S03_T02 Matrix_Structure

August 21, 2021

1 Matrix_Structure


S03_T02

1.1 ## Exercici_1

- Crea un np.array d'una dimensió, que inclogui almenys 8 nombres sencers, data type int64.
- Mostra la dimensió i la forma de la matriu.

[54]:
```
import numpy as np
import pprint

matriu = np.array([1, 2, 3, 4, 5, 6, 7 ,8])
print('Matriu:', pprint.pformat(matriu))
print('Dimensió de la matriu: {} \nForma de la matriu:{} 1x8'.format(matriu.
 ↪ndim, matriu.shape))
```

```
Matriu: array([1, 2, 3, 4, 5, 6, 7, 8])
Dimensió de la matriu: 1
Forma de la matriu:(8,) 1x8
```

1.2 ## Exercici_2

- Valor mitjà de la "matriu"

[55]:
```
matriu_mean = matriu.mean()

resta_matriu = matriu_mean - matriu

print('Valor mitjà de "matriu" = {} \nMean - "Matriu" = {}'
      .format(matriu_mean, resta_matriu))
```

```
Valor mitjà de "matriu" = 4.5
Mean - "Matriu" = [ 3.5  2.5  1.5  0.5 -0.5 -1.5 -2.5 -3.5]
```

1.3 ## Exercici_3

- Matriu bidimensional amb una forma 5x5
- Valor màx. de la matriu i dels seus eixos

[56]:
```
import numpy as np

'''
arr = np.array(range(25), ndmin=2)
arr5x5 = arr.reshape(5, 5)
'''

# randint(min, max,(row, column)) or randint(max, size=(row, column))
matriu5x5 = np.random.randint(26, size=(5,5))

print("Forma de la matriu 'matriu5x5':")
print(matriu5x5)
```

```
Forma de la matriu 'matriu5x5':
[[17  7 14 16 23]
 [10  8 24 15 10]
 [18  1 16  4 20]
 [21  6 16  5 18]
 [ 1 22  5 21 22]]
```

[57]:
```
# Valor màxim de la matriu 'matriu5x5'
print("\nValor màxim de la 'matriu5x5'es:", matriu5x5.max())

# Valor màxim dels eixos de la 'matriu5x5'
max_column = np.amax(matriu5x5, axis=0)
max_rows = np.amax(matriu5x5, axis=1)
print("Valors màx. per columna: {}\nValors màx. per fila: {}".
 ↪format(max_column, max_rows))
```

```
Valor màxim de la 'matriu5x5'es: 24
Valors màx. per columna: [21 22 24 21 23]
Valors màx. per fila: [23 24 20 21 22]
```

1.4 ## Exercici_4

- Broadcasting

[58]:
```
import numpy as np
import pprint as pp
from numpy.random import default_rng #NumpyUserGuide_randomGenerator
rng = default_rng()

# MATRIUS
w = rng.integers(6, size=(4, 5), endpoint=True)
y = rng.integers(7, size=(1, 5), endpoint=True)
z = rng.integers(5, size=(1,1), endpoint=True)
```

```
print('\nmatriu de {}: w ='.format(w.shape))
print(w)

print('\nmatriu de {}: y ='.format(y.shape), y)

print('\nmatriu de {}: z ='.format(z.shape), z)
```

```
matriu de (4, 5): w =
[[4 6 1 4 3]
 [3 6 3 4 3]
 [0 1 5 3 6]
 [0 3 5 0 6]]

matriu de (1, 5): y = [[1 6 1 6 2]]

matriu de (1, 1): z = [[3]]
```

[59]:
```
#SUMA/RESTA DE MATRIUS

ww = w + w # mateixa dimensió (4, 5)
print('\nmatriu de {}: ww ='.format(ww.shape))
print(ww)

wy = w + y # com minim una matriu amb dimensió_1 (4, 5)_(1, 5)
print('\nmatriu de {}: wy ='.format(wy.shape))
print(wy)
```

```
matriu de (4, 5): ww =
[[ 8 12  2  8  6]
 [ 6 12  6  8  6]
 [ 0  2 10  6 12]
 [ 0  6 10  0 12]]

matriu de (4, 5): wy =
[[ 5 12  2 10  5]
 [ 4 12  4 10  5]
 [ 1  7  6  9  8]
 [ 1  9  6  6  8]]
```

