Image analysis with Python

Michele Lanzetta

Compilers

Librerie, OpenCV

Free software installers (various OSs, Unix, Windows, Mac): http://python.org/

Web based compiler: https://jupyter.org/

The Jupyter Notebook (text + latex + programs + data)

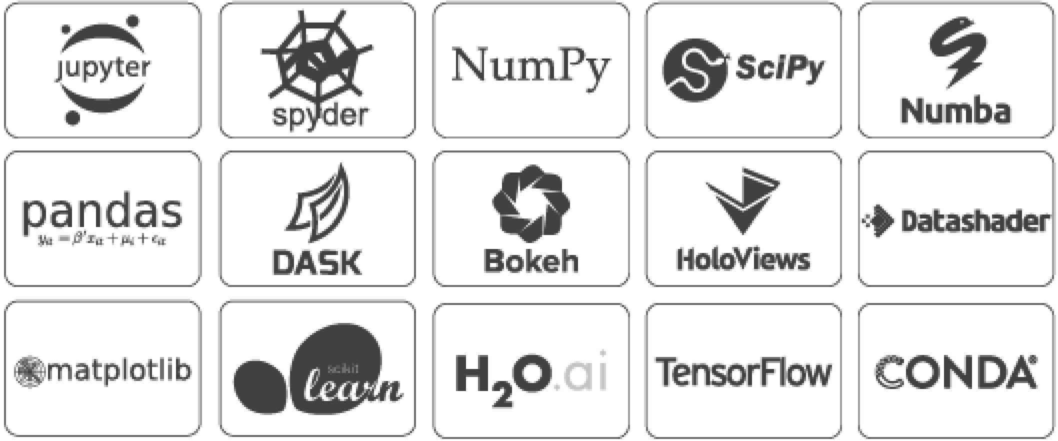
JupyterLab 1.0 (advanced interface)

Windows GUI Anaconda

https://www.anaconda.com/distribution/

to run on PC (offline)

Anaconda navigator: select packages to launch (off line) and tutorials



Examples of main available packages

Jupyter (web based NOT temporary)

Spyder GUI advanced programming and debugging

Numpy N-dimensional arrays

Scipy numerical routines

Matplotlib 2D and 3D outputs

Orange Machine learning

VS code Visual Studio

Jupyter modules

https://jupyter.org/try

- notebook and lab

Modules

- R
- C++
- …

(web based) notebook viewer by url https://nbviewer.jupyter.org/

extension .ipynb vs .py - ipy: interactive python

Jupyter notebook web interface

https://hub.gke.mybinder.org/user/ipython-ipython-in-depth-XXXXXXXX/notebooks/binder/FILENAME.ipynb

rename, save, download - after timeout kernel will not restart

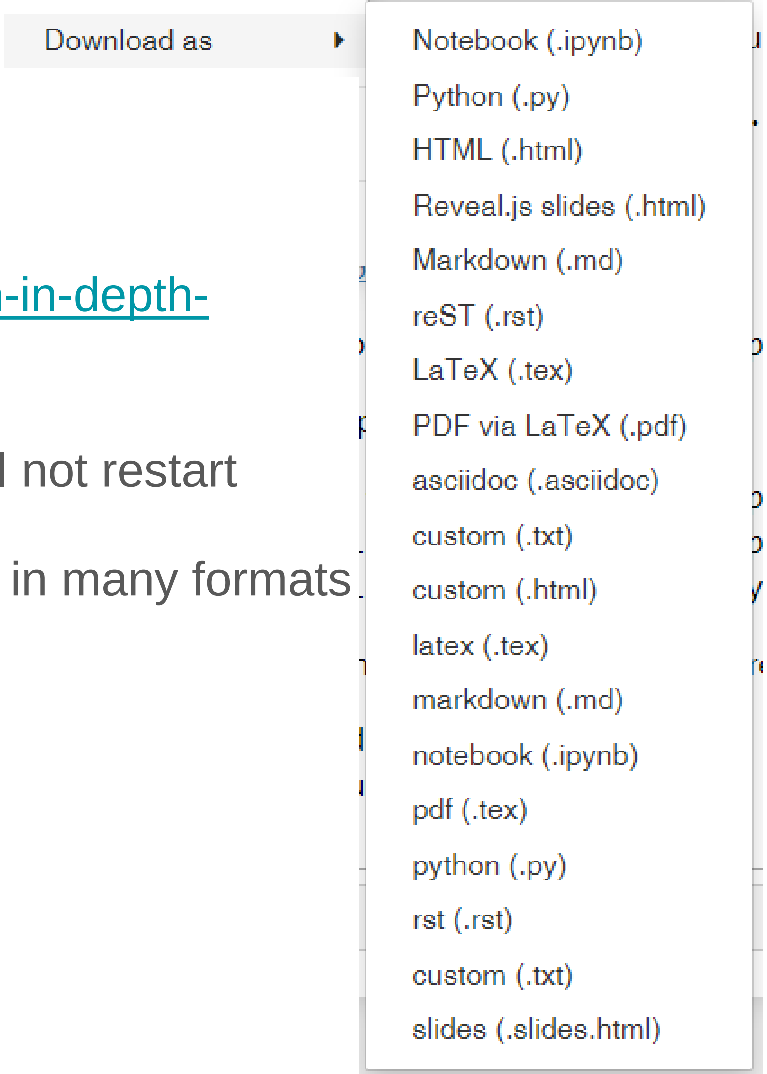
only python 3 available - files can be downloaded in many formats

