NeuroPy week 2: Practical applications and troubleshooting

CENTRE DE RECHERCHE CERVO BRAIN RESEARCH CENTRE

Sandrine Poulin & Antoine Daigle

July 16th 2024

What we have learned last week

- Setting up your Python environment
  - Distributions, IDE and Jupyter Notebook
- Syntax structure
  - int
  - float
  - list
  - dict
  - str
- Control structure
  - if/elif/else
  - for loop
- Function
  - Syntax
  - Documentation

What is the difference between the *int* and the *float*?

What is the difference between the *if*, *elif* and the *else*?

What is the behaviour of the *for* loop?

Plan of this workshop

Part 1 (~45 minutes): Theory

1. Import and manipulate data

   Pandas and NumPy modules
2. Visualise data

   Matplotlib module
3. Troubleshooting

   Resources available

Part 2 (~45 minutes): Examples

Sides:

Google colab:

0 Modules

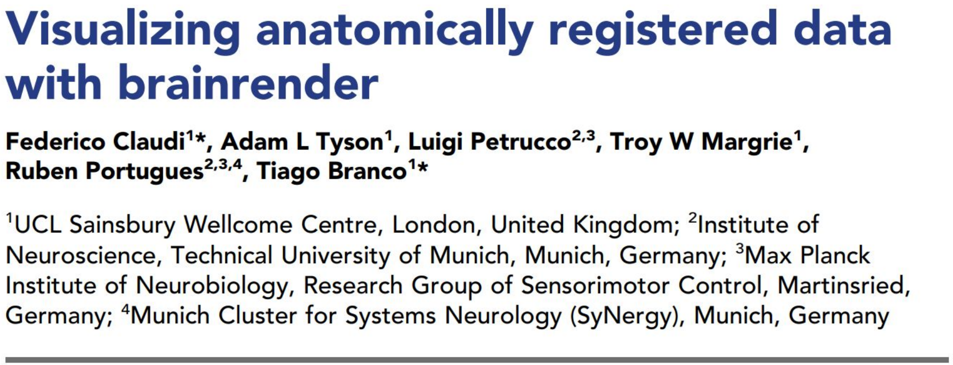
Modules contain pre-coded functions! There are modules for almost everything:

- Math (numpy, math, scipy, sympy, …)
- Visualisation (matplotlib, seaborn, PyQt5, turtle, …)
- Data science (pandas, pyserial…)

You need to install the packages to use them.

- With anaconda: command “conda install XXXX” in the terminal
- With pip: PyPi or with the command “pip install XXXX”

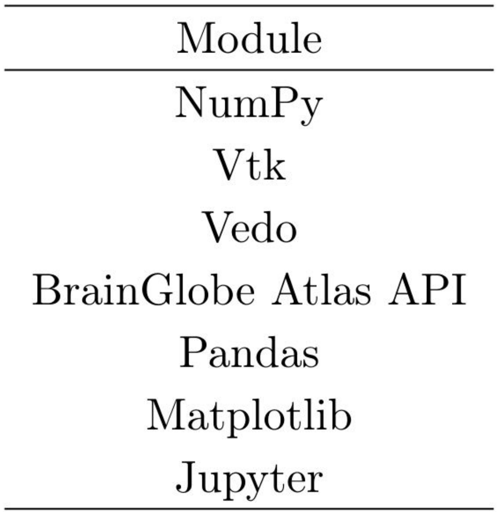
Brainrender module



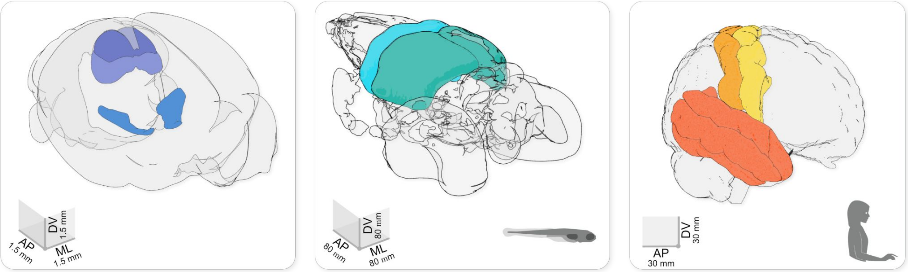
Visualizing anatomically registered data with brainrender

Federico Claudi[1]*, Adam L Tyson[1], Luigi Petrucco[2,3], Troy W Margrie[1], Ruben Portugues[2,3,4], Tiago Branco[1]*

[1]UCL Sainsbury Wellcome Centre, London, United Kingdom; [2]Institute of Neuroscience, Technical University of Munich, Munich, Germany; [3]Max Planck Institute of Neurobiology, Research Group of Sensorimotor Control, Martinsried, Germany; [4]Munich Cluster for Systems Neurology (SyNergy), Munich, Germany



Module
NumPy
Vtk
Vedo
BrainGlobe Atlas API
Pandas
Matplotlib
Jupyter



Part 1

1. **Import and manipulate data**

   Pandas and NumPy modules
2. Visualise data

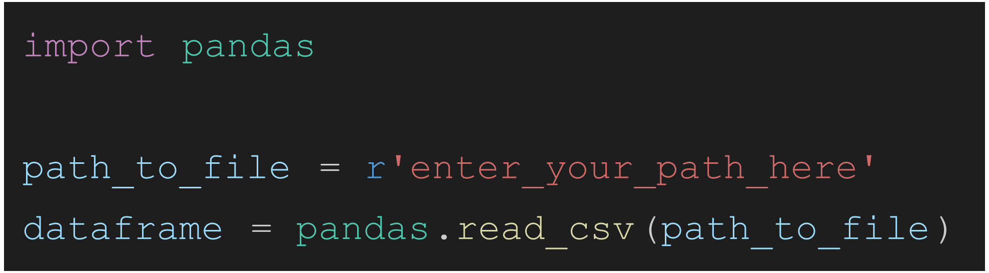
   Matplotlib module
3. Troubleshooting

   Resources available

1 The **Pandas** module

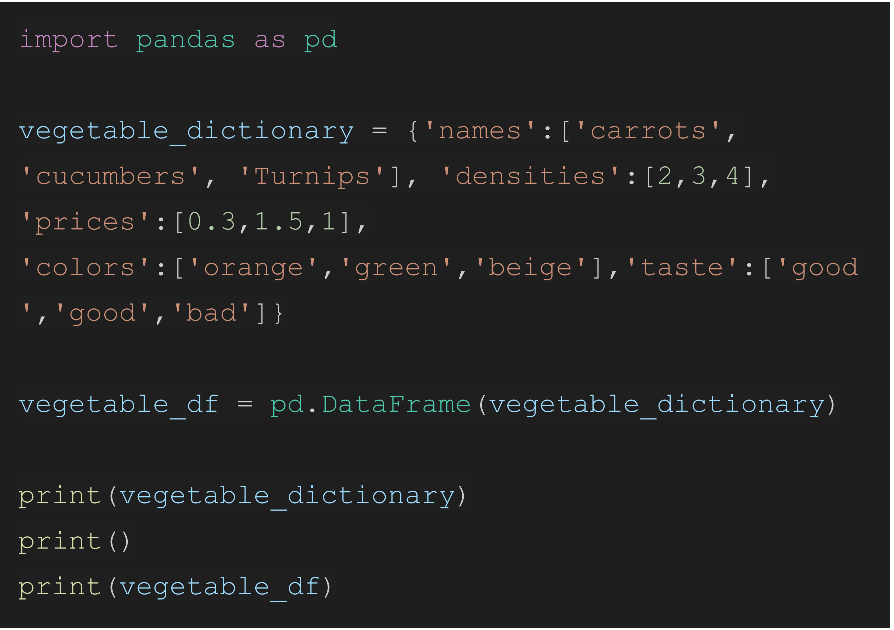
Pandas allows you to work with a dataframe just like in excel.