[60]:
```
# PRODUCTE DE MATRIUS

yxy = y*y # mateixa dimensió (1, 5)
print('\nmatriu de {}: yxy ='.format(yxy.shape), yxy)
## Exercici_4
```

```
wxy = w*y # nº columnas ==. (4, 5)_(1, 5)
print('\nmatriu de {}: wxy ='.format(wxy.shape))
print(wxy)
```

```
matriu de (1, 5): yxy = [[ 1 36  1 36  4]]

matriu de (4, 5): wxy =
[[ 4 36  1 24  6]
 [ 3 36  3 24  6]
 [ 0  6  5 18 12]
 [ 0 18  5  0 12]]
```

1.5 ## Exercici_5

- Indexació d'arrays: Extreure valors de les columnes i les files

[61]:
```
import numpy as np

# randint(min, max,(row, column)) o randint(max, size=(row, column))
M =np.array([[21, 13, -2],
             [ 4, -11, 9],
             [-23, 1, 24]])

print("Array_(3,3):")
print(M)

row1 = M[:1,:] #FILA_1 -- M[row:row, column:column]
print("\nrow_1 =", row1)

column3 = M[:3,2] #COLUM_3 -- M[row:row, column:column]
print("\ncolumn_3 =", column3)

rowcol = row1 + column3
print("\nSuma de row1 i column3 = {}".format(rowcol))
```

```
Array_(3,3):
[[ 21  13  -2]
 [  4 -11   9]
 [-23   1  24]]

row_1 = [[21 13 -2]]

column_3 = [-2  9 24]

Suma de row1 i column3 = [[19 22 22]]
```

1.6 ## Exercici_6

- Concepte " Mask " & cálculs booleans amb vectors

[62]:
```
import numpy as np

ar = np.array([[48, 13, -2],
               [ 4, -12, 9],
               [-23, 1, 24]])
print("ar =")
print(ar)

maskar = ar%4 == 0 # Masked_array | Numbers divisible by 4
print("\nMaskar =")
print(maskar)
```

```
ar =
[[ 48  13  -2]
 [  4 -12   9]
 [-23   1  24]]

Maskar =
[[ True False False]
 [ True  True False]
 [False False  True]]
```

1.7 ## Exercici_7

- Indexar la matriu màscara a la matriu original

[63]:
```
print("\nValues that satisfied the mask:\n", ar[maskar]) #values that satisfied␣
↪the mask
```

```
Values that satisfied the mask:
 [ 48   4 -12  24]
```

