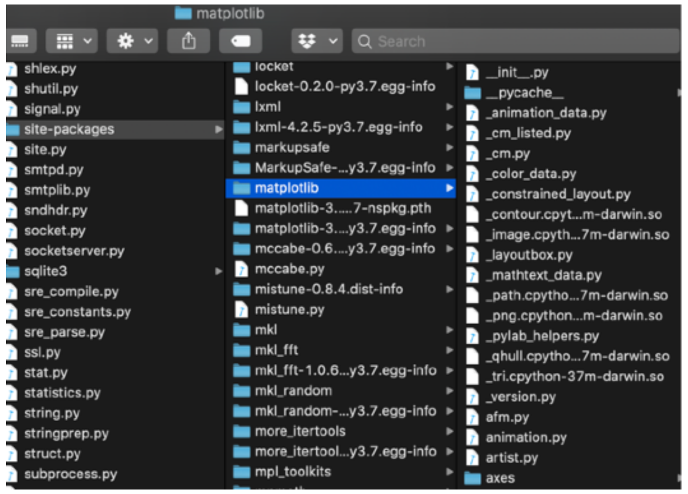
CO

Scripts


Users > danieljdenman > github > mouse_tunnel > mouse_tunnel_auto_CUtest.py

```python
from direct.showbase.ShowBase import ShowBase
from direct.task import Task
# from direct.gui.OnscreenText import OnscreenText
# from direct.showbase.DirectObject import DirectObject
from direct.interval.MetaInterval import Sequence
from direct.interval.LerpInterval import LerpFunc
from direct.interval.FunctionInterval import Func
from panda3d.core import Mat4, WindowProperties, CardMaker, NodePath, TextureStage, MovieTexture, MovieVideo

import sys,glob,time,datetime,os
from math import pi, sin, cos
from numpy.random import randint, exponential
from numpy import arange,concatenate
import numpy as np
from pyglet.window import key

try:
    from toolbox.toolbox.IO.nidaq import DigitalInput,DigitalOutput, AnalogInput, AnalogOutput
    have_nidaq=True
except:# Exception, e:
    print("could not import iodaq.")
    have_nidaq=False


MOUSE_ID = 'test'

#this is used to change whether the mouse's running and licking control the rewards.
```