code: python syntax

markdown: wiki like

raw: plain text ignored

heading: heading level



Interpreter features

Cell menu:

- run whole script
- individual cells (shift Enter)
- output inline

In editing (code) bar goes blue to green

File save and revert to checkpoint

Jupyter file binder: extra charge or timeout: https://gke.mybinder.org/

Jupiter editor supports 50 languages: File Open

Markdown (wiki style) 1

Type Or … to Get

```
*Italic*
_Italic_
```

Italic

```
**Bold**
__Bold__
```

Bold

```
# Heading 1
Heading 1
=========
```

Heading 1

```
## Heading 2
Heading 2
---------
```

Heading 2

```
[Link](http://a.com)

[Link][1]
⋮
[1]: http://b.org
```

[Link](http://b.org)

```
![Image](http://url/a.png)
![Image][1]
⋮
[1]: http://url/b.jpg
```

IMAGE

```
> Blockquote
```

> Blockquote

```
* List

* List

* List

- List

- List

- List
```

- List
- List
- List

Markdown (wiki style) 2

```
1. One

2. Two

3. Three

1) One

2) Two

3) Three
```

1. One
2. Two
3. Three

```
Horizontal Rule

---
Horizontal Rule

***
```

Horizontal Rule

```
`Inline code` with backticks
```

`Inline code` with backticks

insert/mix code to execute

~~~
``` (alt num lock 96) ASCII
# code block
print '3 backticks or'
print 'indent 4 spaces'
```
····# code block
····print '3 backticks or'
····print 'indent 4 spaces'
~~~

```
# code block
print '3 backticks or'
print 'indent 4 spaces'
```

Markdown interactive tutorial

from https://jupyter.org/ run JupyterLab

from Help select Markdown reference

and try the Tutorial

Base syntax

#this is a comment

? #help: alone, with variables, with functions - without ()

Syntax with numbers, strings, matrices

http://cs231n.github.io/python-numpy-tutorial/

Loops (for, in range etc.)

https://www.w3schools.com/python/python_for_loops.asp

If

https://www.w3schools.com/python/python_conditions.asp

MatPlotLib

Advanced graphics and display library
https://matplotlib.org/tutorials/introductory/images.html?highlight=import

connect to a GUI event loop: %matplotlib inline

load libraries

- import matplotlib.pyplot as plt
- import matplotlib.image as mpimg

upload (any) image (short name), e.g. in markdown from Edit: Insert Image

+ img = mpimg.imread('filename')

print(img) - displays the 3 vector components with ...

Image processing

- imgplot = plt.imshow(img)
- lum_img = img[:, :, 0]

plt.imshow(lum_img)

- plt.imshow(lum_img, cmap='hot')

plt.colorbar()

- plt.imshow(lum_img, cmap='gray')

plt.colorbar()

- `plt.hist(lum_img.ravel(), bins=256, range=(0.0, 1.0), fc='k', ec='k')`

Image processing

```
imgplot=plt.imshow(img[0:50,0:50,0:50]) #roi

import numpy as np

a=np.ones([100,100])

a[20:40,20:40]=np.ones([20,20])*500

imgplot = plt.imshow(a)

plt.colorbar() #viene tarata la luminosità, non ho
problemi di dato, posso andare oltre 255

plt.imshow(500-a) # negativo

imgplot = plt.imshow(img, interpolation="bicubic")
#sfocatura

  plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7],
  orientation='horizontal')
  print(type(img)) # <class 'numpy.ndarray'>
```