- Load a csv or xlsx file and work with the column name or localisation.

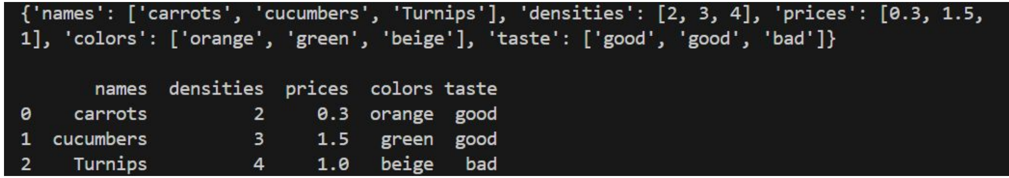


```
import pandas

path_to_file = r'enter_your_path_here'
dataframe = pandas.read_csv(path_to_file)
```

1 The **Pandas** module: creating a dataframe

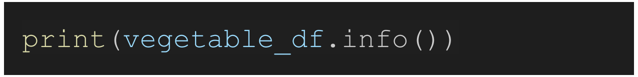


```
import pandas as pd

vegetable_dictionary = {'names':['carrots',
'cucumbers', 'Turnips'], 'densities':[2,3,4],
'prices':[0.3,1.5,1],
'colors':['orange','green','beige'],'taste':['good
','good','bad']}

vegetable_df = pd.DataFrame(vegetable_dictionary)

print(vegetable_dictionary)
print()
print(vegetable_df)
```

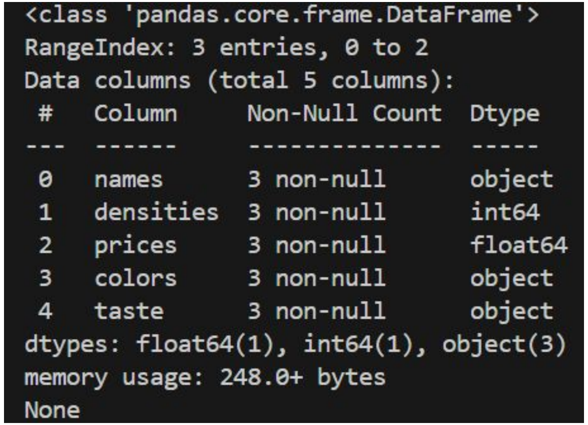


```
{'names': ['carrots', 'cucumbers', 'Turnips'], 'densities': [2, 3, 4], 'prices': [0.3, 1.5,
1], 'colors': ['orange', 'green', 'beige'], 'taste': ['good', 'good', 'bad']}

       names  densities  prices  colors taste
0    carrots          2     0.3  orange  good
1  cucumbers          3     1.5   green  good
2    Turnips          4     1.0   beige   bad
```

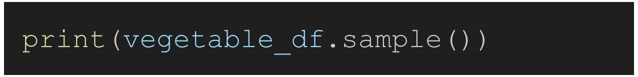
1 The **Pandas** module: useful functions



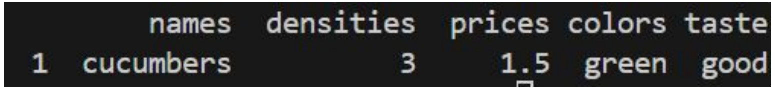
```
print(vegetable_df.info())
```



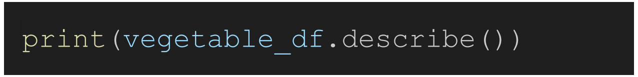
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   names      3 non-null      object
 1   densities  3 non-null      int64
 2   prices     3 non-null      float64
 3   colors     3 non-null      object
 4   taste      3 non-null      object
dtypes: float64(1), int64(1), object(3)
memory usage: 248.0+ bytes
None
```



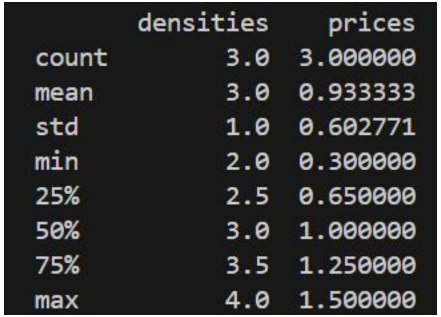
```
print(vegetable_df.sample())
```



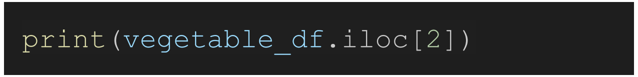
```
       names  densities  prices colors taste
1  cucumbers          3     1.5  green  good
```



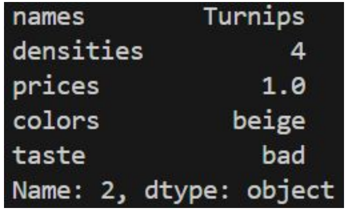
```
print(vegetable_df.describe())
```



```
       densities    prices
count        3.0  3.000000
mean         3.0  0.933333
std          1.0  0.602771
min          2.0  0.300000
25%          2.5  0.650000
50%          3.0  1.000000
75%          3.5  1.250000
max          4.0  1.500000
```

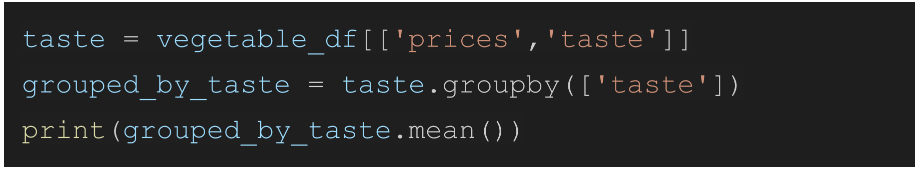


```
print(vegetable_df.iloc[2])
```

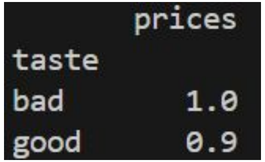


```
names        Turnips
densities          4
prices           1.0
colors         beige
taste            bad
Name: 2, dtype: object
```

1 The **Pandas** module: grouping by categories



```
taste = vegetable_df[['prices','taste']]
grouped_by_taste = taste.groupby(['taste'])
print(grouped_by_taste.mean())
```



