L1 : Introduction to Python : first steps

Michael Graupner

SPPIN – Saint-Pères Institute for the Neurosciences

Université Paris Cité, CNRS

How do YOU deal with data ?

I give you a 2d array of numbers : measurements and time points (*N=1000*).

```
time          0.1 0.2 0.3 0.4 0.5 0.6 ... 99.8 99.9 100.0
measurment    3.2 4.3 3.8 4.5 3.7 5.1 ... 8.3  8.1  9.0
```

- I would like to know the mean and the standard deviation of the measurements.

- I would like to see the data displayed, i.e., plotted as measurement vs time.
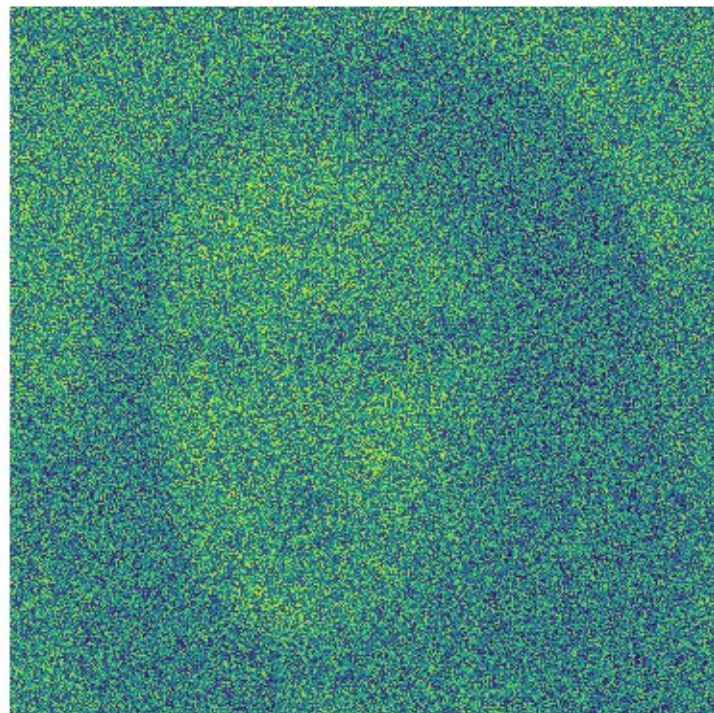
→ **How would you do that today ?**

```
 942  509  688 1093  703  416  664  762  664  822  457  400  614  648  243  218  852  459  524  660
 158  503  226 1016  322  632  168  419 1015  772  893  774  765  237  161  575  351  244  806  584
 692  817  305  168  139  693  507 1121  868  181  943  310  622  345  391  609 1060 1058 1002  401
1050  826  399 1002 1031  140  310  750  394  924  321  425  524  225  428  633  439  686  587  413
 787 1103 1012  157  839  632  917  256  677  631  419 1014  239  398  951  223 1042  637  577  843
 781  512  994  343 1084  941  343  632  234  518  850 1021  367  310  726  773  672  493  914  506
 516  240  575  790  159  657  983  587  933  170 1010 1106  601  868  872  337 1032  533  452  963
 464  902  456  976  943  279  797 1008  341  679  585 1011  569  584  775  740  505  497  690  944
 179  295  928  549  593  850  375 1068  343  436  535  171  931  557  382 1083  781  536  902  781
1119  648  590  617  221  622  149  978  487  154  763 1088  942  941 1032  411  642  912  199  958   ...
 443  150  619  614  747  452  415  502  240  767  160 1066  907  978 1043  382  240  563  939  904
 278  684 1080  582  291  580 1100  701  957  287  650  453  573  730  714  412  311  407 1048  275
 813  520  997  277  308 1069 1082  235  230 1031  972  658 1033  805  869  344  228  586  144  665
1027  540 1067  293  851  269  274  435 1069  869  941  208  163 1079  887  974  265  733  223 1009
 610  546  628  732  218  184  827  412  942  492  624  348  950  676  471 1114  515  355  821  264
 621  328  734  905  340  875 1056 1102 1038  130  338  918  945  174  931 1005  884 1054  158  767
 511  209 1108  378 1064  478  203  739  868  615  541  641  697  571  920  430  908  299  666  638
 728  151  217  404  481  884  231  705  301  908  295 1091  865  169  806  757 1098  913 1082  942
 221  715  474  329  560  738 1044  672  531  412  224  284 1127  198  602  464  393  385  293  608
 778 1045 1005  689  328  718  395  202  656 1040  489  756  459  582 1010  951  442  731  540 1023
```
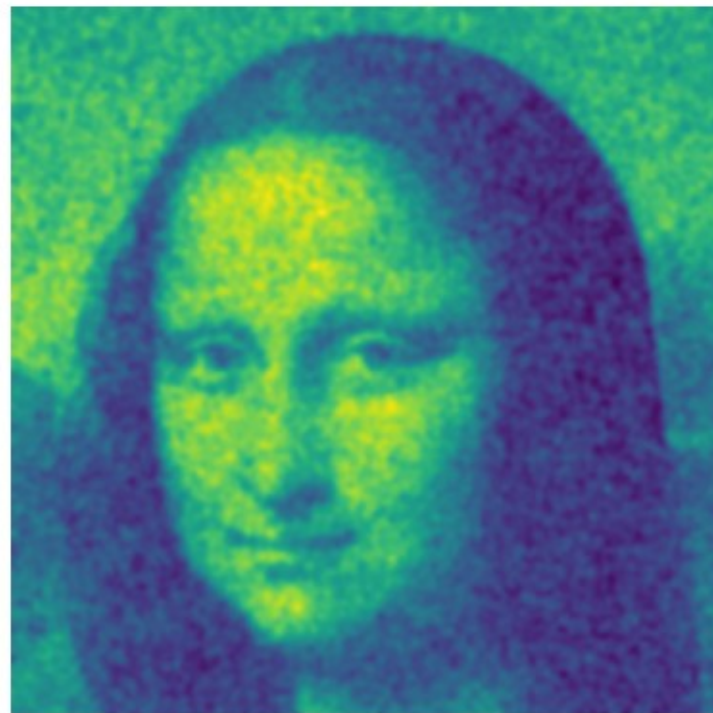
⋮