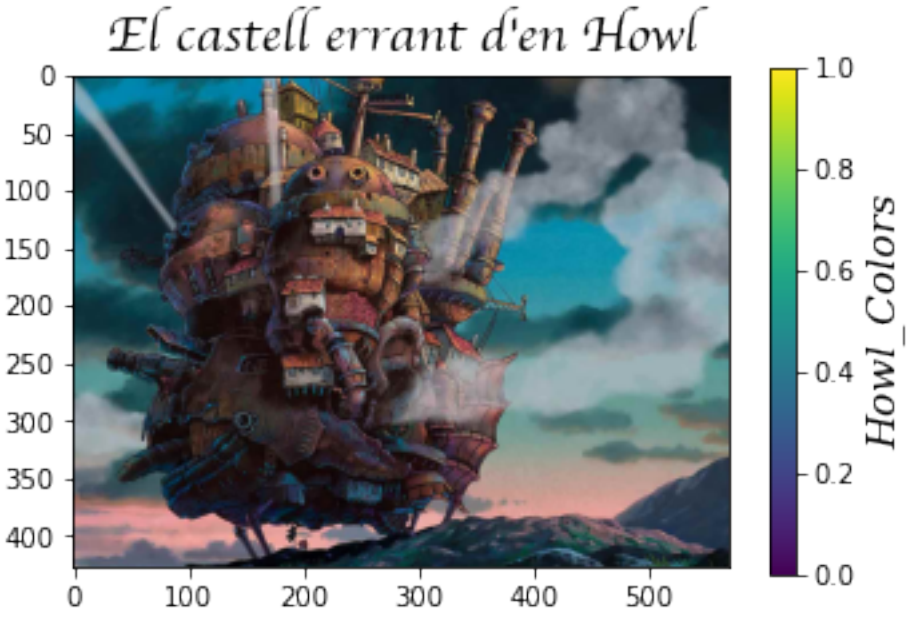
1.8 ## Exercici_8

- Manipulació d'imatges amb **Matplotlib**

[64]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

howl = mpimg.imread('images/castillo.png') # upload_the_img
print('Shape of "cast":\nAmplada = {}, Alçada = {}, RGB = {}'
      .format(howl.shape[0], howl.shape[1], howl.shape[2])) # img_shape

plt.rcParams["figure.figsize"] = 5.5, 3.5 # figure size
```

```
plt.imshow(howl) # img_show
plt.title(label = "El castell errant d'en Howl",
          fontdict = {#"weight":"bold",
                      "fontsize":"20",
                      "fontstyle":"italic",
                      "family":"cursive"}) # title_properties

colorbar=plt.colorbar(orientation="vertical").set_label(label="Howl_Colors",
                                                        size=15,
                                                        #weight="bold",
                                                        family="serif",
                                                       fontstyle="italic")
```

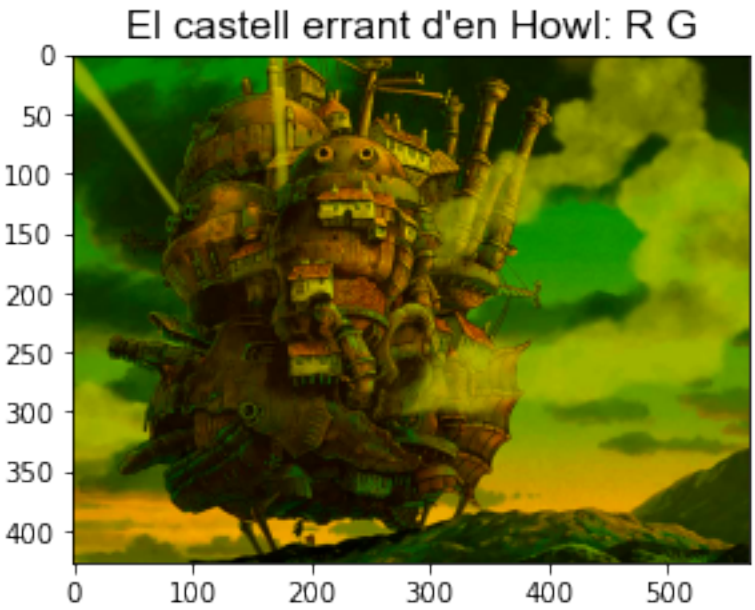
```
Shape of "cast":
Amplada = 427, Alçada = 570, RGB = 3
```



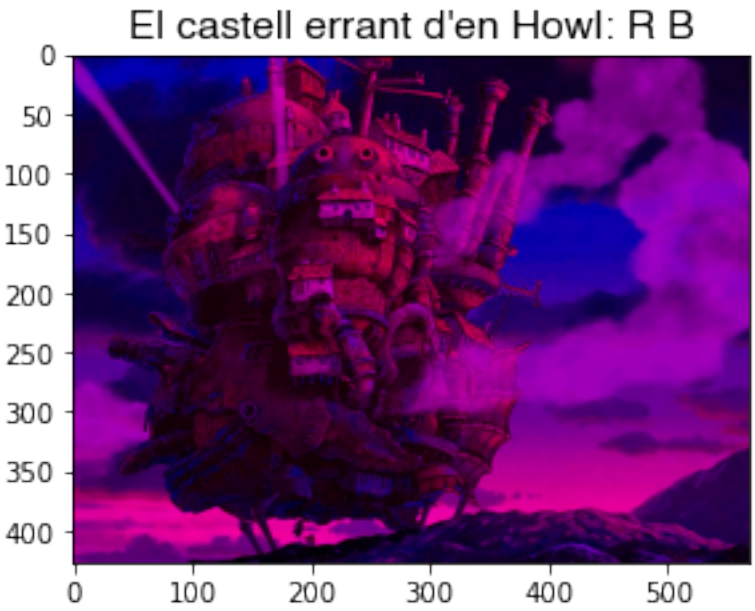
1.8.1 Single plots

[65]:
```
# Before, execute the first code cell of exercise 8