```
       prices
taste
bad       1.0
good      0.9
```

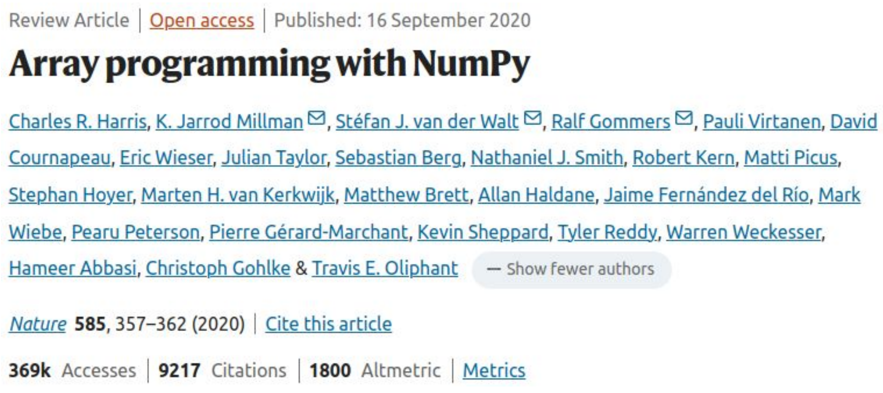
1 The **NumPy** module

The fundamental package for scientific computing. Allows you to work with an array (matrix). It’s like a boosted list.

- You can apply mathematical operations directly on them.

NumPy play a central role in all branches of science.

- Array programming with NumPy



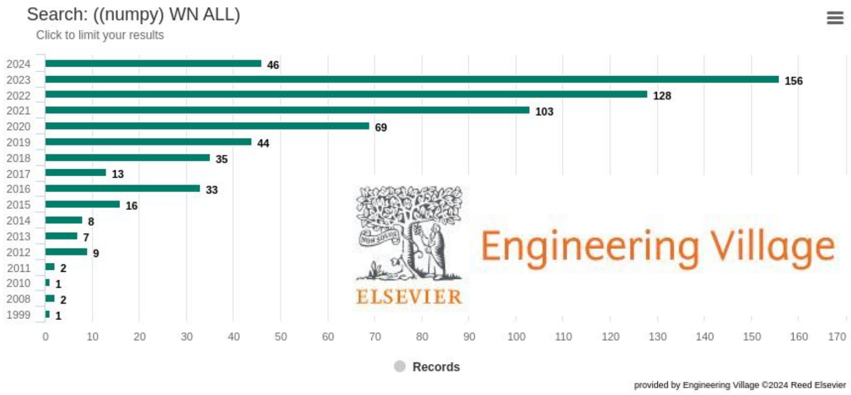
Review Article | Open access | Published: 16 September 2020

Array programming with NumPy

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke & Travis E. Oliphant — Show fewer authors

Nature **585**, 357–362 (2020) | Cite this article

369k Accesses | **9217** Citations | **1800** Altmetric | Metrics



1 NumPy 2.0.0

NumPy got a major update!

- First major update since 2006

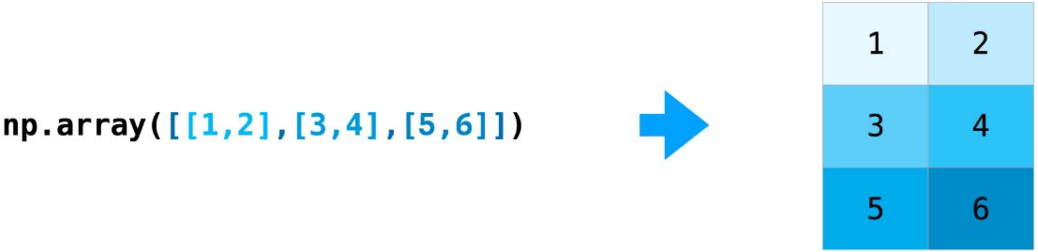
Is it that great?

- "It includes breaking changes that could not happen in a regular minor release [...]".

Stick to NumPy 1.25 or 1.26 for some time. Be careful when you install NumPy.

1 Creating matrices

You can pass Python lists of lists to create a 2-D array (or “matrix”) to represent them in NumPy.



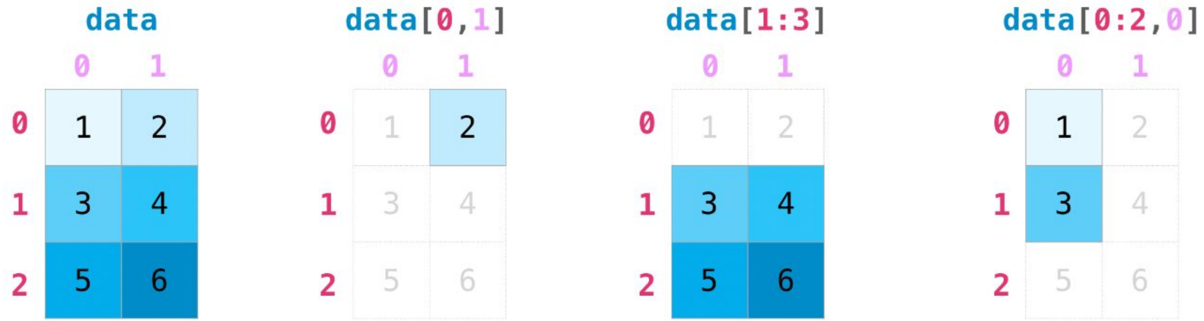
`np.array([[1,2],[3,4],[5,6]])`

1	2
3	4
5	6

```
import numpy as np

data = np.array([[1, 2], [3, 4], [5,
6]])
print(data)
```

1 Indexing and slicing

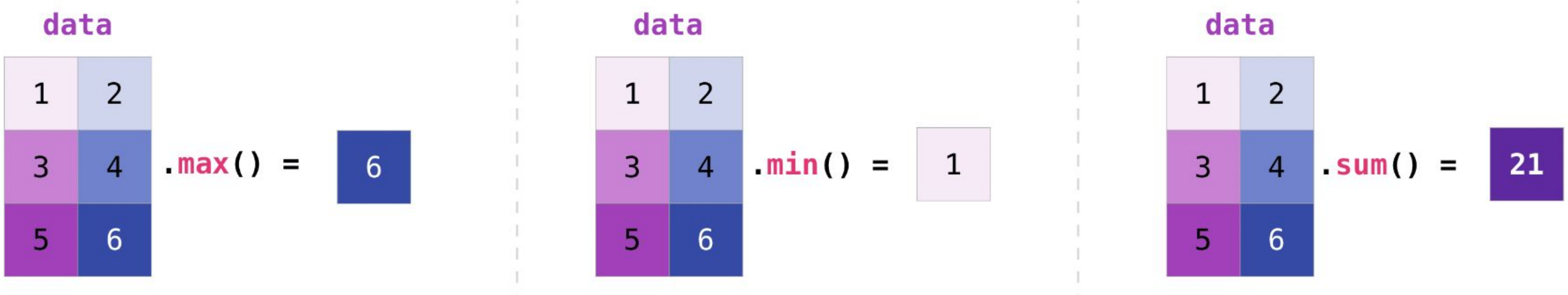
You can index and slice NumPy arrays in the same ways you can slice Python lists.

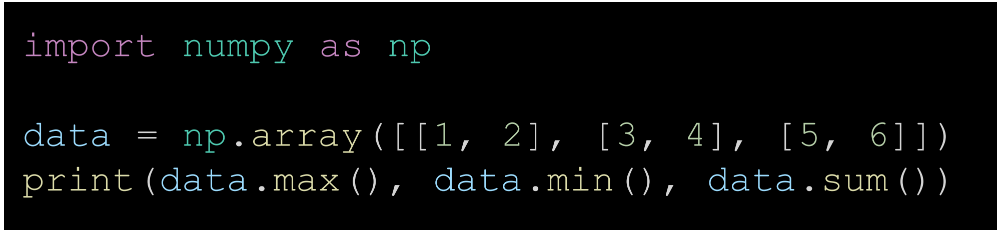


```
import numpy as np

data = np.array([[1, 2], [3, 4], [5, 6]])
print(data, data[0, 1], data[1:3], data[0:2,
0])
```

1 Useful aggregation function

NumPy also performs aggregation functions.