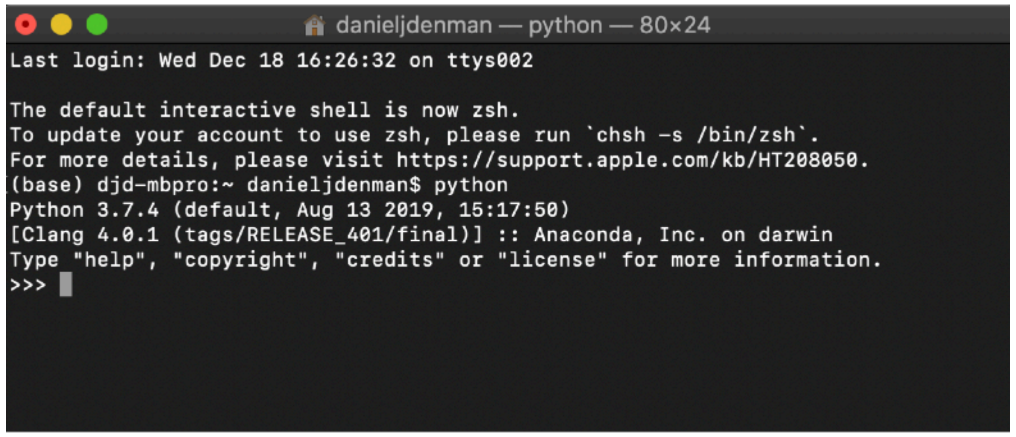
Packages


matplotlib

```
shlex.py              locket                          __init__.py
shutil.py             locket-0.2.0-py3.7.egg-info     __pycache__
signal.py             lxml                            _animation_data.py
site-packages         lxml-4.2.5-py3.7.egg-info       _cm_listed.py
site.py               markupsafe                      _cm.py
smtpd.py              MarkupSafe-...y3.7.egg-info     _color_data.py
smtplib.py            matplotlib                      _constrained_layout.py
sndhdr.py             matplotlib-3.....7-nspkg.pth    _contour.cpyt...m-darwin.so
socket.py             matplotlib-3....y3.7.egg-info   _image.cpyth...7m-darwin.so
socketserver.py       mccabe-0.6....y3.7.egg-info     _layoutbox.py
sqlite3               mccabe.py                       _mathtext_data.py
sre_compile.py        mistune-0.8.4.dist-info         _path.cpytho...7m-darwin.so
sre_constants.py      mistune.py                      _png.cpython...m-darwin.so
sre_parse.py          mkl                             _pylab_helpers.py
ssl.py                mkl_fft                         _qhull.cpytho...7m-darwin.so
stat.py               mkl_fft-1.0.6...y3.7.egg-info   _tri.cpython-37m-darwin.so
statistics.py         mkl_random                      _version.py
string.py             mkl_random-...y3.7.egg-info     afm.py
stringprep.py         more_itertools                  animation.py
struct.py             more_itertool...y3.7.egg-info   artist.py
subprocess.py         mpl_toolkits                    axes
```