# Delete channel B "blue":

castcopy = np.copy(howl) # copy_img
castcopy[:,:,2]=0 # Channel_B_removed

plt.rcParams["figure.figsize"] = 5, 3.5 # figure size
```

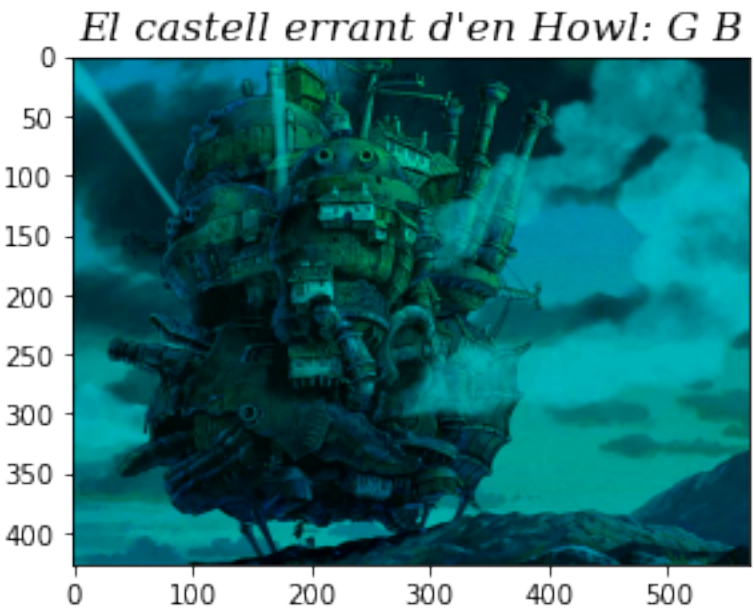
```
plt.imshow(castcopy) # show_img
plt.title(label = "El castell errant d'en Howl: R G",
          fontdict = {#"weight":"bold",
                      "fontsize":"15.5",
                      "family":"arial"}) # title_properties

plt.imsave("castcopy.png", castcopy) # save_img
```



[66]:
```
# Before, execute the first code cell of exercise 8

# Delete channel G "green":

castcopy1 = np.copy(howl) # copy_img
castcopy1[:,:,1]=0 # Channel_G_removed
#castcopy1[213:,285:,1]=0 # Channel_G_removed_Range

plt.rcParams["figure.figsize"] = 5, 3.5 # figure size
plt.imshow(castcopy1) # show_img
plt.title(label = "El castell errant d'en Howl: R B",
          fontdict = {"weight":"bold",
                      "fontsize":"15.5",
                      "family":"helvetica"}) # title_properties

