University: Mohamed Khider University of Biskra
Faculty: Faculty of Exact Sciences
Department: Department of SM
Academic Year: 2026/2027
Prepared by: Dr OUAAR Fatima

Module: Scientific Programming
Level: 2nd Year LMD
Specialty: Physics/MCPC
Session: Practical Work N°6
Subject: Matplotlib – Basic Scientific Plots

SESSION 6
Matplotlib – Basic Scientific Plots

LEARNING GOALS

- Understand the basic structure of a scientific graph.
- Plot data using `plot()`.
- Add axis labels, a title and a grid.
- Display two curves on the same figure.
- Customize colors, line styles and markers.
- Plot discrete measurements with `scatter()` and `errorbar()`.
- Combine several graphs with `subplots()`.
- Use logarithmic scales and save figures for reports.

SHORT THEORY

Matplotlib is a Python library for visualization. A basic scientific graph should clearly show the data, axes, title and, when needed, a legend and grid.

1. First plot

LET'S TRY

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

x = np.linspace(0, 10, 100)
y = x**2

plt.plot(x, y)
plt.xlabel("x")
plt.ylabel("y")
plt.title("y = x^2")
plt.grid()
plt.show()

```

2. Two curves

GUIDED PRACTICE

```
y1 = np.sin(x)
y2 = np.cos(x)

plt.plot(x, y1, label="sin(x)")
plt.plot(x, y2, label="cos(x)")
plt.xlabel("x")
plt.ylabel("Value")
plt.title("Sine and Cosine")
plt.legend()

```

3. Markers, colors and line styles

SHORT THEORY

The look of a curve is controlled with optional arguments: `color`, `linestyle` (`'-'`, `'--'`, `':'`), `linewidth`, `marker` (`'o'`, `'s'`, `'^'`, `'*'`), `markersize` and `markevery`.

LET'S TRY

```
plt.plot(x, y1, color="red", linestyle="--", linewidth=2, label="sin(x)")
plt.plot(x, y2, color="blue", linestyle=":", marker="o",
markevery=10, label="cos(x)")
plt.xlabel("x")
plt.ylabel("Value")
plt.title("Customized curves")
plt.legend()
plt.grid(alpha=0.3)
plt.show()

```

4. Scatter plot (discrete measurements)

SHORT THEORY

Measured data points are drawn with `plt.scatter(x, y)`. Use it together with a theoretical curve to compare experiment and model.

LET'S TRY

```
t = np.array([0, 1, 2, 3, 4, 5])
v = np.array([0.0, 4.9, 9.8, 14.7, 19.6, 24.5])  # measured speed (m/s)

plt.scatter(t, v, color="green", s=80, marker="s", label="experiment")
plt.plot(t, 9.8*t, color="gray", linestyle="--", label="v = g t")
plt.xlabel("t (s)")
plt.ylabel("v (m/s)")
plt.title("Free fall")
plt.legend()
plt.grid(alpha=0.3)
plt.show()

```

5. Several graphs in one figure (subplots)

LET'S TRY

```
fig, axes = plt.subplots(1, 2, figsize=(10, 4))

axes[0].plot(x, y1)
axes[0].set_title("Sine")
axes[0].grid()

axes[1].plot(x, y2, color="red")
axes[1].set_title("Cosine")
axes[1].grid()

plt.tight_layout()
plt.show()

```

6. Physics connection: damped oscillations

PHYSICS CONNECTION

A damped harmonic oscillator follows

$$x(t) = A\,e^{-\gamma t}\cos(\omega t + \varphi).$$

Below we plot the position together with the envelope $\pm Ae^{-\gamma t}$.