942 509 688 1093 703 416 664 762 664 822 457 400 614 648 243 218 852 459 524 660
158 503 226 1016 322 632 168 419 1015 772 893 774 765 237 161 575 351 244 806 584
692 817 305 168 139 693 507 1121 868 181 943 310 622 345 391 609 1060 1058 1002 401
1050 826 399 1002 1031 140 310 750 394 924 321 425
787 1103 1012 157 839 632 917 256 677 631 419 1014
781 512 994 343 1084 941 343 632 234 518 850 1021
516 240 575 790 159 657 983 587 933 170 1010 1106
464 902 456 976 943 279 797 1008 341 679 585 1011
179 295 928 549 593 850 375 1068 343 436 535 171
1119 648 590 617 221 622 149 978 487 154 763 1088
443 150 619 614 747 452 415 502 240 767 160 1066
278 684 1080 582 291 580 1100 701 957 287 650 453
813 520 997 277 308 1069 1082 235 230 1031 972 658
1027 540 1067 293 851 269 274 435 1069 869 941 208
610 546 628 732 218 184 827 412 942 492 624 348
621 328 734 905 340 875 1056 1102 1038 130 338 918
511 209 1108 378 1064 478 203 739 868 615 541 641
728 151 217 404 481 884 231 705 301 908 295 1091
221 715 474 329 560 738 1044 672 531 412 224 284
778 1045 1005 689 328 718 395 202 656 1040 489 756

⋮

2D plot

942 509 688 1093 703 416 664 762 664 822 457 400 614 648 243 218 852 459 524 660
158 503 226 1016 322 632 168 419 1015 772 893 774 765 237 161 575 351 244 806 584
692 817 305 168 139 693 507 1121 868 181 943 310 622 345 391 609 1060 1058 1002 401
1050 826 399 1002 1031 140 310 750 394 924 321 425
787 1103 1012 157 839 632 917 256 677 631 419 1014
781 512 994 343 1084 941 343 632 234 518 850 1021
516 240 575 790 159 657 983 587 933 170 1010 1106
464 902 456 976 943 279 797 1008 341 679 585 1011
179 295 928 549 593 850 375 1068 343 436 535 171
1119 648 590 617 221 622 149 978 487 154 763 1088
443 150 619 614 747 452 415 502 240 767 160 1066
278 684 1080 582 291 580 1100 701 957 287 650 453
813 520 997 277 308 1069 1082 235 230 1031 972 658
1027 540 1067 293 851 269 274 435 1069 869 941 208
610 546 628 732 218 184 827 412 942 492 624 348
621 328 734 905 340 875 1056 1102 1038 130 338 918
511 209 1108 378 1064 478 203 739 868 615 541 641
728 151 217 404 481 884 231 705 301 908 295 1091
221 715 474 329 560 738 1044 672 531 412 224 284
778 1045 1005 689 328 718 395 202 656 1040 489 756

:

smoothing + 2D plot

Python code for the above operations

Python code

```python
img = plt.imread('image-noise.tif')

imgNew = gaussian_filter(img, sigma=10)

ax.imshow(imgNew)
```

→ read image

→ apply Gaussian filter

→ plot/display image

What is Python ?
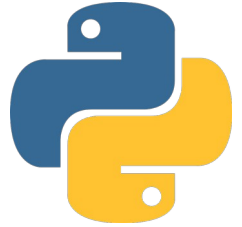
- modern programming language (since 1991)
- interpreted language (no compilation necessary)
- emphasis is put on the readability of the code
- concepts can be expressed in less lines compared to C/C++ or Java
- extensive libraries available
- build-in visualization

Python - modern programming language

Most popular programming languages in 2021

Rank	Language	Type				Score
1	Python ⌄	🌐		🖥	⚙	100.0
2	Java ⌄	🌐	📱	🖥		95.4
3	C ⌄		📱	🖥	⚙	94.7
4	C++ ⌄		📱	🖥	⚙	92.4
5	JavaScript ⌄	🌐				88.1
6	C# ⌄	🌐	📱	🖥	⚙	82.4
7	R ⌄			🖥		81.7
8	Go ⌄	🌐		🖥		77.7
9	HTML ⌄	🌐				75.4
10	Swift ⌄		📱	🖥		70.4
11	Arduino ⌄				⚙	68.4
12	Matlab ⌄			🖥		68.3

[Source : IEEE Spectrum]

Clear and readable syntax → easy to learn

```
In [1]:    1  # import modules
           2  import numpy as np
           3
           4  # function declaration
           5  def update_values(x):
           6      return x+1
           7
           8  x = 1
           9  if x>0:
          10      print('Hello World!')
          11      x = update_values(x)
          12
          13  print(x)
```

```
Hello World!
2
```