```
import numpy as np

data = np.array([[1, 2], [3, 4], [5, 6]])
print(data.max(), data.min(), data.sum())
```

Part 2

1. Import and manipulate data

    Pandas and NumPy modules

2. **Visualise data**

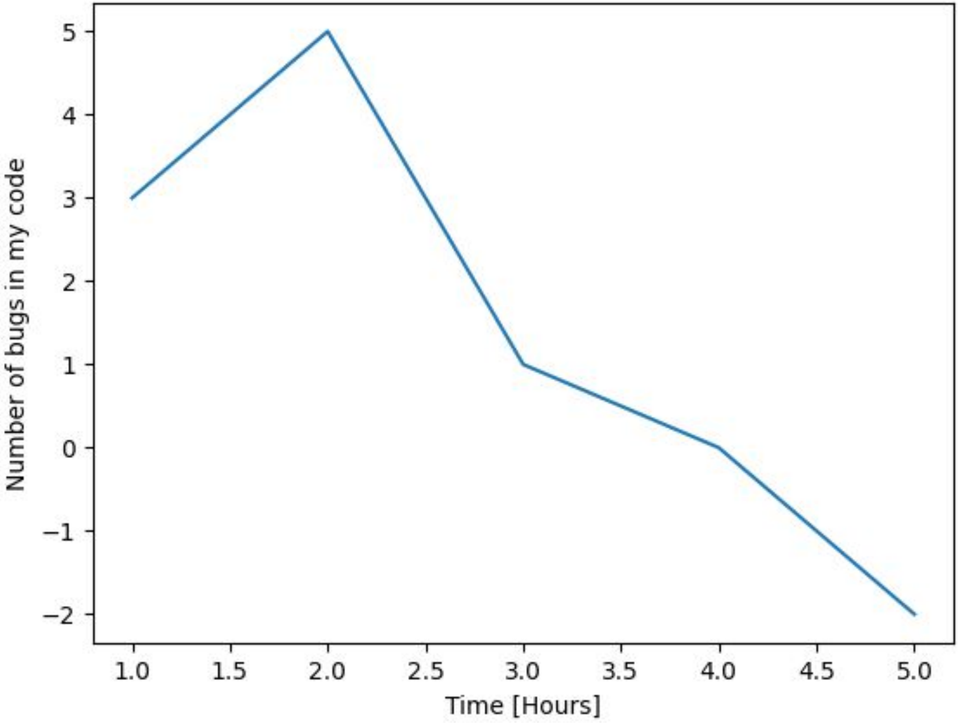
    Matplotlib module

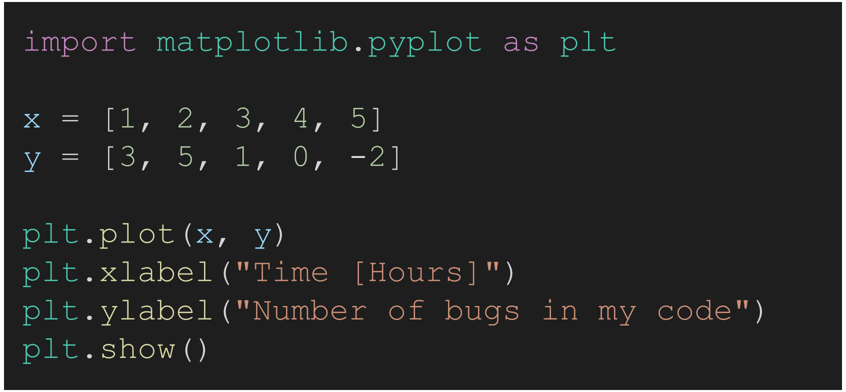
3. Troubleshooting

    Resources available

2 The **Matplotlib** module

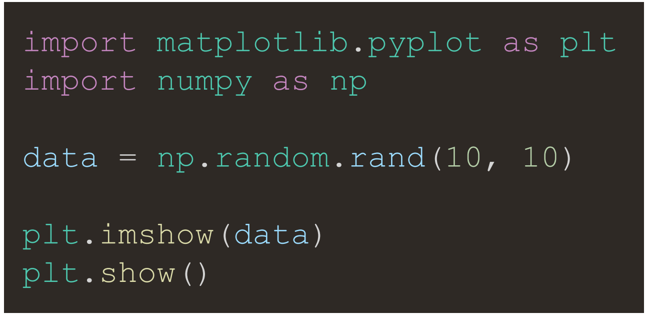
This module is used to plot and visualise data.



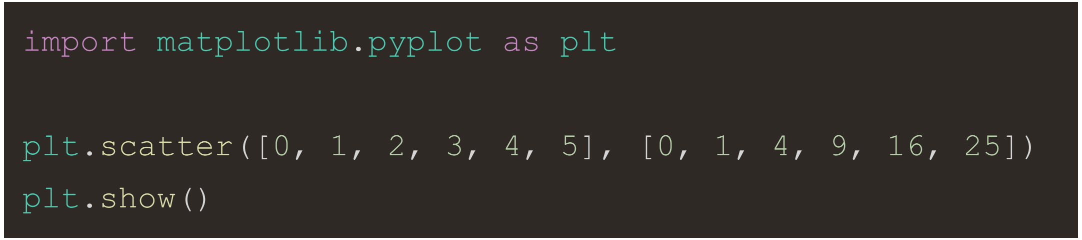


```
import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]
y = [3, 5, 1, 0, -2]

plt.plot(x, y)
plt.xlabel("Time [Hours]")
plt.ylabel("Number of bugs in my code")
plt.show()
```

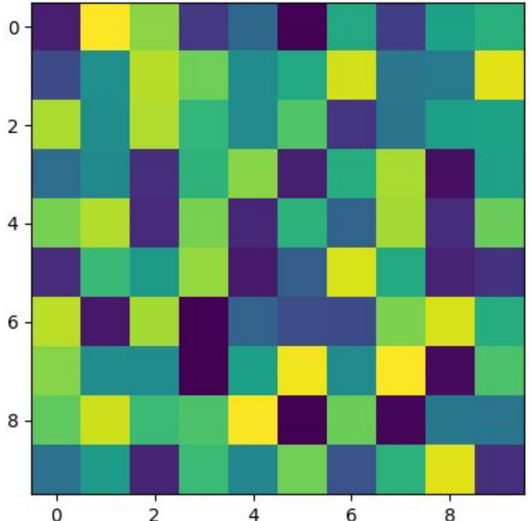
2 Different type of visualization

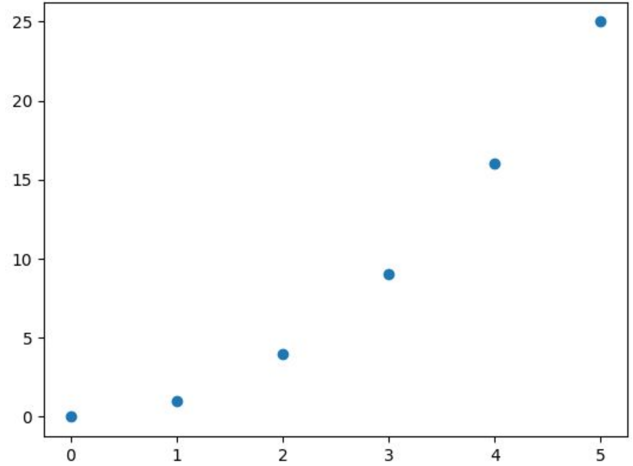