```
t = np.linspace(0, 10, 500)
A, gamma, omega = 1.0, 0.3, 2*np.pi
x = A * np.exp(-gamma*t) * np.cos(omega*t)
envelope = A * np.exp(-gamma*t)

plt.plot(t, x, label="x(t)")
plt.plot(t, envelope, "--", color="gray", label="+/- envelope")
plt.plot(t, -envelope, "--", color="gray")
plt.xlabel("t (s)")
plt.ylabel("x (m)")
plt.title("Damped harmonic oscillator")
plt.legend()
plt.grid(alpha=0.3)
plt.show()

```

7. Error bars

SHORT THEORY

Every experimental value carries an uncertainty. `plt.errorbar(x, y, yerr=...)` draws the points with vertical error bars (`xerr` for horizontal ones, `fmt` for the marker style).

LET'S TRY

```
L = np.array([0.25, 1.00, 2.25, 4.00])    # pendulum length (m)
T = np.array([1.02, 2.03, 3.05, 4.10])    # measured period (s)
dT = np.array([0.04, 0.05, 0.06, 0.08])   # uncertainty on T

plt.errorbar(L, T**2, yerr=2*T*dT, fmt="o", color="purple",
ecolor="gray", capsize=5, label="T^2 vs L")
plt.plot(L, 4*np.pi**2*L/9.8, "--", color="black",
label="T^2 = 4 pi^2 L / g")
plt.xlabel("L (m)")
plt.ylabel("T^2 (s^2)")
plt.title("Pendulum experiment")
plt.legend()
plt.grid(alpha=0.3)
plt.show()

```

8. Logarithmic scales

SHORT THEORY

`plt.loglog(x, y)` uses logarithmic axes on both sides, `plt.semilogx` / `plt.semilogy` on one side only. A power law becomes a straight line on a log–log plot, and an exponential on a semi-log plot.

LET'S TRY

```
f = np.logspace(0, 5, 200)                # frequency, 1 to 10^5 Hz
gain = 1.0 / np.sqrt(1 + (f/1000)**2)       # RC low-pass filter

plt.loglog(f, gain)
plt.xlabel("Frequency (Hz)")
plt.ylabel("Gain (V/V)")
plt.title("RC filter response")
plt.grid(True, which="both", alpha=0.3)
plt.show()

```

9. Saving the figure

QUICK TIP

`plt.savefig("figure.png", dpi=300)` must be called *before* `plt.show()`. Use `dpi=300` (or higher) for reports and posters, and `bbox_inches="tight"` to remove empty margins.

LET'S TRY

```
plt.plot(x, y1, label="sin(x)")
plt.legend()
plt.savefig("my_plot.png", dpi=300, bbox_inches="tight")
plt.show()

```

10. Exercises (no solution – try it yourself)

TRY IT YOURSELF – NO SOLUTION

Exercise 1. Plot $y = e^{-x}\sin(3x)$ on $[0, 10]$ with a red dashed line, axis labels and a grid.

TRY IT YOURSELF – NO SOLUTION

Exercise 2. On the same figure, plot $\sin(x)$ (solid line with circles every 10 points) and $\sin(2x)/2$ (dashed line with squares). Add a legend.

TRY IT YOURSELF – NO SOLUTION

Exercise 3. A ball is thrown with $v_0 = 20$ m/s at an angle of 45°. Plot the trajectory $y(x)$ for $g = 9.8$ m/s^2, with x and y in meters, and mark the landing point with a red star.

TRY IT YOURSELF – NO SOLUTION

Exercise 4. With `plt.subplots(2, 1)`, draw x^2 and x^3 one above the other, sharing the same x axis, each with its own title.

TRY IT YOURSELF – NO SOLUTION

Exercise 5. Radioactive decay: $I(t) = I_0\, e^{-t/\tau}$ with $\tau = 3.8$ days. Plot $I(t)/I_0$ with `plt.semilogy` for $0 \le t \le 20$ days and verify that the curve is a straight line.

ASSESSMENT – 4 POINTS

Assessment (4 points). Newton's cooling law: $T(t) = T_0 + (T_i - T_0)\, e^{-kt}$, with $T_0 = 25\,^\circ$C, $T_i = 90\,^\circ$C, $k = 0.05$ min^{-1}.

1. (1 pt) Plot $T(t)$ for $t \in [0, 60]$ min (labels, title, grid).
2. (1 pt) Add the measured points `t_exp = [0, 10, 20, 30, 40, 50, 60]`, `T_exp = [90, 82, 70, 58, 48, 40, 33]` as red circles.
3. (1 pt) Add a legend and save the figure as `cooling.png` at 300 dpi.
4. (1 pt) From the graph, estimate after how many minutes the liquid reaches 40 °C.