Python - free and easy to learn
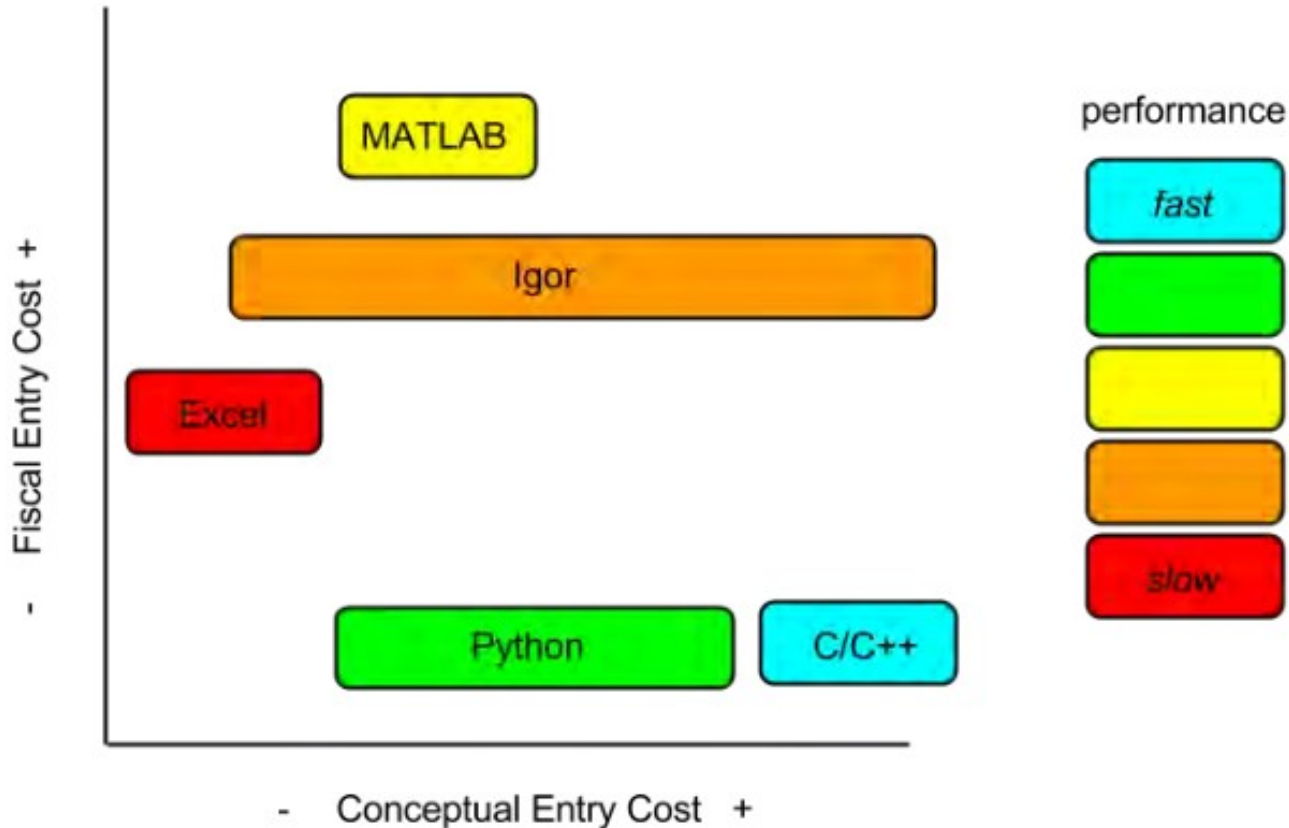
Extensive standard and third-party libraries

- **wxPython :** graphical toolbox library for GUI development

- **SymPy :** library for symbolic mathematics : can do algebraic evaluations, differentiation, expansions, complex numbers, etc.

- **Pygame :** library for 2D game development

- **Twisted :** major tool for development of network applications

- **OpenCV :** library for extensive computer vision applications

Python modules for Neuroscience applications

- simulators and simulator interfaces

- data collection and analysis

- sharing, re-use, storage and databasing of data and models

- stimulus generation

- parameter search and optimization

- visualization

- machine learning : GPU-based computing

BRIAN

spiking neural network simulator

Excitatory   Inhibitory

randomly connected,
recurrent network of
excitatory and inhibitory
neurons
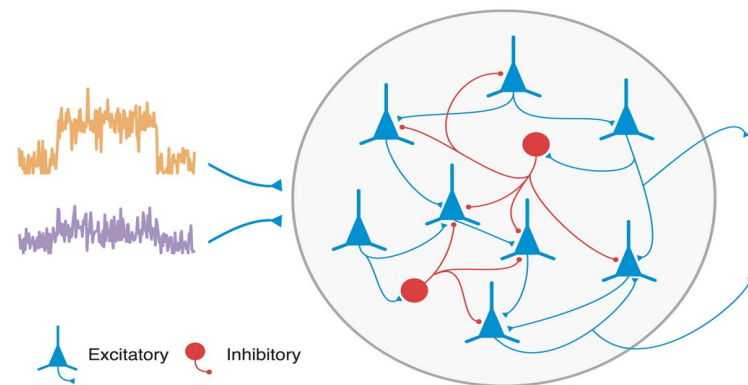
```
1   from brian import *
2   eqs = '''
3   dv/dt = (ge+gi-(v+49*mV))/(20*ms) : volt
4   dge/dt = -ge/(5*ms) : volt
5   dgi/dt = -gi/(10*ms) : volt
6   '''
7   P = NeuronGroup(4000, eqs, threshold=-50*mV, reset=-60*mV)
8   P.v = -60*mV+10*mV*rand(len(P))
9   Pe = P.subgroup(3200)
10  Pi = P.subgroup(800)
11  Ce = Connection(Pe, P, 'ge', weight=1.62*mV, sparseness=0.02)
12  Ci = Connection(Pi, P, 'gi', weight=-9*mV, sparseness=0.02)
13  M = SpikeMonitor(P)
14  run(1*second)
15  raster_plot(M)
16  show()
```
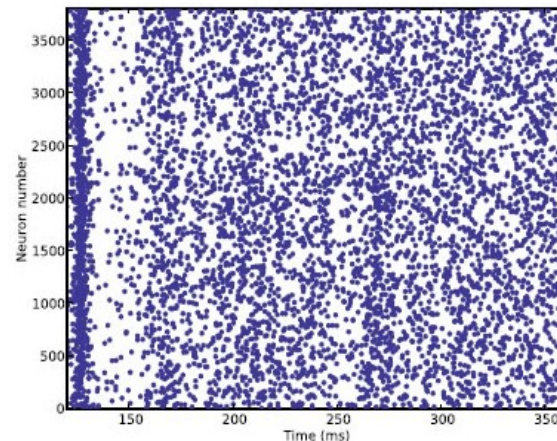
Python in Neuroscience : machine learning