overview: levels

System



```
Last login: Wed Dec 18 16:26:32 on ttys002

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) djd-mbpro:~ danieljdenman$ python
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
[Clang 4.0.1 (tags/RELEASE_401/final)] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Native in Mac OS X, Linux; in Windows store (free)

Package Managers
Environments

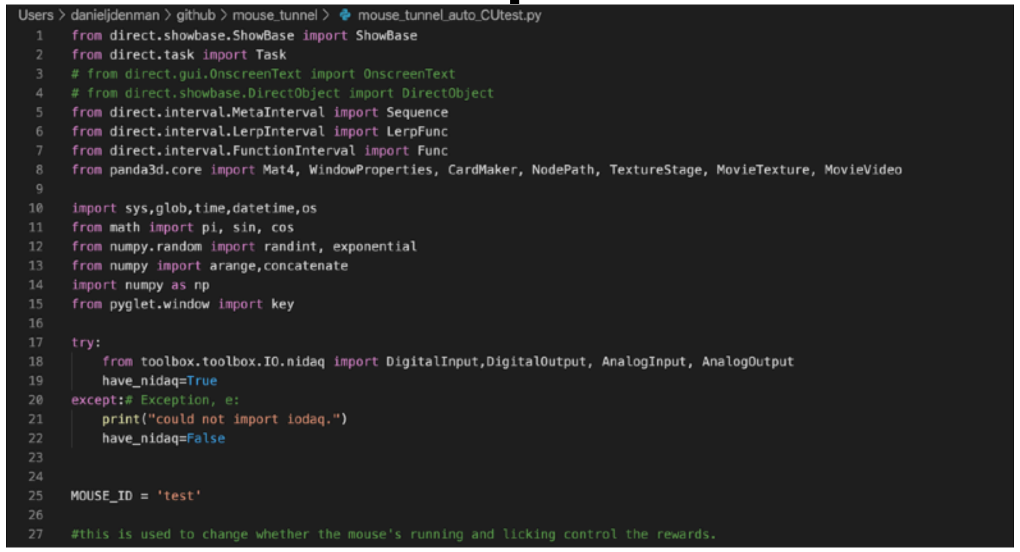
ANACONDA®

Containerized

docker

COCALC

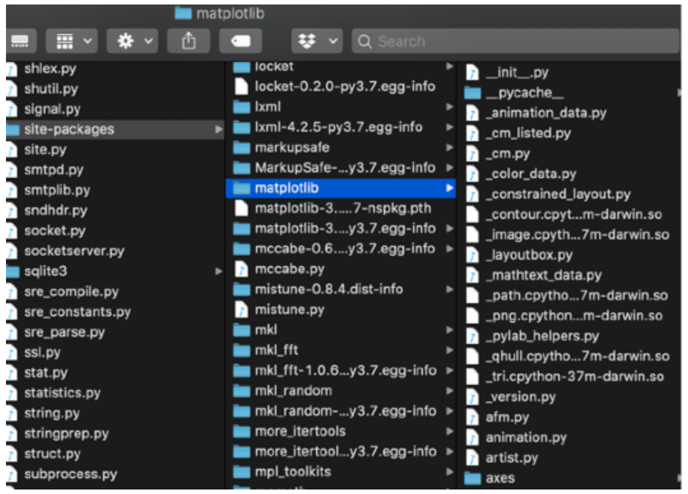
Scripts



Users > danieljdenman > github > mouse_tunnel > mouse_tunnel_auto_CUtest.py

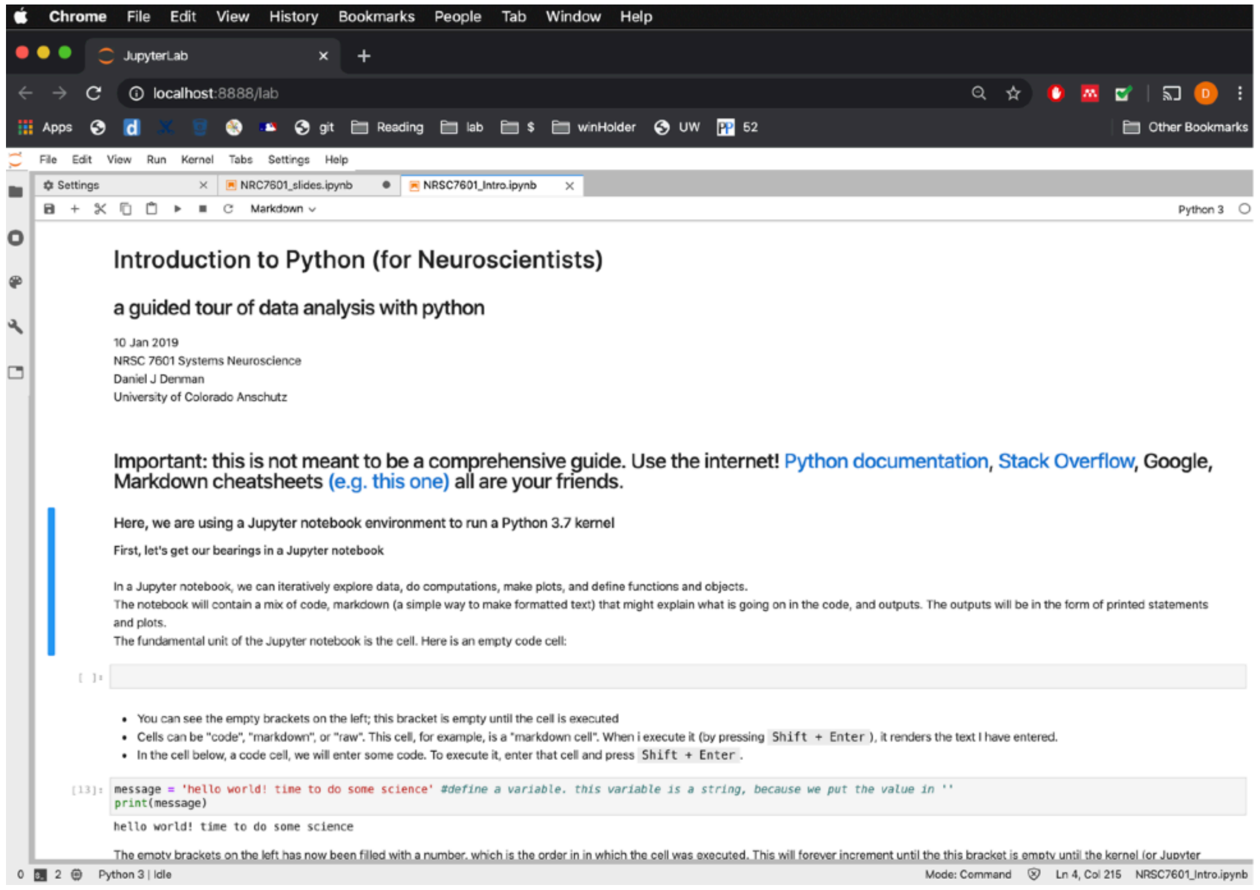
```python
from direct.showbase.ShowBase import ShowBase
from direct.task import Task
# from direct.gui.OnscreenText import OnscreenText
# from direct.showbase.DirectObject import DirectObject
from direct.interval.MetaInterval import Sequence
from direct.interval.LerpInterval import LerpFunc
from direct.interval.FunctionInterval import Func
from panda3d.core import Mat4, WindowProperties, CardMaker, NodePath, TextureStage, MovieTexture, MovieVideo

import sys,glob,time,datetime,os
from math import pi, sin, cos
from numpy.random import randint, exponential
from numpy import arange,concatenate
import numpy as np
from pyglet.window import key

try:
    from toolbox.toolbox.IO.nidaq import DigitalInput,DigitalOutput, AnalogInput, AnalogOutput
    have_nidaq=True
except:# Exception, e:
    print("could not import iodaq.")
    have_nidaq=False


