11 – Plotting

Bálint Aradi

Scientific Programming in Python (2026)

https://atticlectures.net/scipro/python-2026/

Plotting with matplotlib

Matplotlib interfaces

- Matlab-like simplified interface with instance variables (recommended)
- Matlab-like simplified interface with global variables
- Fully object oriented interface (when embedding)

Matplotlib render engines

- Embedding plots into the IPython/Jupyter notebook

```
%matplotlib inline
```

In JupyterLab this is already the default

- Showing plots in separate windows (when using from script or from IPython-console)
- Creating graphical files (pdf, jpg, etc.)

Self-containing plotting example

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


xx = np.linspace(0.0, 4.0 * np.pi, 200, endpoint=True)
y1 = np.cos(xx)
y2 = np.sin(xx)


fig, ax = plt.subplots()
ax.plot(xx, y1, color='red', linewidth=1.0, linestyle="--", label='cos(x)')
ax.plot(xx, y2, color='blue',linewidth=1.0,linestyle="-", label='sin(x)')
ax.legend()
plt.show()
```

Generating x/y values

Create Figure and Axes objects (multiple subplots possible)

Plot curves through given x/y values

Create legend box

Render plot/figure (optional in JupyterLab)

- If you do not use **plt.show()** in Jupyter, append semicolon (“**;**”) to last line of the cell to suppress additional non-graphical output
- If you use a GUI-backend, you can also use **fig.show()** to render a figure

Self-containing plotting example

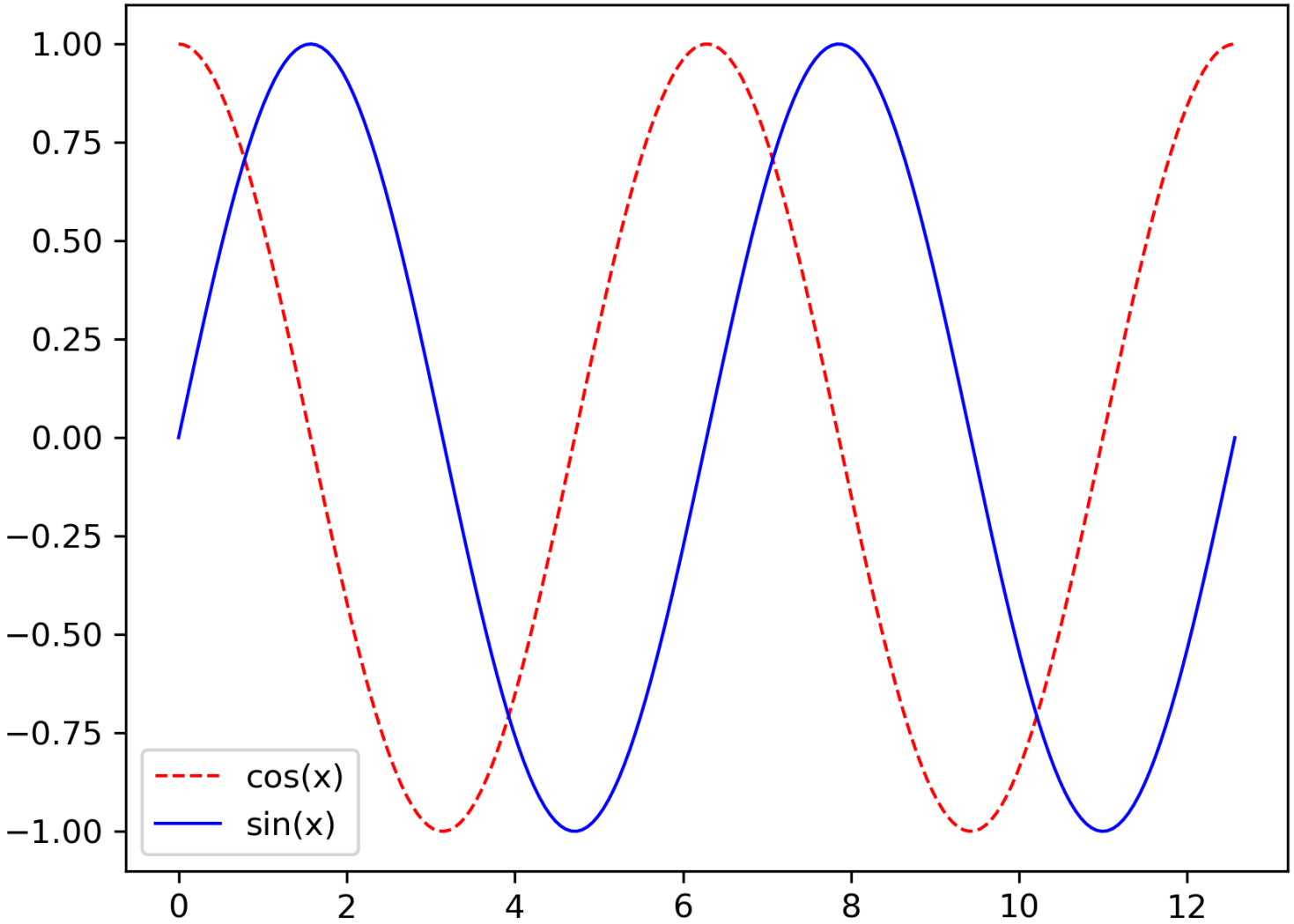


Figure and Axes objects

Figure

- A Figure object instance represents the figure
- Figure objects enables to manipulate the global figure parameters or to execute global actions

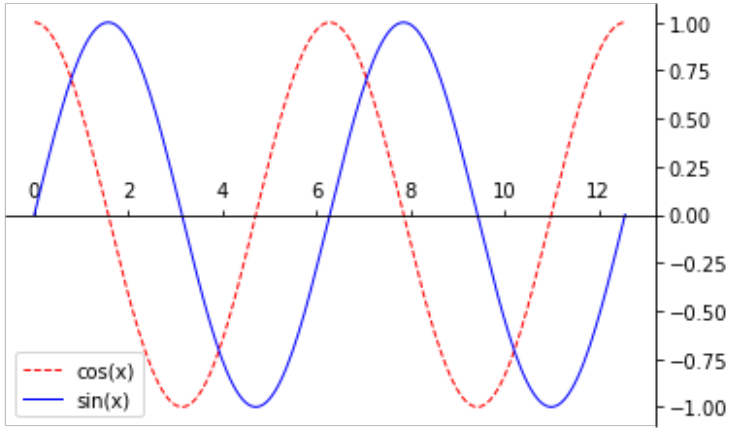
```
fig.set_size_inches(10, 8)
fig.set_dpi(300)
```