Machine Learning in Python. Simple and efficient tools for data mining and data analysis

e.g. classification using several classifiers
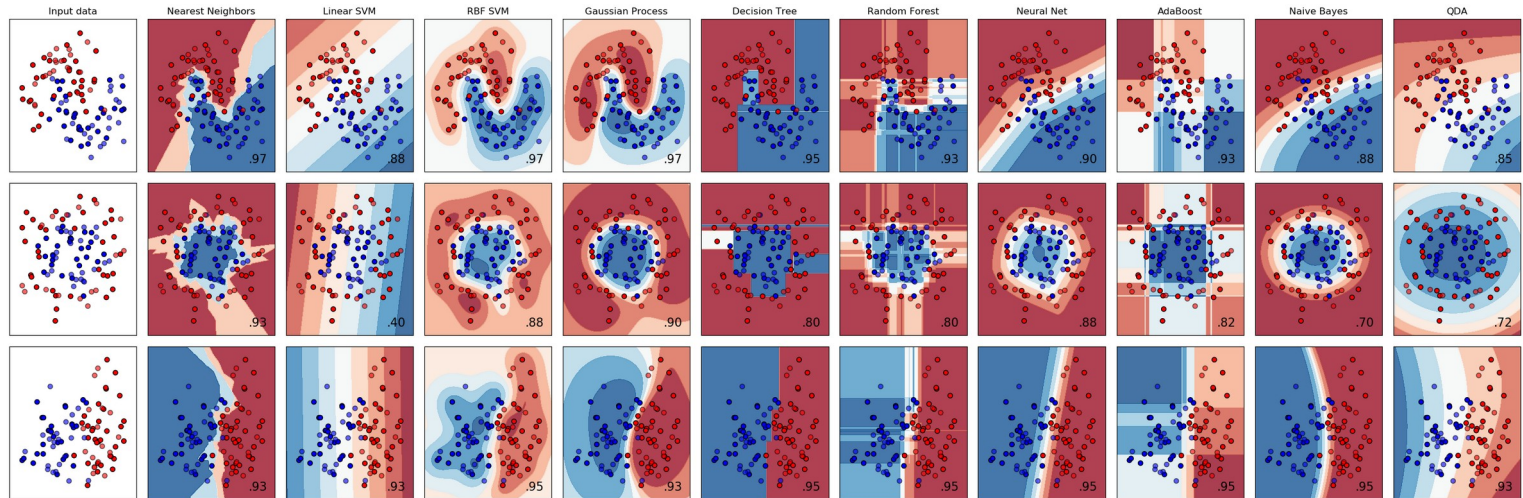
Python in Neuroscience : visualization

e.g. matplotlib library
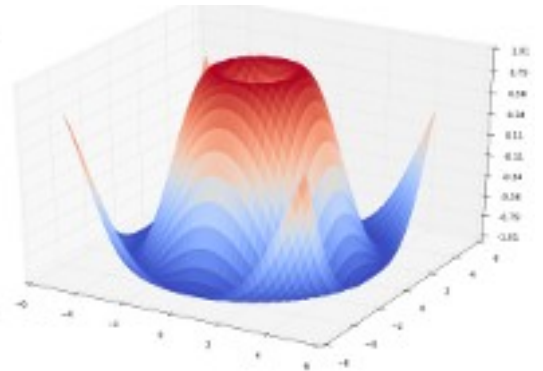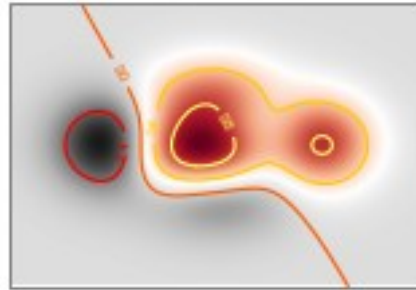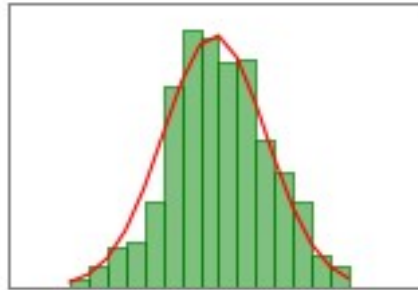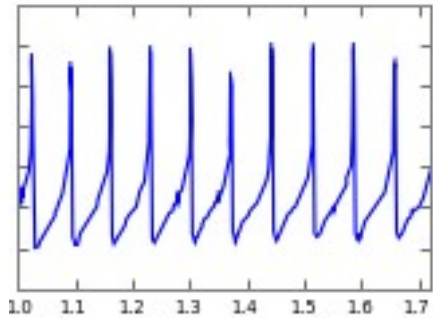
Python in Neuroscience : deep learning/networks

simulate multi-layer networks for deep-learning applications
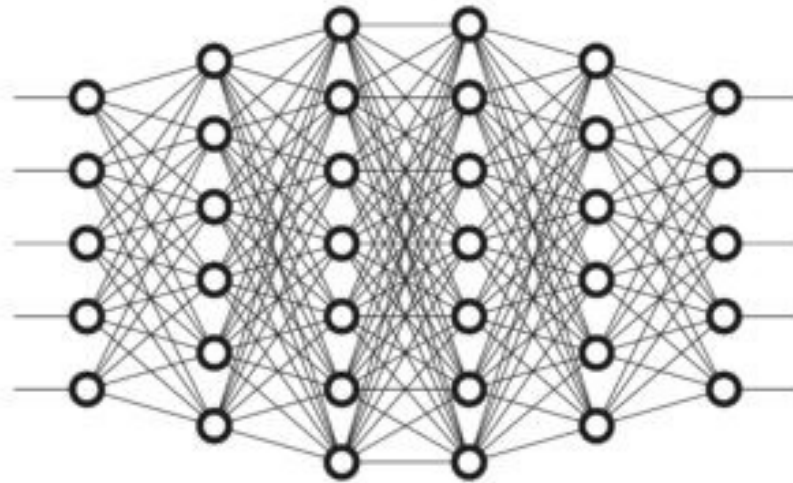
Python in Neuroscience : stimulus generation