plt.imsave("castcopy1.png", castcopy1) # save_img
```

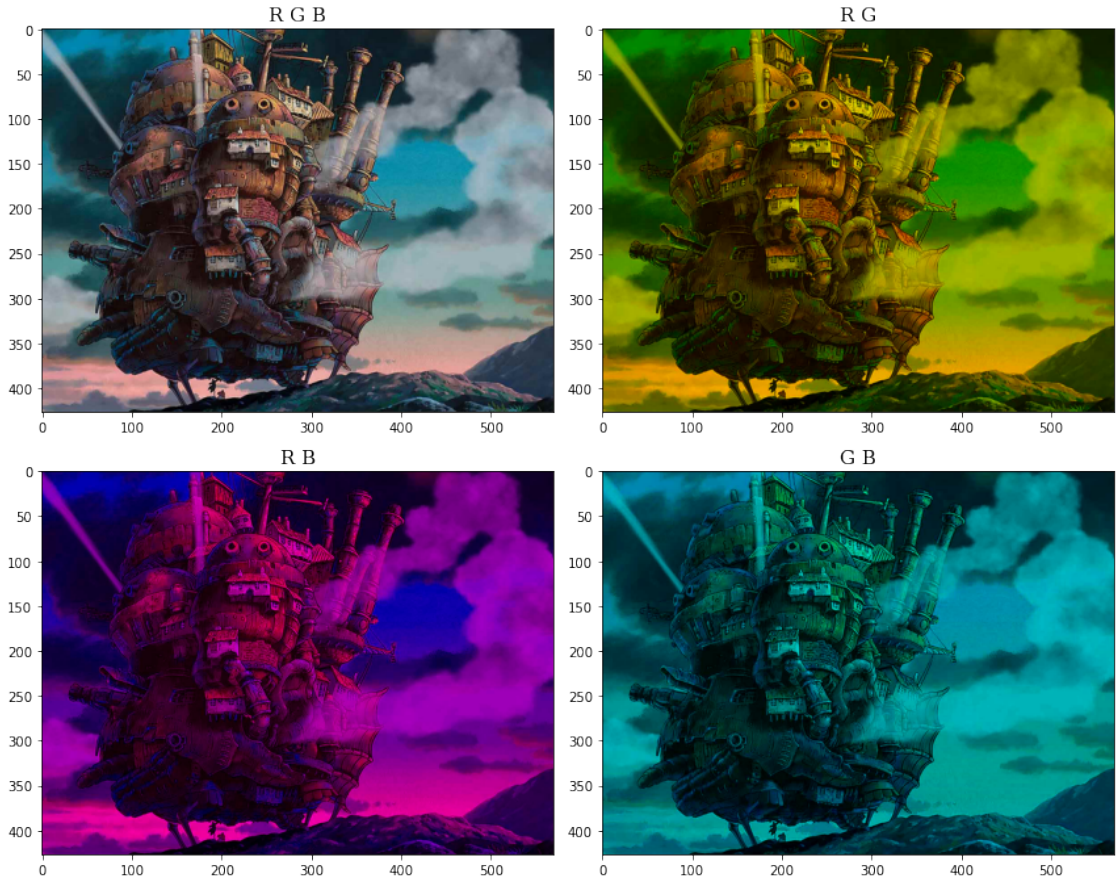


[67]:
```
# Before, execute the first code cell of exercise 8

# Delete channel R "red":

castcopy2 = np.copy(howl) # copy_img
castcopy2[:,:,0]=0 # Channel_R_removed


plt.rcParams["figure.figsize"] = 5, 3.5 # figure size
plt.imshow(castcopy2) # show_img
plt.title(label = "El castell errant d'en Howl: G B",
          fontdict = {"fontsize":"15.5",
                      "family":"serif",
                      "fontstyle":"italic"
                     }) # title_properties

plt.imsave("castcopy2.png", castcopy2) # save_img
```



1.8.2 Multiple subplots

[68]:
```
fig = plt.figure(figsize=(12, 9.5)) # Create new figure...

# Plot nº1: RGB
rgb = fig.add_subplot(2, 2, 1)
howlplot = plt.imshow(howl) # Show_img
rgb.set_title('R G B', family='serif', fontsize=15) # title_properties

# Plot nº2: RG
rg = fig.add_subplot(2, 2, 2)
howlplot = plt.imshow(castcopy)
rg.set_title('R G', family='serif', fontsize=15)

# Plot nº3: RB
rb = fig.add_subplot(2, 2, 3)
howlplot = plt.imshow(castcopy1)
rb.set_title('R B', family='serif', fontsize=15)

# Plot nº4: GB
gb = fig.add_subplot(2, 2, 4)
howlplot = plt.imshow(castcopy2)
gb.set_title(label='G B', family='serif', fontsize=15)


fig.tight_layout() # Mantiene automaticamente el espacio entre los plot
```


R G B
R G
R B
G B