```
fig.savefig('plot.pdf')
```

Axes

- An Axes-object instance represents one plot within the figure
- Axes-object enables very detailed tuning of the resulting plot

```
ax.xaxis.set_ticks_position('top')
ax.yaxis.set_ticks_position('right')
ax.spines['top'].set_position(('data', 0))
ax.spines['bottom'].set_color('none')
ax.spines['left'].set_color('none')
```



Mulitple subplots

- The **subplots()** command can create multiple subfigures
- It returns individual Axes objects (one for each subfigure)

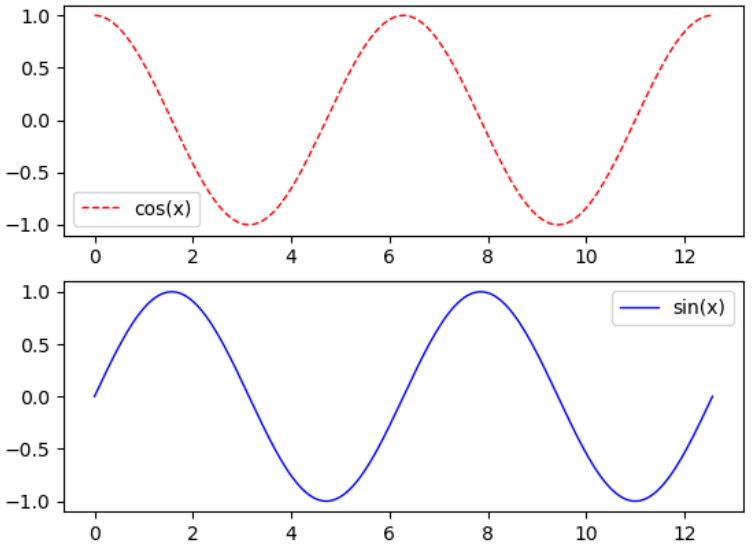
```
fig, (ax1, ax2) = plt.subplots(2, 1)