e.g. Vision EGG, or PsychoPy

Vison Egg multi stimulus demo - Press any key to quit

A — Sin Grating 2D (color)
B — Spinning Drum
C — put_pixels()
D — Dot Area 2D
E — Sin Grating 2D (gabor)
F — put_new_framebuffer()
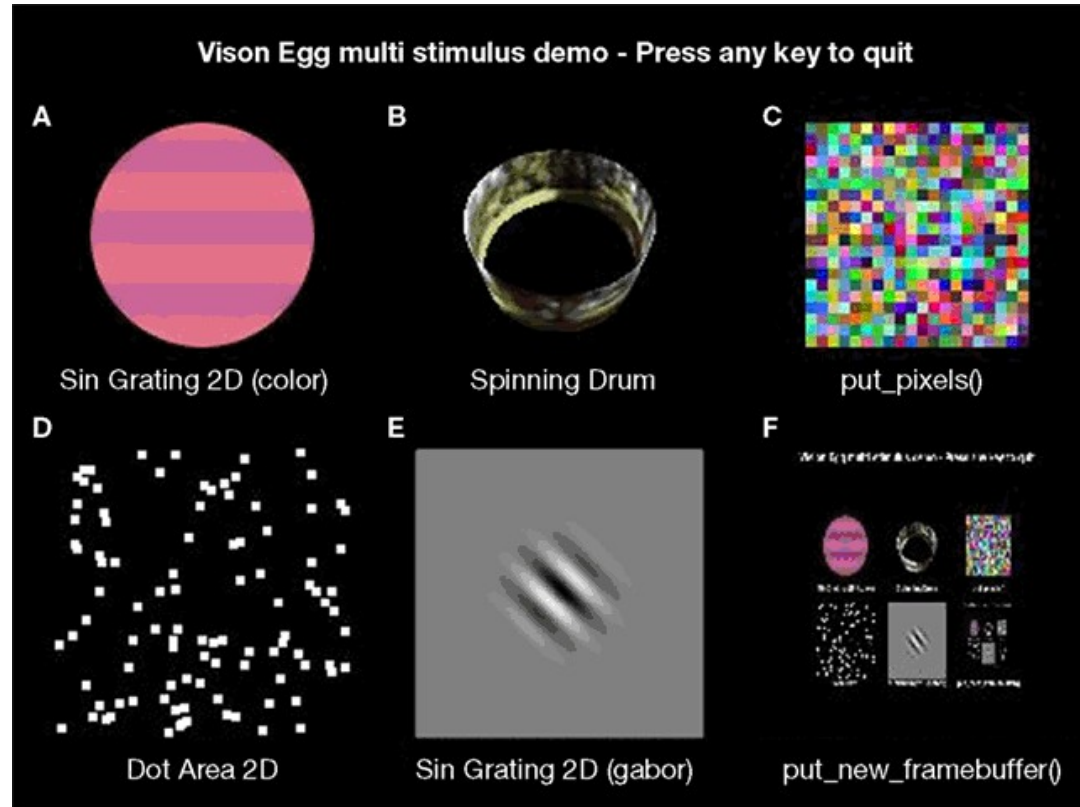
Getting started : Python installation

- Windows, Mac OS X, Linux (distributions for package handling)
  - **Anaconda** from Continuum Analaytics : https://www.continuum.io/downloads
  - Enthought Python : https://www.enthought.com/
  - Python(x,y) : http://python-xy.github.io/

- Debian + Ubuntu Linux

```
$ apt-get install python-numpy python-scipy python-matplotlib \
  ipython
```

- Mac OS X : Install Fink, then

```
$ fink install scipy-core-py25 scipy-py25 matplotlib-py25 ipython-py25
```

- Note that "$" symbolizes command line prompt, it is not part of the command

Getting started : interpreter and IDEs

- **ipython**

  - command line interpreter : interactive shell for enhanced introspection, code highlighting and tab completion

- **Jupyter Notebook**

  - command line interpreter in the browser
  - combines code execution, rich text, and visualizations

- **Spyder** (IDE): Scientific PYthon Development EnviRonment

- **PyCharm** (IDE): code development environment
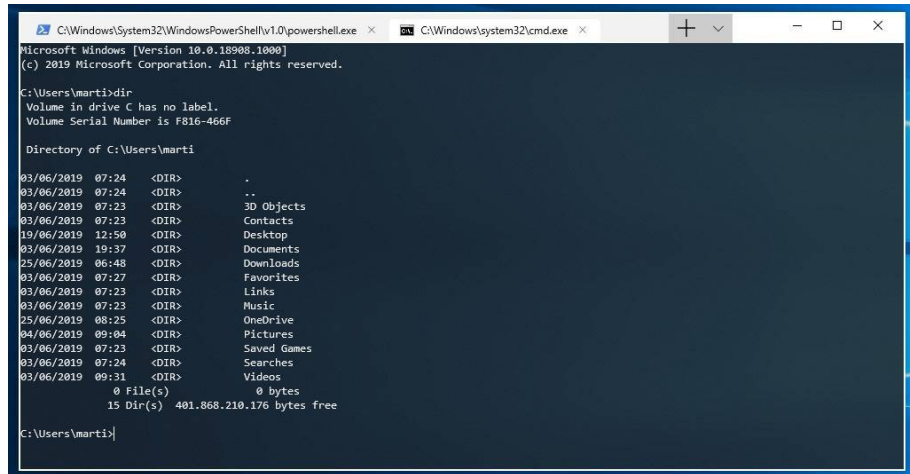
  IDE ... Integrated Development Environment