RGB

imageR=image[:,:,0]

imageG=image[:,:,1]

imageB=image[:,:,2]

plt.imshow(imageB, cmap='rainbow')

plt.colorbar()

plt.show() # opens figures and plots

Figure

```
fig = plt.figure()

a = fig.add_subplot(1, 2, 1)

imgplot = plt.imshow(lum_img)

a.set_title('Before')

plt.colorbar(ticks=[0.1, 0.3, 0.5, 0.7], 
orientation='horizontal')

a = fig.add_subplot(1, 2, 2)

imgplot = plt.imshow(lum_img)

imgplot.set_clim(0.0, 0.7)

plt.legend(title='Parameters where:')

finestra=plt.plot(a[:,25]) #profile image
```

```
plt.axis('off')

Colormap

plt.imshow(imageB, cmap='gray') #hot rainbow …

[] per le variabili

() per chiamare le funzioni
```

For

```
i=0

rows=len(image)

cols=len(image[0])

print(rows)

print(cols)

for row in range(rows):

    for col in range(cols):

        i=i+1

print(i)
```

```
i=0

rows=len(image)

cols=len(image[0])

print('rows =', rows)

print('cols =', cols)

dark=[]     #print(dark)
```

```
for row in range(rows-1):     #print(row)

    for col in range(cols-1):          #print(col)

        #print(image[row,col,0])

        if image[row,col,0]!=1:

            dark.append(col)

        else:

            col=cols

plt.plot(dark)
```

Figure subplot

```
fig = plt.figure()
plt.title('titolo del grafico')
for i in range(3):
    plt.subplot(1,3,i+1)
    a = fig.add_subplot(1, 3, i+1)
    plt.imshow(image[:,:,i])
```

```
fig = plt.figure()
plt.title('RGB')
for i in range(3):
    #plt.subplot(3,1,i+1)
    fig.add_subplot(1, 3, i+1)
    plt.imshow(coin[:,:,i], cmap='gray')
```

scipy.ndimage.filters https://docs.scipy.org/doc/scipy-0.14.0/reference/ndimage.html

```
import scipy.ndimage #as filt

plt.imshow(scipy.ndimage.rotate(img[:,:,0],45))

print(filt.maximum(img[:,0,0]))

print(filt.mean(img[:,:,:]))

plt.plot(filt.histogram(img[:,:,0], 0, 255, 256))

plt.plot(filt.histogram(img[:,:,1], 0, 255, 256))

plt.plot(filt.histogram(img[:,:,2], 0, 255, 256))
```

convolutions and morphological operations

```
plt.imshow(filt.laplace(img[:,:,0]), cmap='gray')

plt.imshow(filt.grey_erosion(img[:,:,0],5), cmap='gray')
```

binarization

```
rishor=len(image[:,0,0])

print(rishor)

risver=len(image[0,:,0])

print(risver)

thresh=254

imbin=np.ones((850,850))
```

```
for riga in range(risver):

    for col in range(rishor):

        if image[riga,col,0]<=thresh:

            imbin[riga,col]=0


plt.imshow(imbin, cmap='gray')

plt.colorbar()
```

binarization 2

```
threshold, upper, lower = 250, 1, 0 #create binary image

coinbin=((coin[:,:,2]>threshold)*upper) # use only blue (higher contrast) component (2)

plt.imshow(coinbin, cmap='gray')
```

morphological operations

```
kernel = np.array([[0, 1, 0],
                   [1, -4, 1],
                   [0, 1, 0]])
plt.imshow(scipy.ndimage.filters.convolve(imerode[:,:],kernel), cmap='gray')
# provare con imerode o imbin o image
```

```
imerode=scipy.ndimage.morphology.binary_opening(imbin, iterations=15)
plt.imshow(imerode, cmap='gray')
#plt.colorbar()

# provare anche erosion dilation opening closing
```

declare array

import numpy as np

zeros oppure ones

Compare lines

```
plt.plot(image[100,:,0])

plt.plot(image[200,:,0])
```

Convolutions

esempi di maschere

https://it.wikipedia.org/wiki/Matrice_di_convoluzione

vedere filtraggio (passa basso) media mobile excel

```
plt.imshow(filt.laplace(coin[:,:,2]), cmap='gray')
#edges
```

```
plt.imshow((filt.laplace(coin[:,:,2])>100)*1,
cmap='gray') #binarized edges
```

```
binedge=filt.laplace((coin[:,:,2])>200)*1
#binarized edges
```

```
plt.imshow(binedge, cmap='gray')
```

```
print(filt.center_of_mass(binedge))
```