ax1.plot(xx, y1, color='red', linewidth=1.0, linestyle="--", label='cos(x)')
ax1.legend()

ax2.plot(
    xx, y2, color='blue', linewidth=1.0,
    linestyle="-", label='sin(x)'
)
ax2.legend()

plt.show()
```

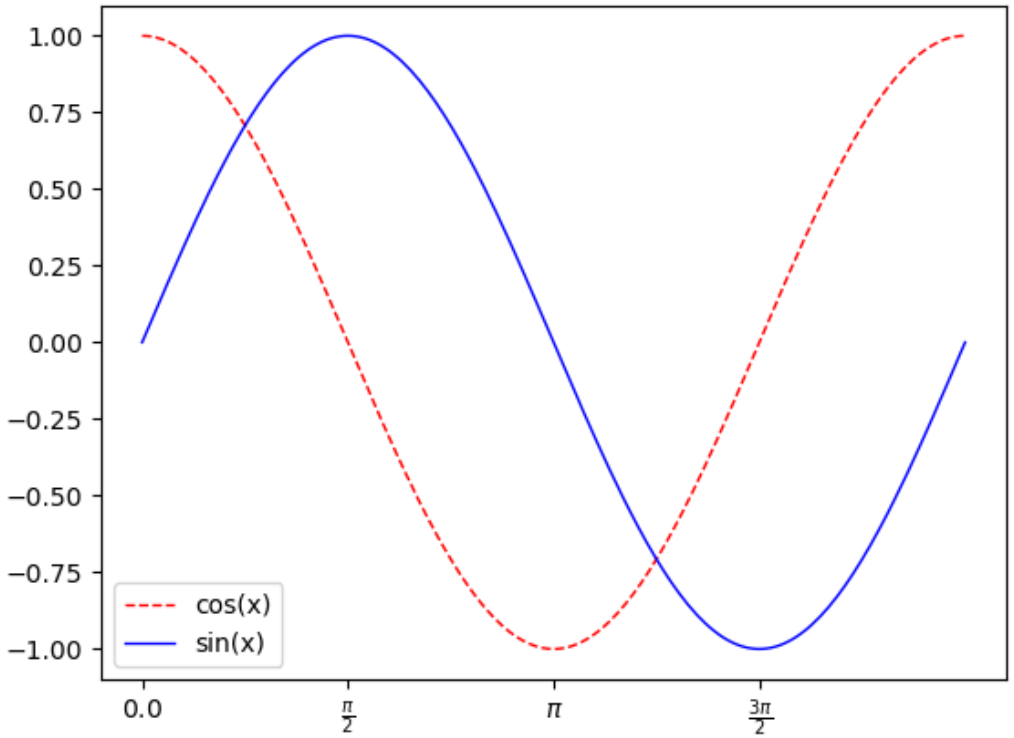
Two rows, one column (2 figures)



Rendering TeX within plots

```
ax.set_xticks(
    [0.0, np.pi / 2, np.pi, 3 * np.pi / 2],
    [r'$0.0$', r'$\frac{\pi}{2}$', r'$\pi$', r'$\frac{3\pi}{2}$']
)
```

- Matplotlib can render TeX sequences in plots
- TeX-sequences should be delimited by $
- It is advisable to put TeX-sequences into **raw-strings** (**r'something'**)
- In raw-strings, backslashes are interpreted literally and not as special Python commands (e.g. \n as "\" "n" and not as newline)
- Useful when passing backslash commands to various enginens (TeX-sequences in Matplotlib, regular expressions, ...)



Further useful Axes methods

ax.set_xlim(), **ax.set_ylim()**	Setting/Querying x/y limits
ax.set_xticks(), **ax.set_yticks()**	Setting customized ticks (and tick labels)
ax.annotate()	Write text into the plot
ax.plot()	Curve plot
ax.scatter()	Scatter plot
ax.bar()	Bar plot
ax.contour()	Contour plot
ax.imshow()	Bitmap image
ax.pie()	Pie charts
ax.quiver()	Quiver plots
:	

- Various excellent tutorials on Matplotlib available
- See for example Matplotlib Quick Start Guide or Matplotlib: Plotting (in Scipy Lecture Notes)
- Some tutorials (e.g. Scipy-lectures) use the global interface access (easy to convert)