Everything starts with the Terminal

- **What is a Terminal ?**

  - Terminal, also known as command line, console or shell

  - allow to accomplish tasks on a computer without the use of graphical user interfaces (GUIs)

  - simple text commands are used to accomplish tasks

Windows terminal (command line, Powershell)   **Linux terminal**

iPython

- Started by typing
  and executing (by pressing
  enter) **ipython** in the
  terminal application

  `$ ipython`

- useful for short
  explorations
- tab completion!


```
(locorungs) mgraupe@thinkpadX1B:~> ipython
Python 3.6.7 (default, Apr 19 2019, 16:04:00)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.5.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import numpy as np

In [2]: a = np.arange(20)

In [3]: print(a)

[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]

In [4]:
```

Jupyter Notebook
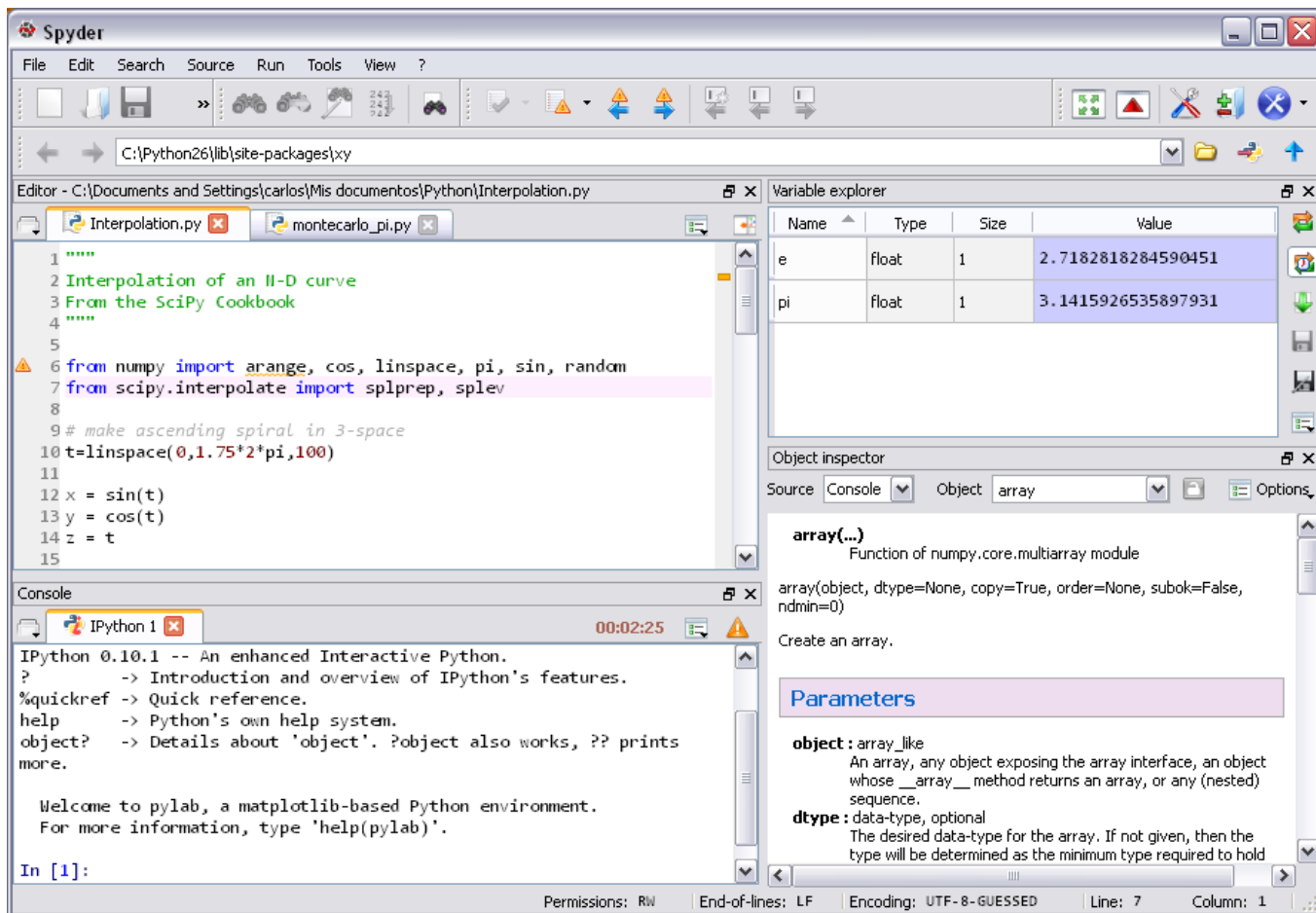
- Started by typing and executing

