Tutorial: plotting with Matplotlib

Jakub Tworzydło

Institute of Theoretical Physics
phone: (022)5532-919, room 5.19
Jakub.Tworzydlo@fuw.edu.pl

28/02/2023 and 1/03/2023 ul. Pasteura, Warszawa

NumPy arrays

It is more Python-like not to loop over the elements!

```
# Filename: arrays.py
# 3 ways to initialize numpy arrays
import numpy as np # short name of numpy module

n = 7
v = np.zeros(n)     # vector as a numpy array

for i in range(n):
   v[i] = i/2.0     # sets elements v_i
print( 'v = ', v )

u = np.arange(0.,1.,0.2)          # range of values
print( 'u = ', u )

w = np.linspace(-np.pi,np.pi,6) # also some range, but for
print( 'w = ', w )                # a given number of points
```

Simplest plot

Plotting is easy, check it with the example below!

```
# Filename: plots.py

import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(-np.pi, np.pi, 500)
# Numpy math functions operate on whole arrays (!)
y_cos, y_sin = np.cos(x), np.sin(x)

plt.plot(x, y_cos) # plot cosine function
plt.plot(x, y_sin) # plot sine function

plt.show()
```

Enhanced plot

We improve a bit by adding figure size, scale, line styles etc.

```
# Filename: nice_plots.py
import matplotlib.pyplot as plt
import numpy as np
x = np.linspace(-np.pi, np.pi, 500)
y_cos, y_sin = np.cos(x), np.sin(x)

# we can set the figure size (in inches)
plt.figure(figsize=(5, 3), dpi=150)
# calculate and set data range
plt.xlim(x.min() * 1.0, x.max() * 1.0)
plt.ylim(y_cos.min() * 1.1, y_cos.max() * 1.1)

plt.plot(x, y_cos, color="b", linewidth=2., linestyle="-")
plt.plot(x, y_sin, color="r", linewidth=2., linestyle="--")

plt.grid()
plt.savefig('nice_plots.png')
```

More controll over the plot

We can add and manipulate the descriptions! Add the following lines below `plt.figure` command:

```
# Figure and axis title
plt.title('Functions: cosine and sine', fontsize=16)
plt.xlabel('x value', fontsize=12)
plt.ylabel('y result', fontsize=12)
# Tell matplotlib to use LaTeX to render text
plt.rc('text', usetex=False)
# Set xticks values and description, modify yticks
plt.xticks( [-np.pi, -np.pi/2, 0, np.pi/2, np.pi],
            ['$-\pi$', '$-\pi/2$', '0', '$\pi/2$', '$\pi$']
plt.yticks( ticks=np.arange(-1,1.1,0.5) )
```

After `plt.plot` add the legend description

```
# Add legend
plt.legend( ("cos(x)","sin(x)"), loc='upper left')
```

and save with

```
plt.savefig('nice_plots.png',bbox_inches="tight")
```