Clase 035 — Matplotlib: anatomía figura/axes

Parte: 0 — Prerrequisitos · Fuente: VanderPlas, cap. 4 § 4.1 Visualization with Matplotlib. · Duración estimada: 60 min.

Clase 035 — Matplotlib: anatomía figura/axes

> *Parte: 0 — Prerrequisitos · Fuente: VanderPlas, cap. 4 § 4.1 Visualization with Matplotlib. · Duración estimada: 60 min.*

Objetivo

Que el alumno entienda la jerarquía de objetos de matplotlib (Figure → Axes → Artist) y use la API orientada a objetos (fig, ax = plt.subplots()) en vez del interfaz pyplot estilo MATLAB. Esto es lo que separa gráficos publicables de notebooks de cualquier curso introductorio.

Resultados de aprendizaje

Al finalizar la clase, el alumno podrá:

1. Explicar la jerarquía Figure → Axes → Artist y por qué la API OO es preferible.
2. Crear una figura con fig, ax = plt.subplots(figsize=(8, 4)) y configurar título, ejes, leyenda.
3. Guardar una figura a PNG/SVG/PDF con DPI controlado.
4. Cerrar figuras explícitamente para liberar memoria en notebooks que generan muchas.
5. Configurar defaults con plt.rcParams (font, line width, colors).

Temas

#	Tema	Por qué importa
1	Figure (canvas) → Axes (gráfico) → Artist	Modelo mental.
2	pyplot vs OO API	plt.plot (state-based) vs ax.plot (explíci
3	fig, ax = plt.subplots()	El patrón canónico.
4	fig.savefig y formatos	PNG raster vs SVG/PDF vector.
5	Liberar memoria: plt.close(fig)	Importante en loops.
6	plt.rcParams y stylesheets	Defaults globales.

Definiciones y características

Figure

: El canvas/lienzo completo (ventana, archivo PNG). Una Figure contiene N Axes.

Axes

: Un gráfico individual con sus ejes X/Y, ticks, labels, leyenda. No es plural de axis — es la palabra técnica de matplotlib para 'el rectángulo donde se dibuja'.

Artist

: Todo lo dibujado: lines (Line2D), puntos (PathCollection), texto, leyenda. Cada element es un Artist con propiedades modificables.

API OO vs pyplot

: OO: fig, ax = plt.subplots(); ax.plot(...) — explícito, escalable. pyplot: plt.plot(...) — estilo MATLAB, estado

global, menos predecible. Usa OO siempre.

rcParams

: Dict global con defaults de matplotlib: plt.rcParams['figure.figsize'] = (10, 5). Modificar afecta todos los plots subsiguientes hasta reset.

Dataset / recursos

Sintético: serie temporal corta + scatter. Sin descarga.

Ejercicios

1. Hello world. Crea figura 8×4, plot de y = sin(x) para x [0, 2π]. Título, xlabel, ylabel.

2. Dos líneas en un axes. Misma figura: sin(x) y cos(x) con colores distintos y leyenda.

3. Guarda 3 formatos. Mismo plot a PNG (100 DPI), PNG (300 DPI), SVG. Compara tamaños.

4. Loop sin leak. Genera 20 plots en loop. Cierra cada uno con plt.close(fig). Verifica que len(plt.get_fignums()) queda en 0.

5. rcParams. Cambia font.size y lines.linewidth para tu sesión. Verifica el efecto.

Homework verificable

Notebook: (a) figura sin/cos con todos los elementos (título, labels, leyenda, grid); (b) guardar PNG@300dpi y SVG; (c) generar 50 plots en loop sin memory leak; (d) demo de rcParams modificados.

Criterio de aceptación: Plot publicable (labels, leyenda, tamaño razonable). Loop deja 0 figuras abiertas.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
Mi notebook se llena de memoria al hacer m	Cada plt.subplots() deja Figure abierta. F
savefig('out.png') da PNG con fondo transp	Default es transparent. Fix: fig.savefig('
Labels cortados al guardar	Bounding box no incluye text fuera del axe
plt.plot(x, y) no muestra nada en notebook	Falta %matplotlib inline (default suele es
Texto en español sale mal	Default font no tiene tildes/ñ. Fix: plt.r

Preguntas frecuentes

¿fig, ax = plt.subplots() o fig = plt.figure(); ax = fig.add_subplot()?

subplots() es el shortcut más usado. Da figure + axes en una línea, devuelve grid si pasas (n, m). El segundo es más explícito y útil para layouts custom (GridSpec).

¿PNG, SVG o PDF?

PNG para web/presentaciones (raster, DPI fijo). SVG para documentos editables / publicación (vector, escala infinita). PDF para LaTeX (también vector).

¿Cuándo constrained_layout=True vs tight_layout()?

constrained_layout=True (al crear figure): más nuevo, más confiable, maneja colorbars y leyendas externas mejor. tight_layout() (después): legacy, falla con elementos no estándar.

¿Por qué mi plot se ve diferente entre script y notebook?

Backend distinto: notebook usa inline, script usa Qt5Agg/Tk. DPI y tamaño cambian. Para consistencia: plt.rcParams['figure.dpi'] = 100 explícito.

¿Está bien usar plt.plot() directo?

Para 1 plot rápido en notebook, OK. Para cualquier cosa más compleja (subplots, savefig, reutilizable), siempre API OO.

Referencias

- VanderPlas, cap. 4 § 4.1.
- matplotlib quick start
- Anatomy of a figure (matplotlib gallery)

Siguiente clase

Clase 036 — Matplotlib: line, scatter, bar, histogram, boxplot

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
import numpy as np
import matplotlib.pyplot as plt
print('matplotlib:', plt.matplotlib.__version__)
```

Archivos complementarios

- notebook.ipynb