  (by pressing *enter*)
  jupyter-notebook in the

  terminal application :

  ```
  $ jupyter-notebook
  ```

  - launched and accessed in a
  browser (firefox, chrome,
  safari ...) window

Spyder

PyCharm
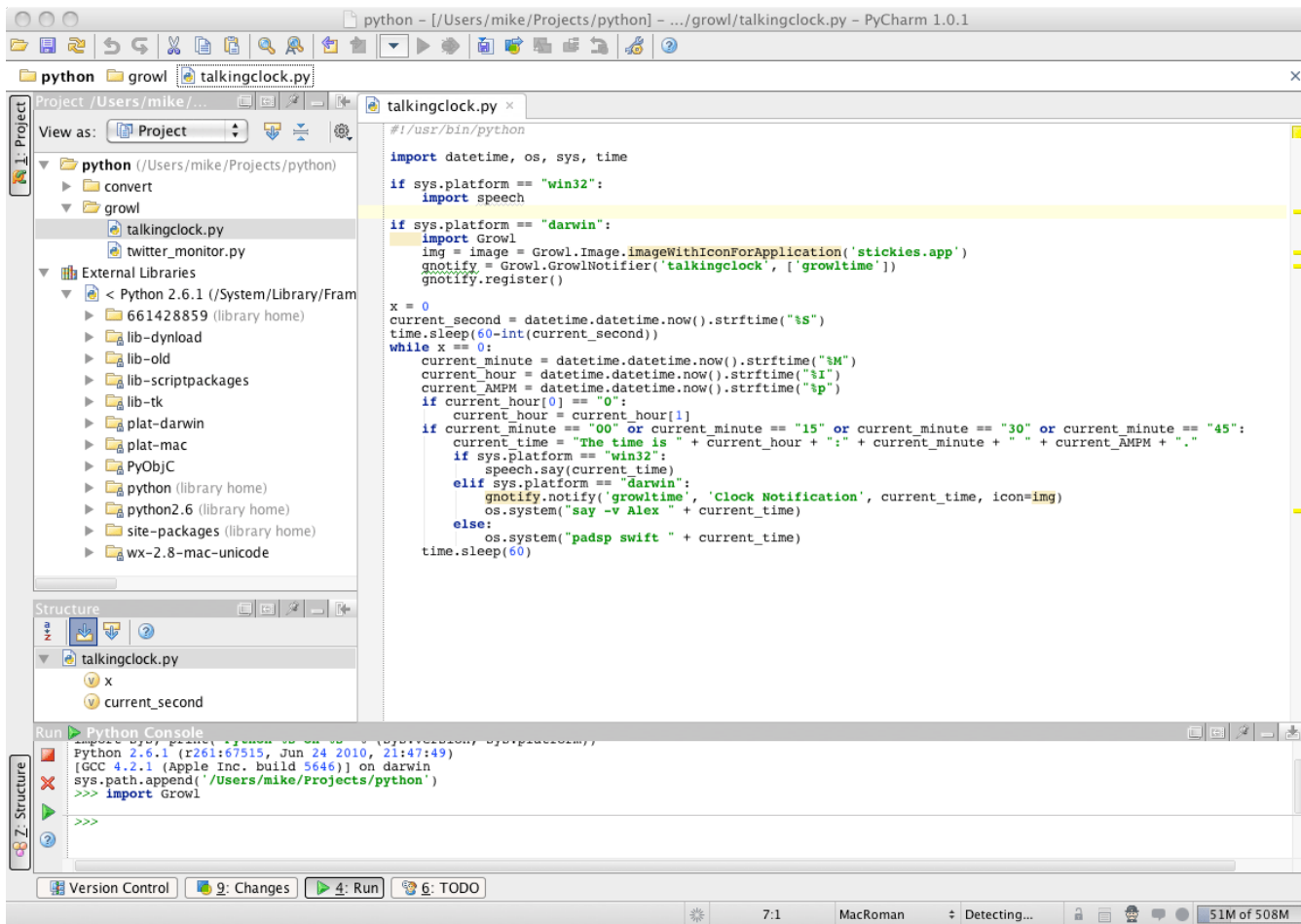
Executing Python programs

- Python programs can be run either interactively or as scripts stored in a file
- An interpreter is started by calling **ipython** (or plain **python, or jupyter-notebook**)

```
mgraupe@atp:~$ ipython3
Python 3.5.7 (default, Apr  4 2019, 12:02:34)
Type "copyright", "credits" or "license" for more information.

In [1]: print('Hello World!')
Hello World!
In [2]: x = 3
In [3]: print(x+5)
8
In [4]: exit
mgraupe@atp:~$
```

- Scripts are supplied as arguments to the interpreter

```
mgraupe@atp:~$ python hello_world.py
Hello world!
```

Online resources : introductions and references

- **W3Schools - Python Tutorials :**
  https://www.w3schools.com/python/default.asp

- The Python documentation index :
  https://docs.python.org/3.6/

- Python library reference :
  https://docs.python.org/3.6/library/

- Dive into Python :
  http://histo.ucsf.edu/BMS270/diveintopython3-r802.pdf

- Activestate Python [popular Python recipes] :
  http://code.activestate.com/recipes/langs/python/

- Python tutorial :
  https://docs.python.org/3.6/tutorial/index.html

- Numpy tutorial :
  http://www.time.mk/trajkovski/teaching/imi/2010-fall/NumPy/Tentative%20NumPy%20Tutorial%20-.html

- Scipy reference :
  http://docs.scipy.org/doc/scipy/reference/genindex.html

Online resources : general

- a simple Google search :

  - use the keyword "python"

  - specify your operating system (*window, linux, mac*) for package installation, importing queries

  - use the best matching terminology for code questions

  - common sites for useful help : *stackoverflow*, *askubuntu*, *github*

Online resources : ask ChatGPT for code

GR How can I create an array and plot it in python?

Attention! Code can be erroneous.

⬡ :

1. Install required libraries (if not already installed):
   You can install NumPy and Matplotlib using pip if you haven't already:

   ```bash
   pip install numpy matplotlib
   ```

2. Import the necessary libraries:
   In your Python script or Jupyter Notebook, import NumPy and Matplotlib:

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

3. Create an array:
   You can create a NumPy array by using the `np.array()` function. For example, let's create a simple array of numbers:

   ```python
   # Create an array of numbers from 0 to 9
   my_array = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
   ```

4. Plot the array:
   To plot the array, you can use Matplotlib's `plt.plot()` function. Here's a basic example:

   ```python
   # Plot the array
   plt.plot(my_array)

   # Add labels and title (optional)
   plt.xlabel('X-axis')
   plt.ylabel('Y-axis')
   plt.title('Array Plot')

   # Show the plot
   plt.show()
   ```

   This code will create a simple line plot of your array.

