Clase 040 — Matplotlib: 3D plotting

Parte: 0 — Prerrequisitos · Fuente: VanderPlas, cap. 4 § 4.12 Three-Dimensional Plotting in Matplotlib. · Duración estimada: 45 min.

Clase 040 — Matplotlib: 3D plotting

> *Parte: 0 — Prerrequisitos · Fuente: VanderPlas, cap. 4 § 4.12 Three-Dimensional Plotting in Matplotlib. · Duración estimada: 45 min.*

Objetivo

Que el alumno sepa cuándo (raramente) usar 3D y cómo hacerlo bien: scatter 3D, superficies (plot_surface), wireframes y contornos. Spoiler: la mayoría de las veces un buen 2D + color comunica mejor.

Resultados de aprendizaje

Al finalizar la clase, el alumno podrá:

1. Crear axes 3D con projection='3d'.
2. Scatter, line, surface, wireframe, contour en 3D.
3. Controlar ángulo de vista con ax.view_init(elev, azim).
4. Reconocer cuándo NO usar 3D: la mayoría de las veces hay una alternativa 2D mejor.

Temas

#	Tema	Por qué importa
1	projection='3d'	Habilita el 3D toolkit.
2	Scatter 3D con codificación por color	3 dims + 4 (color).
3	plot_surface para z = f(x, y)	Funciones bivariadas.
4	plot_wireframe y contour3D	Alternativas más simples.
5	view_init: rotar interactivo	En notebooks con %matplotlib widget.
6	Cuándo NO usar 3D	Casi siempre.

Definiciones y características

projection='3d'

: Parámetro al crear axes (fig.add_subplot(111, projection='3d')) que habilita el toolkit 3D (mplot3d). Sin esto, las llamadas 3D fallan.

plot_surface(X, Y, Z)

: Superficie 3D para z = f(x, y). Requiere mesh: X, Y = np.meshgrid(x, y); Z = f(X, Y). Soporta cmap para color por valor de Z.

plot_wireframe(X, Y, Z)

: Como surface pero solo líneas — más liviano, mejor para densidad alta, peor para forma.

view_init(elev, azim)

: Define ángulo de cámara. elev elevación (0=horizontal, 90=vista superior). azim rotación horizontal en grados. Animar varia azim para 360°.

Oclusión

: Limitación fundamental del 3D: objetos al frente tapan los del fondo, sin forma confiable de elegir qué ver. La razón principal por la que 3D es problemático.

Dataset / recursos

Sintético: superficie analítica + nube de puntos. Sin descarga.

Ejercicios

1. Scatter 3D. 200 puntos con coords (x, y, z) y color por una 4ª variable.

2. Superficie. z = sin(sqrt(x² + y²)) en mesh 50×50. plot_surface con colormap.

3. Wireframe + contour. Misma función con plot_wireframe. Compara legibilidad con superficie llena.

4. view_init. Cambia (elev, azim) a 4 ángulos y graba una grilla 2×2.

5. Reto: 2D que vence al 3D. Para tu scatter 3D del ejercicio 1, propón un 2D + color/tamaño que comunique igual o mejor.

Homework verificable

Notebook: (a) scatter 3D con 4 dimensiones (xyz + color); (b) superficie z=f(x,y); (c) wireframe del mismo z; (d) grilla 2×2 con 4 view_init distintos; (e) ejercicio de "2D vence al 3D": versión 2D del scatter.

Criterio de aceptación: Plots 3D legibles (no espagueti). Versión 2D del scatter comparable.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
projection='3d' da error "unknown projecti	No importaste el toolkit. Fix: from mpl_to
Surface plot tarda mucho con mesh grande	100×100 mesh = 10k vertices. Fix: reduce r
3D scatter con miles de puntos = mancha	Oclusión. Fix: reduce N (sampling), o usa
view_init no se aplica si llamado después	El orden importa: configura ángulo ANTES d
Labels Z se cortan o solapan	3D tiene limitaciones de layout. Fix: ajus

Preguntas frecuentes

¿Realmente necesito 3D?

Pregúntate: ¿hay una versión 2D con color/tamaño que comunique igual? Casi siempre sí. 3D vale para superficies analíticas (z = f(x,y)), datos físicos 3D, o cuando es interactivo (rotable).

¿%matplotlib widget para rotar interactivo?

Sí — en JupyterLab/Notebook permite rotar con el mouse. Requiere pip install ipympl. Default inline da imagen estática.

¿plotly 3D mejor?

Sí para uso interactivo en web/dashboard. No para reportes estáticos (mismo problema de oclusión, peso del HTML).

¿Animaciones 3D?

matplotlib.animation.FuncAnimation + variando view_init por frame. Bonito pero costoso de generar. Considera plotly que es interactivo nativo.

¿Axes3D deprecated?

En matplotlib moderno, basta subplot_kw={'projection': '3d'} — no necesitas importar Axes3D explícitamente.

Referencias

- VanderPlas, cap. 4 § 4.12.
- matplotlib mplot3d tutorial

Siguiente clase

Clase 041 — Seaborn: distribuciones, relaciones, categóricas, facetas

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
import numpy as np
import matplotlib.pyplot as plt
rng = np.random.default_rng(42)
```

Archivos complementarios

- notebook.ipynb