```python
import matplotlib.pyplot as plt
import numpy as np

data = np.random.rand(10, 10)

plt.imshow(data)
plt.show()
```



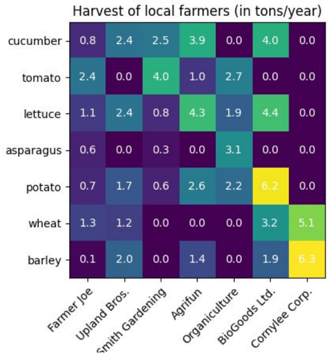
```python
import matplotlib.pyplot as plt


plt.scatter([0, 1, 2, 3, 4, 5], [0, 1, 4, 9, 16, 25])
plt.show()
```

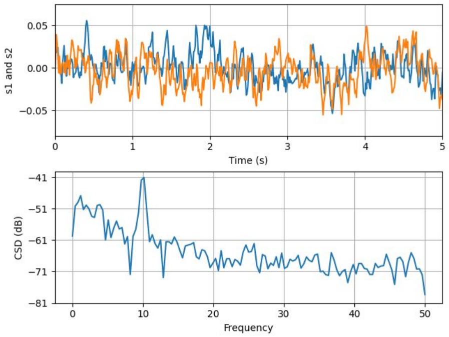


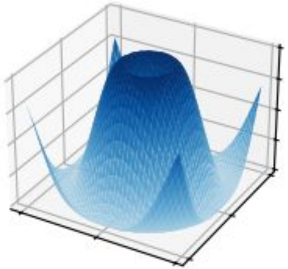


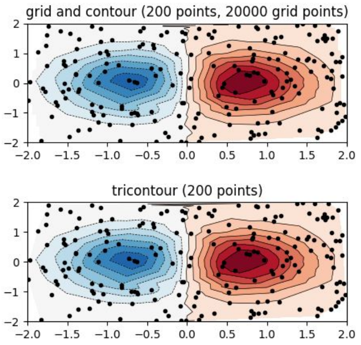
2 Multiple examples or tutorials

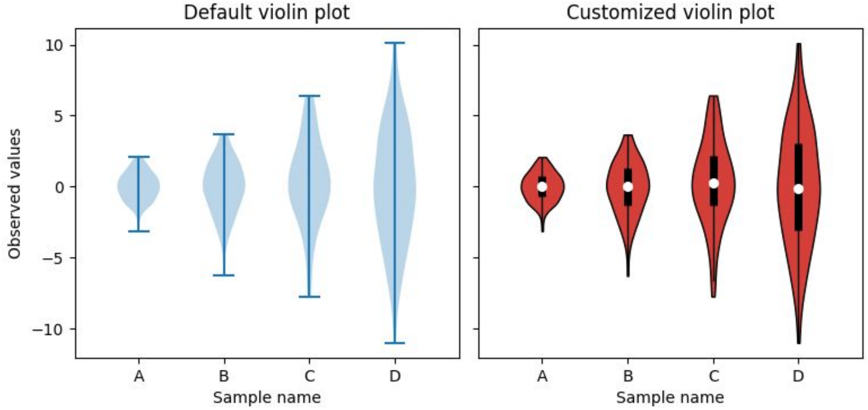
The gallery on Matplotlib website can give you great ideas.

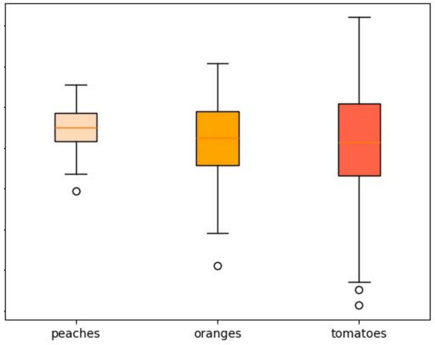


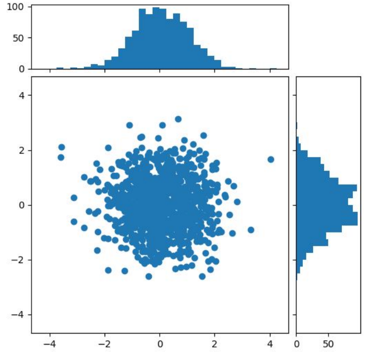




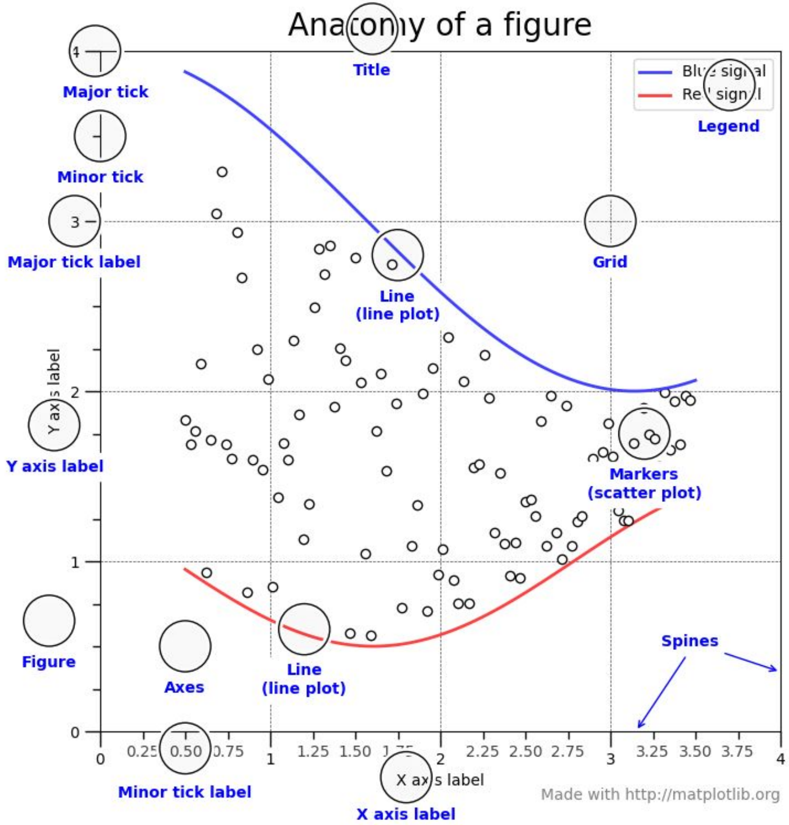








2 Anatomy of a figure



1 Initialize