Online resources : ask ChatGPT to interpret code

GR What does the following python code do ?

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

%matplotlib inline

t = np.linspace(0,10,1001)
tau = 2. y = np.exp(-t/tau)  # multiple commands on a line are separted by ;

plt.plot(t,y)
```

The provided Python code performs the following actions:

1. Imports the necessary libraries:
   - It imports NumPy as `np` and Matplotlib as `plt`. These libraries are commonly used for numerical operations and plotting, respectively.
2. `%matplotlib inline`:
   - This is a special Jupyter Notebook command that tells the notebook to display Matplotlib plots directly within the notebook rather than opening them in separate windows. It's not required in regular Python scripts, as it's specific to Jupyter Notebook.
3. Creates a time array `t`:
   - It generates an array `t` using `np.linspace()` that spans from 0 to 10 with 1001 evenly spaced points. This array represents a time series.
4. Defines a decay function `y`:
   - It calculates an exponential decay function `y` using the formula `y = np.exp(-t/tau)`. Here, `tau` is a time constant set to 2.0.
5. Plots `t` against `y`:
   - It uses `plt.plot(t, y)` to create a line plot of the time array `t` on the x-axis and the exponential decay values `y` on the y-axis.

Overall, this code generates an exponential decay curve and plots it using Matplotlib, with the resulting plot displayed in the Jupyter Notebook output cell (thanks to `%matplotlib inline`). The decay curve is based on the exponential decay function with a time constant `tau` of 2.0, and it covers the time range from 0 to 10.

Online resources : DataCamp

- an interactive to learn, practice and develop further your programming skills

- ideal to to dive into a programming language for starters (videos, extensive help, hints, small assignments with direct feedback)

- learn and extend your programming skills at your own pace

- a commercial website, you have access through this course : https://learn.datacamp.com/

Online resources : Neuroscience

- Front Neuroinform 2015 – *Python in Neuroscience*
  `http://journal.frontiersin.org/article/10.3389/fninf.2015.00011/full`

- BCCN cours - *Advanced Scientific Programming in Python* :
  `https://python.g-node.org/wiki/schedule`

- Brian simulator :
  `http://briansimulator.org/`

General reference for the course : Cheat Sheet
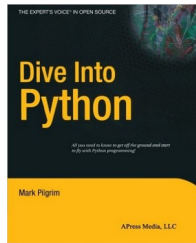

- Contains basic Python concepts and information

- Reference for often used code in class

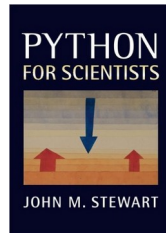# General Python books
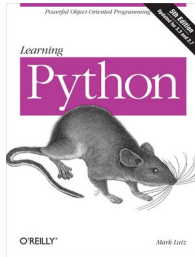
- Learning Python, 5th Edition
  Mark Lutz
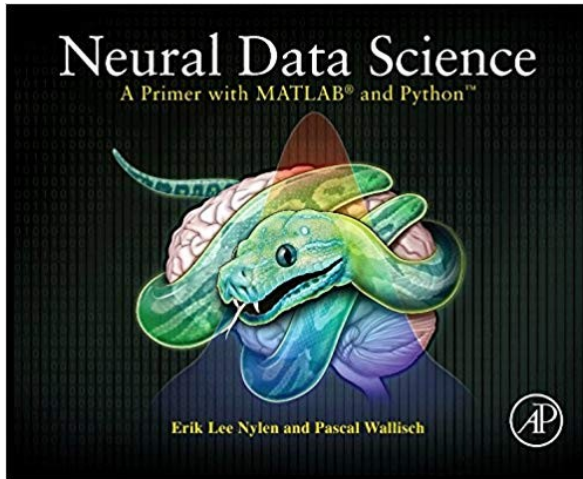 ISBN : 978-1-4493-5573-9

- Dive Into Python (3)
  Mark Pilgrim
  ISBN: 978-1590593561 (978-1430224150)

- Python for Scientists
  John M. Stewart
  ISBN: 978-1107686427

Neuroscience specific book

- Neural Data Science
  A primer with Matlab and Python
  Erik Lee Nylen (Author), Pascal Wallisch (Author)
  ISBN-10: 9780128040430

Workflow of the course lecture

1) **All course material** (.pdf file of lecture; .ipynb for tutorial; .ipynb for homework assignment) can be accessed on github (code repository site) :

   https://github.com/mgraupe/NeuralDataSciPy2024

2) **Visit course website** : Launch the browser (🌐 "Navigateur Web") and navigate to the link above

3) **Download lecture** : Click on lecture link and hit the **Download** button (file will be downloaded automatically to the downloads – Téléchargements – folder); annotate lecture PDF

Workflow of the course tutorial

1) **Tutorial material :** can be accessed on github, launch the browser (Navigateur Web) and got to:
   https://github.com/mgraupe/NeuralDataSciPy2024

2) **Save tutorial file** : Click on the tutorial link, hit the **Raw** button (the raw file content will be displayed),
   save (Enregister sous ...) the raw file to your computer
   Attention : make sure that the file ending remains **.ipynb**

3) **Launch jupyter-notebook :** Start the **terminal** application (▸_ "Emulateur de Terminal") on your
   computer, launch the notebook environment by typing and executing **$ jupyter-notebook** , the
   jupyter environment starts up in the browser in the directory in which it was started

4) **Load the jupyter-notebook file :** In the jupyter environment, you first see the directory and file
   structure on your computer (relative to the directory in which jupyter was started), **navigate** to
   the downloaded **.ipynb** file and click on it to launch it

5) **Start editing**

6) **You can follow the tutorial progress/coding of the lecturer on Deepnote**.
   allows you to copy code into your own Jupyter notebook.