MOUSE_ID = 'test'

#this is used to change whether the mouse's running and licking control the rewards.
```

Packages



```
shlex.py             locket                               __init__.py
shutil.py            locket-0.2.0-py3.7.egg-info          __pycache__
signal.py            lxml                                 _animation_data.py
site-packages        lxml-4.2.5-py3.7.egg-info            _cm_listed.py
site.py              markupsafe                           _cm.py
smtpd.py             MarkupSafe-...y3.7.egg-info          _color_data.py
smtplib.py           matplotlib                           _constrained_layout.py
sndhdr.py            matplotlib-3.....7-nspkg.pth         _contour.cpyt...m-darwin.so
socket.py            matplotlib-3....y3.7.egg-info        _image.cpyth...7m-darwin.so
socketserver.py      mccabe-0.6....y3.7.egg-info          _layoutbox.py
sqlite3              mccabe.py                            _mathtext_data.py
sre_compile.py       mistune-0.8.4.dist-info              _path.cpytho...7m-darwin.so
sre_constants.py     mistune.py                           _png.cpython...m-darwin.so
sre_parse.py         mkl                                  _pylab_helpers.py
ssl.py               mkl_fft                              _qhull.cpytho...7m-darwin.so
stat.py              mkl_fft-1.0.6...y3.7.egg-info        _tri.cpython-37m-darwin.so
statistics.py        mkl_random                           _version.py
string.py            mkl_random-...y3.7.egg-info          afm.py
stringprep.py        more_itertools                       animation.py
struct.py            more_itertool...y3.7.egg-info        artist.py
subprocess.py        mpl_toolkits                         axes
```

Notebooks (IPython, Jupyter, Jupyter Lab)



Chrome File Edit View History Bookmarks People Tab Window Help
JupyterLab
localhost:8888/lab
Apps git Reading lab $ winHolder UW 52 Other Bookmarks
File Edit View Run Kernel Tabs Settings Help
Settings
NRC7601_slides.ipynb
NRSC7601_Intro.ipynb
Markdown
Python 3

Introduction to Python (for Neuroscientists)

a guided tour of data analysis with python

10 Jan 2019
NRSC 7601 Systems Neuroscience
Daniel J Denman
University of Colorado Anschutz

Important: this is not meant to be a comprehensive guide. Use the internet! Python documentation, Stack Overflow, Google, Markdown cheatsheets (e.g. this one) all are your friends.

Here, we are using a Jupyter notebook environment to run a Python 3.7 kernel

First, let's get our bearings in a Jupyter notebook

In a Jupyter notebook, we can iteratively explore data, do computations, make plots, and define functions and objects.
The notebook will contain a mix of code, markdown (a simple way to make formatted text) that might explain what is going on in the code, and outputs. The outputs will be in the form of printed statements and plots.
The fundamental unit of the Jupyter notebook is the cell. Here is an empty code cell:

```
[ ]:
```

- You can see the empty brackets on the left; this bracket is empty until the cell is executed
- Cells can be "code", "markdown", or "raw". This cell, for example, is a "markdown cell". When i execute it (by pressing `Shift + Enter`), it renders the text I have entered.
- In the cell below, a code cell, we will enter some code. To execute it, enter that cell and press `Shift + Enter`.

```
[13]: message = 'hello world! time to do some science' #define a variable. this variable is a string, because we put the value in ''
      print(message)

hello world! time to do some science
```

The empty brackets on the left has now been filled with a number, which is the order in in which the cell was executed. This will forever increment until the this bracket is empty until the kernel (or Jupyter

why is it good for doing neuroscience?

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)

packages!

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)
- automate boring stuff / use other people's hard work

packages!

Stack Overflow <— not cheating!

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)
- automate boring stuff / use other people’s hard work
- make your science, and science., better

<u>packages!</u>

Stack Overflow <— not cheating!

also it is free —> democratizing science

in this realm, cloud resources (data, compute) also open science to a wider group that aren’t collecting their own data and running their own super computers

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)
- automate boring stuff / use other people's hard work
- make your science, and science., better

packages!

Stack Overflow <— not cheating!

also it is free —> democratizing science

in this realm, cloud resources (data, compute) also open science to a wider group that aren't collecting their own data and running their own super computers

Hardware control

RasberryPi
Arduino
PyDAQMX
PsychoPy
…many APIs…

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)
- automate boring stuff / use other people's hard work
- make your science, and science., better

packages!

Stack Overflow <— not cheating!

also it is free —> democratizing science

in this realm, cloud resources (data, compute) also open science to a wider group that aren't collecting their own data and running their own super computers

Hardware control

RasberryPi
Arduino
PyDAQMX
PsychoPy
…many APIs…

Data science

scikit-learn
Pandas
TensorFlow

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)
- automate boring stuff / use other people's hard work
- make your science, and science., better

packages!

Stack Overflow <— not cheating!

also it is free —> democratizing science

in this realm, cloud resources (data, compute) also open science to a wider group that aren't collecting their own data and running their own super computers

Hardware control

RasberryPi
Arduino
PyDAQMX
PsychoPy
…many APIs…

Data science

scikit-learn
Pandas
TensorFlow

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)
- automate boring stuff / use other people's hard work
- make your science, and science., better

packages!

Stack Overflow <— not cheating!

also it is free —> democratizing science

in this realm, cloud resources (data, compute) also open science to a wider group that aren't collecting their own data and running their own super computers

Hardware control

RasberryPi
Arduino
PyDAQMX
PsychoPy
…many APIs…

Data science

scikit-learn
Pandas
TensorFlow

Specialized Tools

Image anlysis: PIL / OpenCV
Ca2+ analysis: Suite2P, AQUA
Movement Tracking: DeepLabCut
Expression Analysis: scanpy
…

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)
- automate boring stuff / use other people's hard work
- make your science, and science., better

packages!

Stack Overflow <— not cheating!

also it is free —> democratizing science

in this realm, cloud resources (data, compute) also open science to a wider group that aren't collecting their own data and running their own super computers

Hardware control

RasberryPi
Arduino
PyDAQMX
PsychoPy
…many APIs…

Data science

scikit-learn
Pandas
TensorFlow

Specialized Tools

Image anlysis: PIL / OpenCV
Ca2+ analysis: Suite2P, AQUA
Movement Tracking: DeepLabCut
Expression Analysis: scanpy
…

Sharing

Docker
Google Colab
Jupyter
[it is free!]

why is it good for doing neuroscience?

- Do analyses that would be a whole PhD to implelement yourself (i.e., ML)
- automate boring stuff / use other people's hard work
- make your science, and science., better

packages!

Stack Overflow <— not cheating!

also it is free —> democratizing science

in this realm, cloud resources (data, compute) also open science to a wider group that aren't collecting their own data and running their own super computers

Hardware control

RasberryPi
Arduino
PyDAQMX
PsychoPy
…many APIs…

Data science

scikit-learn
Pandas
TensorFlow

Specialized Tools

Image anlysis: PIL / OpenCV
Ca2+ analysis: Suite2P, AQUA
Movement Tracking: DeepLabCut
Expression Analysis: scanpy
…

Sharing

Docker
Google Colab
Jupyter
[it is free!]

Guided example:

- ☐ Variables: definitions, types
- ☐ Some important packages
- ☐ Making plots
- ☐ Functions

git

…and GitHub, are **version control**

- This is important. And not intuitive. It will likely make you frustrated and/or confused at some point.
- Version control is not optional; if you don't use git for version control, you are going to use something else (e.g,: analysis_script_v1.py, analysis_script_v2.py, analysis_script_v2_20210622.py, analysis_script_v3_07142021.py, analysis_script_final.py,analysis_script_final2.py,…, analysis_script_final2_for.py)
- Making git a part of your workflow can simplify and provide redundancy and flexibility; more advanced features also makes sharing simpler. Evaluation.
- git has to be installed, which we will use Anaconda to do so
- GitHub Desktop https://desktop.github.com/ is by far the easiest way; git bash (command line) is another option
- We're going to go over some git interactively to get course materials today.