```
import numpy as np
import matplotlib.pyplot as plt
```

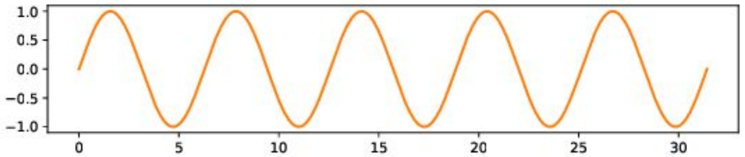
2 Prepare

```
X = np.linspace(0, 10*np.pi, 1000)
Y = np.sin(X)
```

3 Render

```
fig, ax = plt.subplots()
ax.plot(X, Y)
plt.show()
```

4 Observe



Part 3

1. Import and manipulate data

   Pandas and NumPy modules

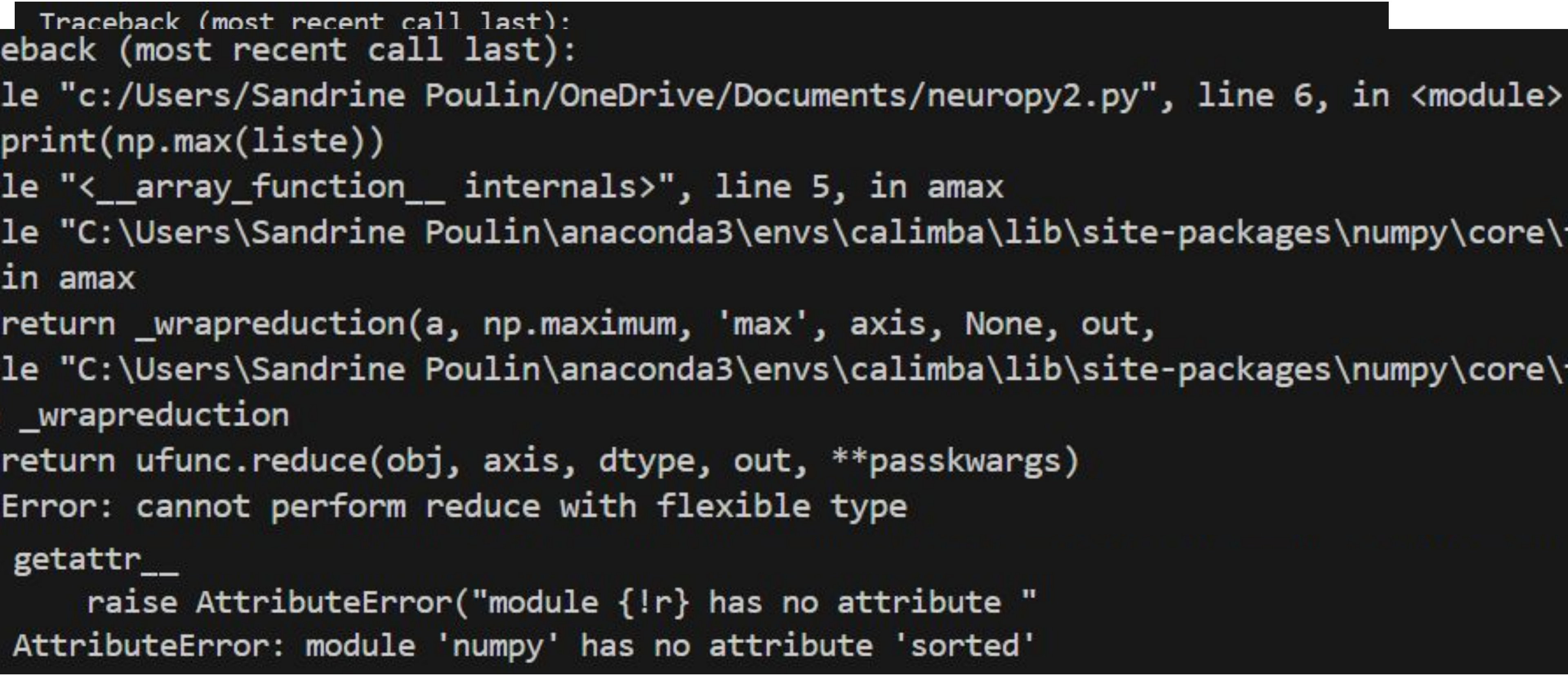
2. Visualise data

   Matplotlib and Seaborn module

3. **Troubleshooting**

   Resources available

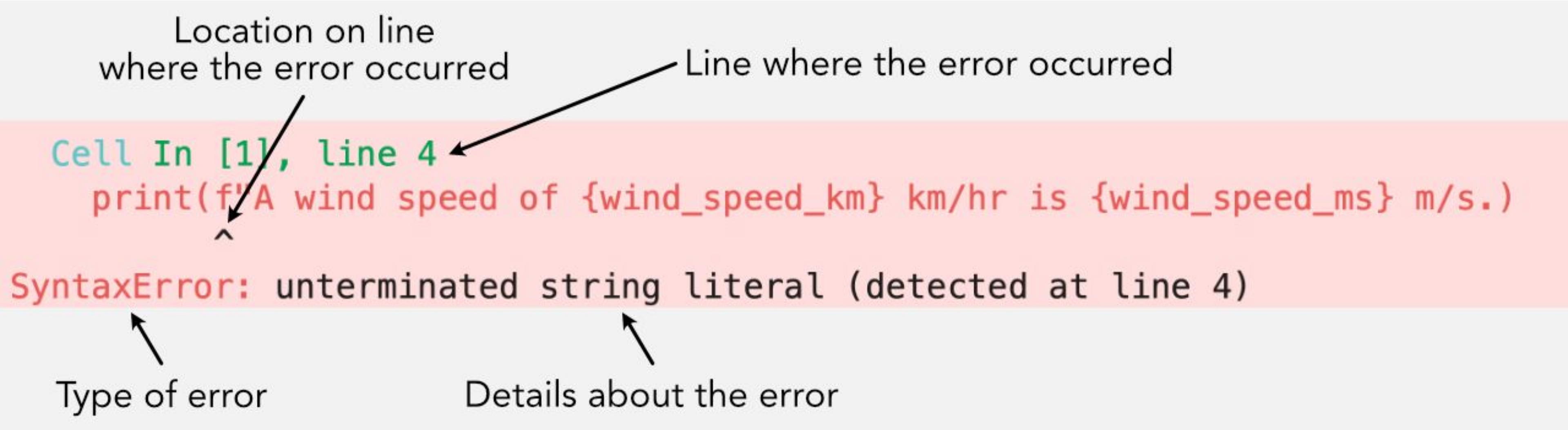
3 Troubleshooting



```
Traceback (most recent call last):
eback (most recent call last):
le "c:/Users/Sandrine Poulin/OneDrive/Documents/neuropy2.py", line 6, in <module>
print(np.max(liste))
le "<__array_function__ internals>", line 5, in amax
le "C:\Users\Sandrine Poulin\anaconda3\envs\calimba\lib\site-packages\numpy\core\
in amax
return _wrapreduction(a, np.maximum, 'max', axis, None, out,
le "C:\Users\Sandrine Poulin\anaconda3\envs\calimba\lib\site-packages\numpy\core\
 _wrapreduction
return ufunc.reduce(obj, axis, dtype, out, **passkwargs)
Error: cannot perform reduce with flexible type
 getattr__
    raise AttributeError("module {!r} has no attribute "
 AttributeError: module 'numpy' has no attribute 'sorted'
```

3 Troubleshooting

Anatomy of a python error:



```
  Cell In [1], line 4
    print(f"A wind speed of {wind_speed_km} km/hr is {wind_speed_ms} m/s.)
         ^
SyntaxError: unterminated string literal (detected at line 4)
```

Location on line where the error occurred
Line where the error occurred
Type of error
Details about the error

https://geo-python-site.readthedocs.io/en/latest/notebooks/L6/errors.html

3 Online resources

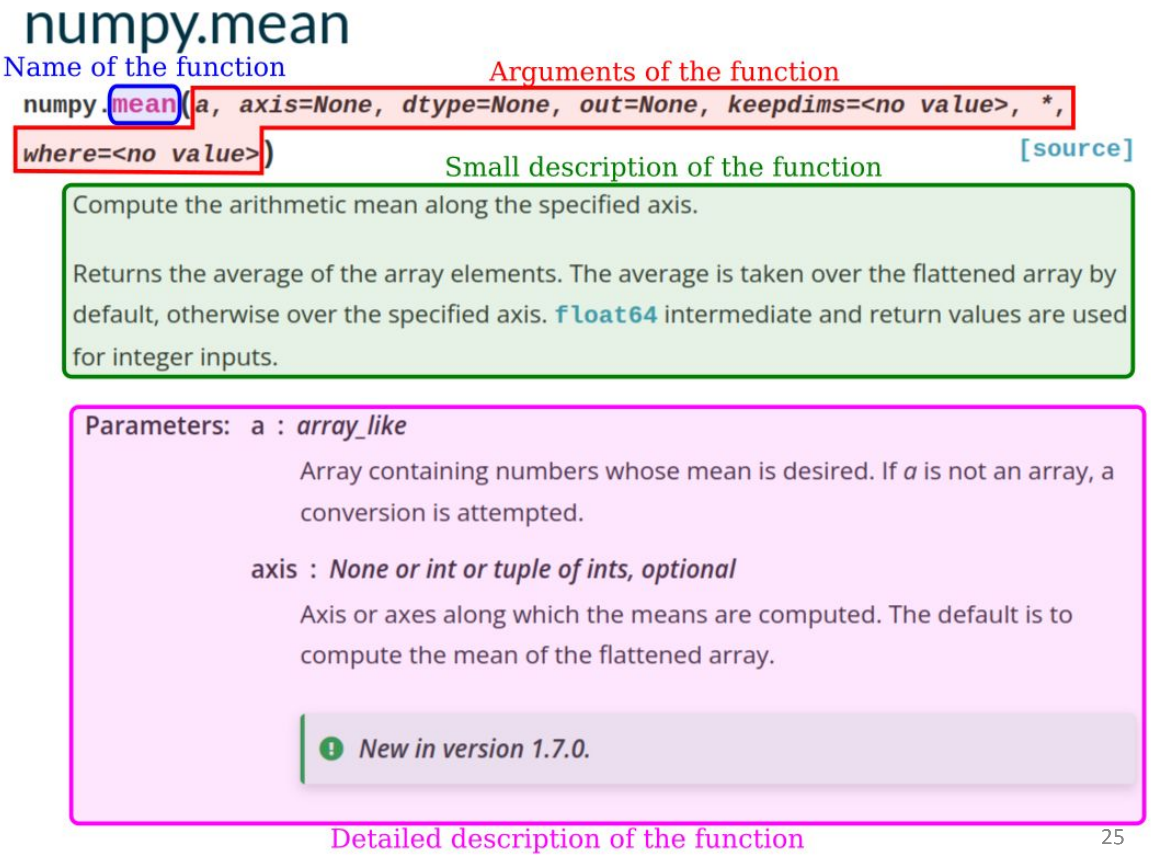
What are the online resources that can help you?

- Stackoverflow
  - Forum that is massively used by the community.
  - Someone already had your question.
- The documentation
  - The most important resource.
  - Learning to read the documentation will help you in the long term.
- ChatGPT
  - Generative AI, useful to get ideas and optimise your code.
  - Always double check.

3 How to read the documentation

The documentation is available

- on their website
- in your IDE (hold the cursor over the function).



numpy.mean

Name of the function

Arguments of the function

```
numpy.mean(a, axis=None, dtype=None, out=None, keepdims=<no value>, *,
where=<no value>)
```

[source]

Small description of the function

Compute the arithmetic mean along the specified axis.

Returns the average of the array elements. The average is taken over the flattened array by default, otherwise over the specified axis. `float64` intermediate and return values are used for integer inputs.

Parameters: a : *array_like*

Array containing numbers whose mean is desired. If *a* is not an array, a conversion is attempted.

axis : *None or int or tuple of ints, optional*

Axis or axes along which the means are computed. The default is to compute the mean of the flattened array.

New in version 1.7.0.

Detailed description of the function

3 How to read a Stackoverflow question

Stackoverflow is the main website for all of your questions:

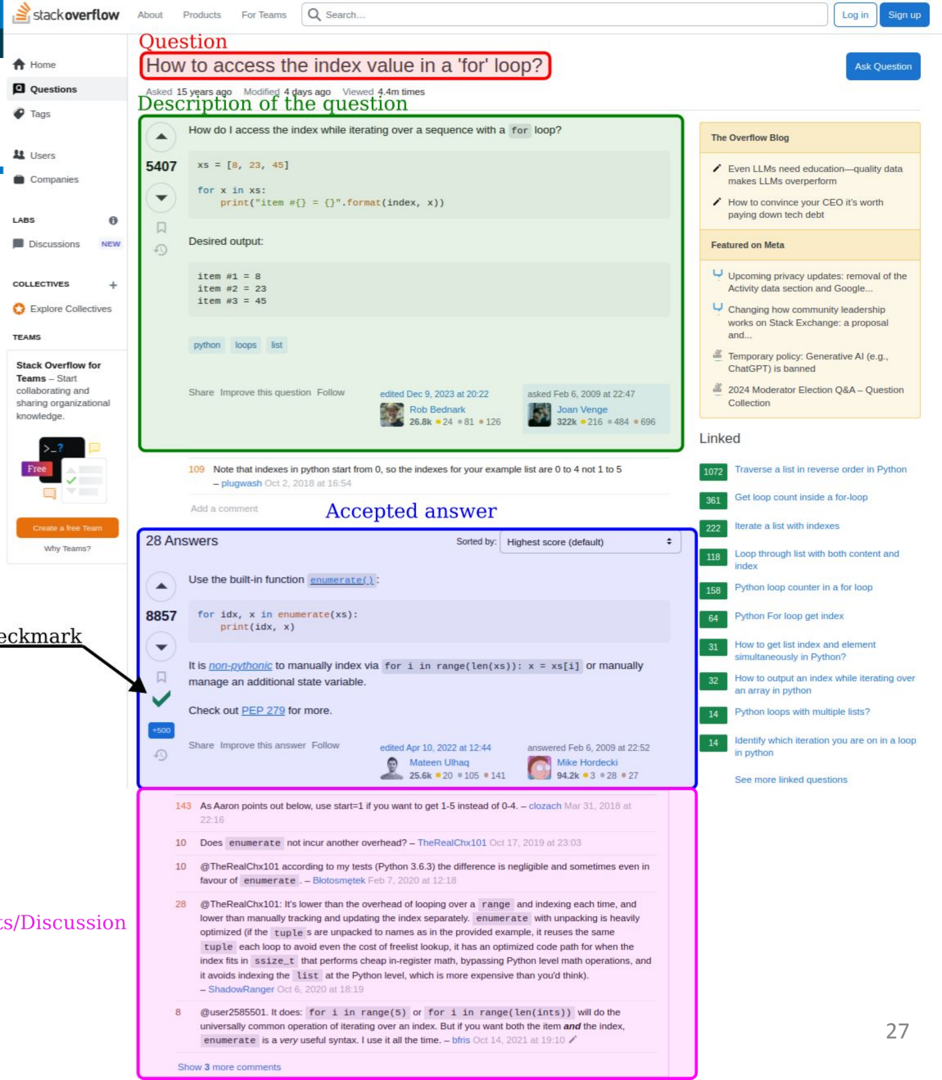
- Over 2 186 475 questions tagged with *Python* (from 27/02/2024).
- Multiple people can propose answer. The accepted answer have a green checkmark.
- The first post is the question (pssst… the code is not working).
- Multiple proposed answer can work.

stack overflow

3 Example with Stackoverflow

Link of this thread

Notice the details of some answers and how active this old question is!



Stack Overflow for Teams – Start collaborating and sharing organizational knowledge.

Question

How to access the index value in a 'for' loop?

Asked 15 years ago Modified 4 days ago Viewed 4.4m times

Description of the question

5407

How do I access the index while iterating over a sequence with a `for` loop?

```
xs = [8, 23, 45]

for x in xs:
    print("item #{} = {}".format(index, x))
```

Desired output:

```
item #1 = 8
item #2 = 23
item #3 = 45
```

python loops list

Share Improve this question Follow — edited Dec 9, 2023 at 20:22 Rob Bednark 26.8k ●24 ●81 ●126 — asked Feb 6, 2009 at 22:47 Joan Venge 322k ●216 ●484 ●696

109 Note that indexes in python start from 0, so the indexes for your example list are 0 to 4 not 1 to 5 – plugwash Oct 2, 2018 at 16:54

Add a comment

Accepted answer

28 Answers — Sorted by: Highest score (default)

8857

Use the built-in function `enumerate()`:

```
for idx, x in enumerate(xs):
    print(idx, x)
```

It is *non-pythonic* to manually index via `for i in range(len(xs)): x = xs[i]` or manually manage an additional state variable.

Check out PEP 279 for more.

Checkmark

✓ +500

Share Improve this answer Follow — edited Apr 10, 2022 at 12:44 Mateen Ulhaq 25.6k ●20 ●105 ●141 — answered Feb 6, 2009 at 22:52 Mike Hordecki 94.2k ●3 ●28 ●27

Comments/Discussion

143 As Aaron points out below, use start=1 if you want to get 1-5 instead of 0-4. – clozach Mar 31, 2018 at 22:16

10 Does `enumerate` not incur another overhead? – TheRealChx101 Oct 17, 2019 at 23:03

10 @TheRealChx101 according to my tests (Python 3.6.3) the difference is negligible and sometimes even in favour of `enumerate`. – Błotosmętek Feb 7, 2020 at 12:18

28 @TheRealChx101: It's lower than the overhead of looping over a `range` and indexing each time, and lower than manually tracking and updating the index separately. `enumerate` with unpacking is heavily optimized (if the `tuple`s are unpacked to names as in the provided example, it reuses the same `tuple` each loop to avoid even the cost of freelist lookup, it has an optimized code path for when the index fits in `ssize_t` that performs cheap in-register math, bypassing Python level math operations, and it avoids indexing the `list` at the Python level, which is more expensive than you'd think). – ShadowRanger Oct 6, 2020 at 18:19

8 @user2585501. It does: `for i in range(5)` or `for i in range(len(ints))` will do the universally common operation of iterating over an index. But if you want both the item ***and*** the index, `enumerate` is a *very* useful syntax. I use it all the time. – bfris Oct 14, 2021 at 19:10

Show 3 more comments

The Overflow Blog

- Even LLMs need education—quality data makes LLMs overperform
- How to convince your CEO it's worth paying down tech debt

Featured on Meta

- Upcoming privacy updates: removal of the Activity data section and Google...
- Changing how community leadership works on Stack Exchange: a proposal and...
- Temporary policy: Generative AI (e.g., ChatGPT) is banned
- 2024 Moderator Election Q&A – Question Collection

Linked

1072	Traverse a list in reverse order in Python
361	Get loop count inside a for-loop
222	Iterate a list with indexes
118	Loop through list with both content and index
158	Python loop counter in a for loop
64	Python For loop get index
31	How to get list index and element simultaneously in Python?
32	How to output an index while iterating over an array in python
14	Python loops with multiple lists?
14	Identify which iteration you are on in a loop in python

See more linked questions

4 Final tips

- **Modules** are there to **help you save time**, use them well!

- Pay attention to the **error messages** Python provides.
    - They often give you a clue about what went wrong and where. You can often copy paste the message to get help on stack overflow.

- When trying to find your mistake, use **print() statements** to check the values of variables at different points in your code.

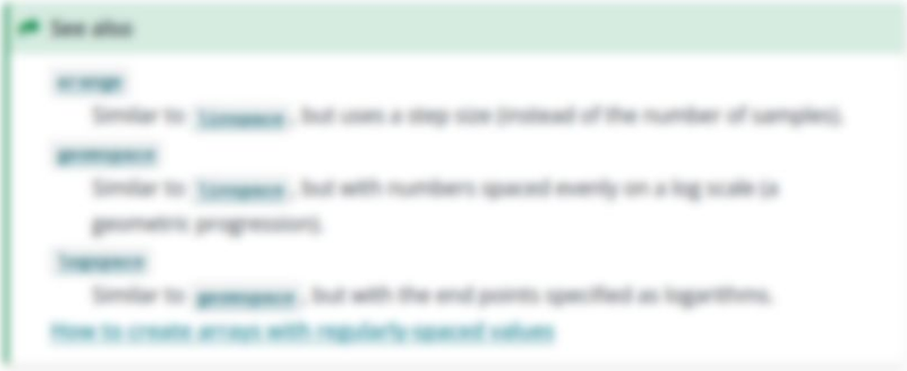
- **Comment. Your. Code.**

4

Time for exercises!

Find the documentation for the;

- numpy linspace() function.
  - What is the purpose of this function?
  - How many point will this function generate by default?
  - By default, is the last point included in the array?
  - If you wanted to generate an array with fixed step size, how would you do so? (pssst, what's with the blurry picture of the documentation???)



See also

arange
Similar to linspace, but uses a step size (instead of the number of samples).

geomspace
Similar to linspace, but with numbers spaced evenly on a log scale (a geometric progression).

logspace
Similar to geomspace, but with the end points specified as logarithms.

How to create arrays with regularly spaced values

4 Time for exercises!

Sides:

Google colab: