

14

3D Line Plot

三维线图

将三维散点顺序连线



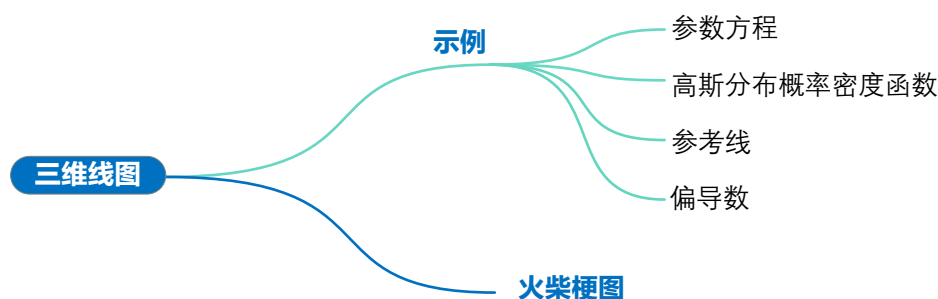
一，记住要仰望星空，而不是低头看脚。二、永不放弃工作。工作赋予你意义和目的，没有它，生活会变得空虚。三、如果你有幸找到爱情，记住它就在那里，要珍视。

One, remember to look up at the stars and not down at your feet. Two, never give up work. Work gives you meaning and purpose and life is empty without it. Three, if you are lucky enough to find love, remember it is there and don't throw it away.

—— 史蒂芬·霍金 (Stephen Hawking) | 英国理论物理学家、宇宙学家 | 1942 ~ 2018



- ▶ matplotlib.pyplot.plot_wireframe() 绘制线框图
- ▶ matplotlib.pyplot.scatter() 绘制散点图
- ▶ matplotlib.pyplot.stem() 绘制火柴梗图
- ▶ matplotlib.pyplot.text() 在图片上打印文字
- ▶ numpy.arange() 根据指定的范围以及设定的步长，生成一个等差数组
- ▶ numpy.exp() 计算括号中元素的自然指数
- ▶ numpy.linspace() 在指定的间隔内，返回固定步长的数据
- ▶ numpy.meshgrid() 创建网格化数据



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

14.1 线图

如图1所示，在 Matplotlib 中绘制的三维线图实际上也是三维散点顺序连线得到的“折线”。因此，在绘制三维线图时，大家也需要注意颗粒度的问题。比如图2所示的两个三维线图颗粒度较为细腻。

本书前文已经介绍过平面线图中颗粒度这个话题，本章不再展开。

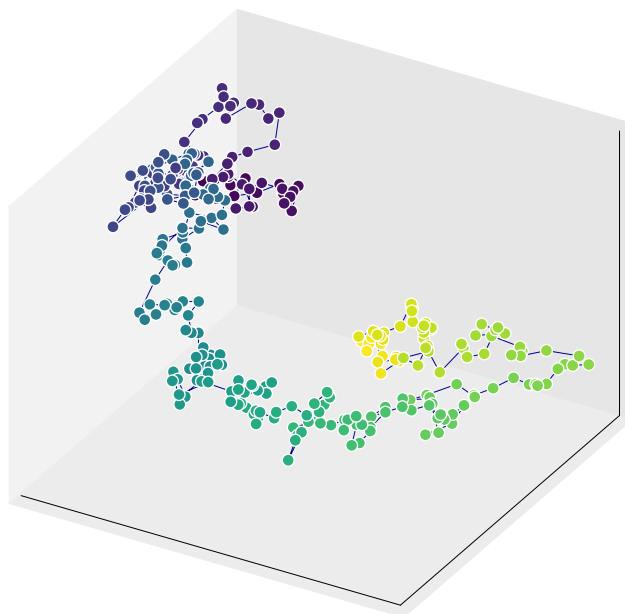


图 1. 用 Matplotlib 绘制微粒随机漫步线图，来自《编程不难》

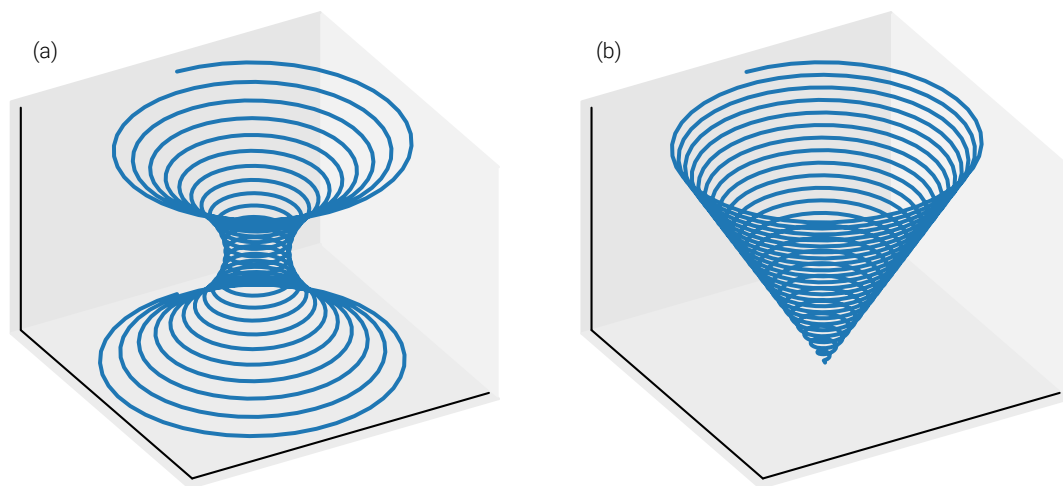


图 2. 颗粒度细腻的三维线图 |  BK_2_Ch14_1.ipynb

BK_2_Ch14_1.ipynb 绘制图 2；代码 1 绘制图 2 (a)，下面聊聊其中关键语句。

- a** 用 `numpy.linspace()`，简作 `np.linspace()`，生成 $[-24\pi, 24\pi]$ 区间内弧度数组。
- b** 获得曲线 z 轴坐标。
- c** 和 **d** 利用参数方程获得曲线的 x 和 y 轴坐标。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

- e 用 `matplotlib.pyplot.figure()`，简作 `plt.figure()`，创建图形对象 `fig`。
- f 在 `fig` 上用 `add_subplot()` 增加一个三维轴对象。
- g 利用 `plot()` 方法在三维轴对象 `ax` 上绘制三维线图。
- h 对三维轴对象进行装饰。

此外，我们还可以用 `ax.view_init(elev=elev, azim=azim)` 设置视角，从不同角度观察三维线图，比如图3这三幅子图。请大家在 `BK_2_Ch14_1.ipynb` 查看完整代码。

```
# 导入包
import numpy as np
import matplotlib.pyplot as plt

# 弧度数组
a theta = np.linspace(-24 * np.pi, 24 * np.pi, 1000)
# 曲线z轴坐标
b z = np.linspace(-2, 2, 1000)
# 半径
r = z**2 + 1
# 参数方程
c x = r * np.sin(theta) # 曲线x轴坐标
d y = r * np.cos(theta) # 曲线y轴坐标

# 可视化三维线图
e fig = plt.figure(figsize=(5,5))
f ax = fig.add_subplot(projection='3d')
# 绘制三维线图
g ax.plot(x, y, z)

h ax.set_proj_type('ortho')
ax.grid(False)
ax.set_xticks([])
ax.set_yticks([])
ax.set_zticks([])
ax.grid(False)
plt.show()
```

代码 1. 绘制三维线图 | `BK_2_Ch14_1.ipynb`

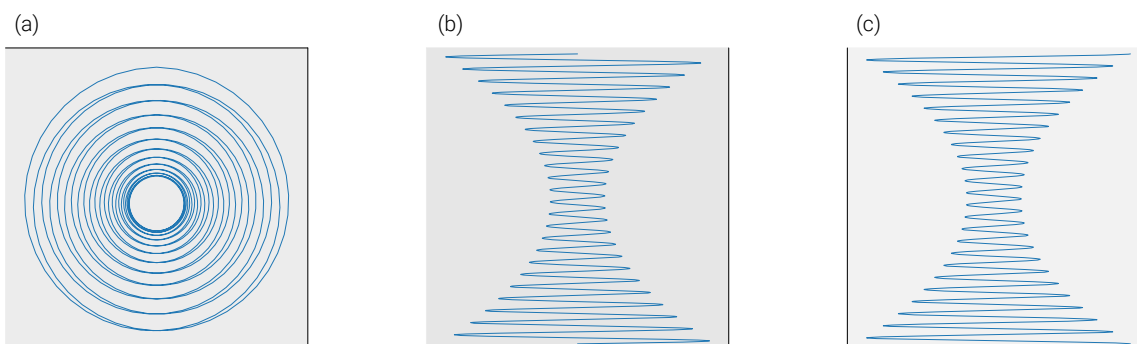


图 3. 三个视角观察三维线图 | `BK_2_Ch14_1.ipynb`

渲染

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

`matplotlib.pyplot.plot()` 可以用来绘制平面线图，也可以用来绘制三维线图。

图 4 (a) 所示为一元高斯分布概率密度函数曲线随 μ 变化。图 4 (b) 所示为一元高斯分布概率密度函数曲线随 σ 变化。

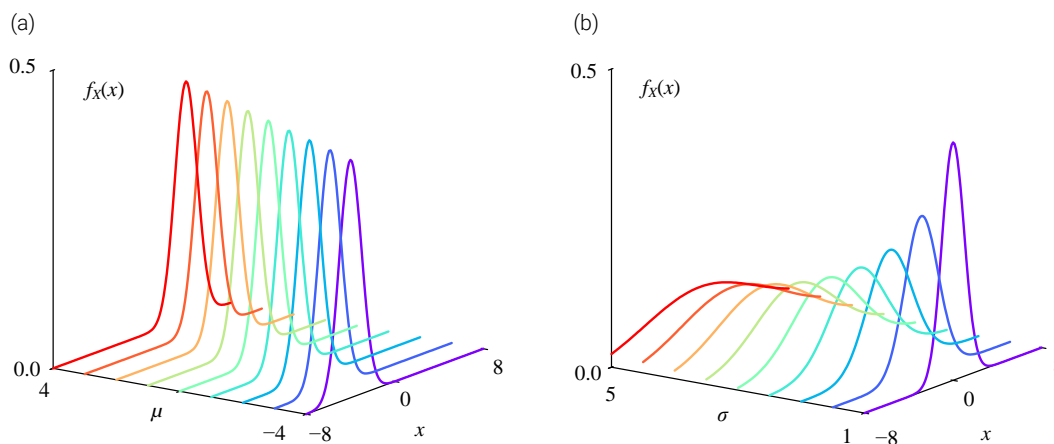


图 4. 一元高斯密度函数分别随 μ 、 σ 变化 | BK_2_Ch14_2.ipynb

BK_2_Ch14_2.ipynb 绘制图 4；代码相对比较简单，我们仅仅聊聊代码 2 这段。

a 利用 `matplotlib.cm.rainbow()`，简作 `cm.rainbow()`，将一组 $[0, 1]$ 区间的数组映射到 rainbow 色谱上，结果是一组渐变色号，存在 `colors` 中。`colors` 中颜色数量和图 4 (a) 中三维曲线数量一致。

b 用 `for` 循环，每次迭代绘制一条三维曲线。`zip()` 函数让我们可以同时迭代两个可迭代对象 `mu_array` 和 `colors`。

c 中 `x_array*0 + mu_idx`，用来生成一个和 `x_array` 元素相同的数组；数组中元素的值均为 `mu_idx`。其中，`gaussian_1D` 为自定义的一元高斯分布概率密度函数。

BK_2_Ch14_2.ipynb 中注释掉的代码还提供了另外一种方法可视化这些曲线，请大家自行学习。

```
fig, ax = plt.subplots(subplot_kw={'projection': '3d'})

a colors = cm.rainbow(np.linspace(0,1,num_lines))
# 选定色谱，并产生一系列色号

b for mu_idx, color_idx in zip(mu_array, colors):

    c ax.plot(x_array, # x 坐标
            x_array*0 + mu_idx, # y 坐标
            gaussian_1D(x_array, mu_idx, 1), # z 坐标
            color = color_idx)
```

代码 2. 渲染三维线图 | BK_2_Ch14_2.ipynb

投影

类似上一个话题的散点图，我们也可以在三维空间的特定平面绘制三维线图。

图 5 所示为两个例子。图 5 (a) 的蓝色线图绘制在 $x_1 = 3$ 平面上，而橘色线图绘制在 $x_2 = 3$ 平面上。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图 5 (b) 的蓝色线图绘制在 $x_1 = -3$ 平面上，而橘色线图绘制在 $x_2 = -3$ 平面上。

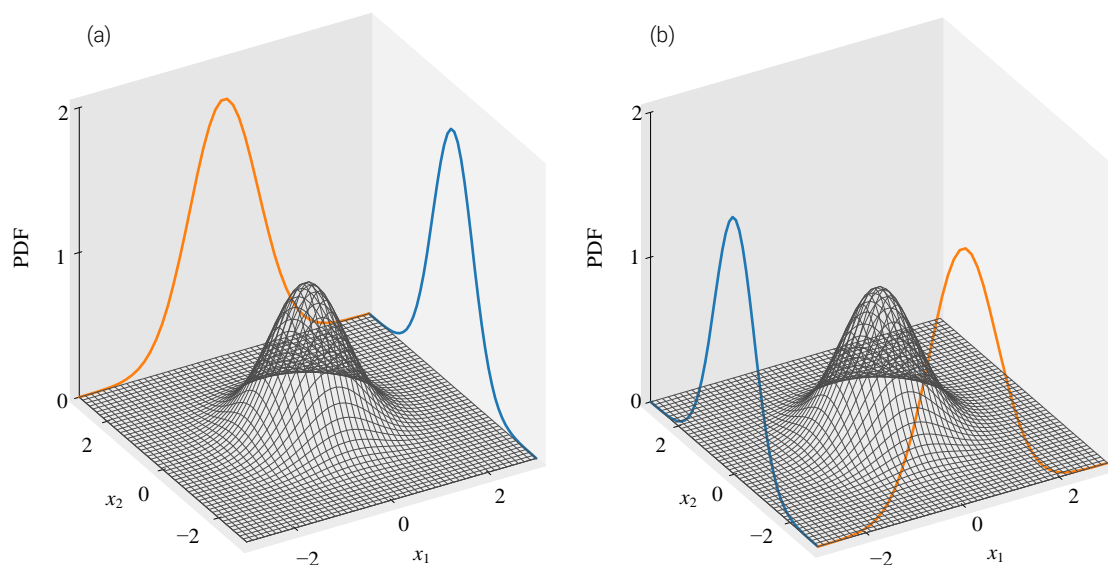


图 5. 投影到背平面、前平面 | BK_2_Ch14_3.ipynb

BK_2_Ch14_3.ipynb 绘制图 5，下面聊聊这段关键语句。

a 在三维轴对象上用 `plot_wireframe()` 绘制网格曲面可视化二元高斯函数。本书后续会专门介绍如何绘制网格曲面。

b 在使用 `plot()` 绘制三维线图时，通过设置 `zs=3` 和 `zdir='x'` 将三维曲线在沿着 $x = 3$ (即 $x_1 = 3$) 方向展开，对应图 5 (a) 中的蓝色曲线。

类似地，**c** 使用 `plot()` 绘制三维线图时，通过设置 `zs=3` 和 `zdir='y'` 将三维曲线在沿着 $y = 3$ (即 $x_2 = 3$) 方向展开，对应图 5 (a) 中的橘色曲线。

请大家自行分析 BK_2_Ch14_3.ipynb 中剩余代码。

```
fig, ax = plt.subplots(subplot_kw={'projection': '3d'})

# 绘制曲面
ax.plot_wireframe(xx1, xx2, ff,
                  color = [0.3,0.3,0.3],
                  linewidth = 0.25)

# 绘制两条曲线
ax.plot(grid, # y坐标
        np.sqrt(np.pi) * np.exp(-grid**2), # z坐标
        zs=3, zdir='x') # x坐标值固定为3

ax.plot(grid, # x坐标
        np.sqrt(np.pi) * np.exp(-grid**2), # z坐标
        zs=3, zdir='y') # y坐标值固定为3
```

代码 3. 渲染三维线图 | BK_2_Ch14_3.ipynb

参考线

图 6 所示的单位立方体有 8 个顶点。我们可以用三维散点绘制这些顶点，用两点连线绘制这个单位立方体的 12 条边。

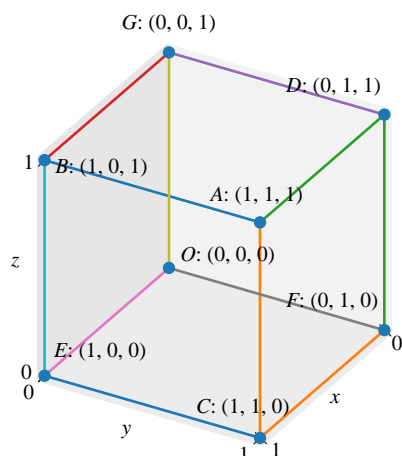


图 6. 单位正方体的 12 条边 | BK_2_Ch14_4.ipynb

BK_2_Ch14_4.ipynb 绘制图 6，请大家自行分析。

偏导数

如图 7 所示，简单来说，对于一个二元函数 $f(x,y)$ ，我们可以用网格曲面来可视化这个函数。

如果这个曲面光滑，偏导数则告诉我们曲面沿不同方向切线斜率。这一点在图 9 上看到更清楚。在图 9 中，每一个“小彩灯”代表光滑曲面上的一个点，我们可以沿着 x 和 y 方向绘制曲面在特定“小彩灯”处切线。

需要大家格外注意的是，如果整个曲面光滑，我们可以在曲面上任意点处找到沿 x 和 y 方向的切线斜率（偏导数）。也就是说，这些切线斜率（偏导数）本身也是二元函数，即随着 x 和 y 变化。



鸢尾花书《数学要素》第 16 章将专门介绍偏导数。

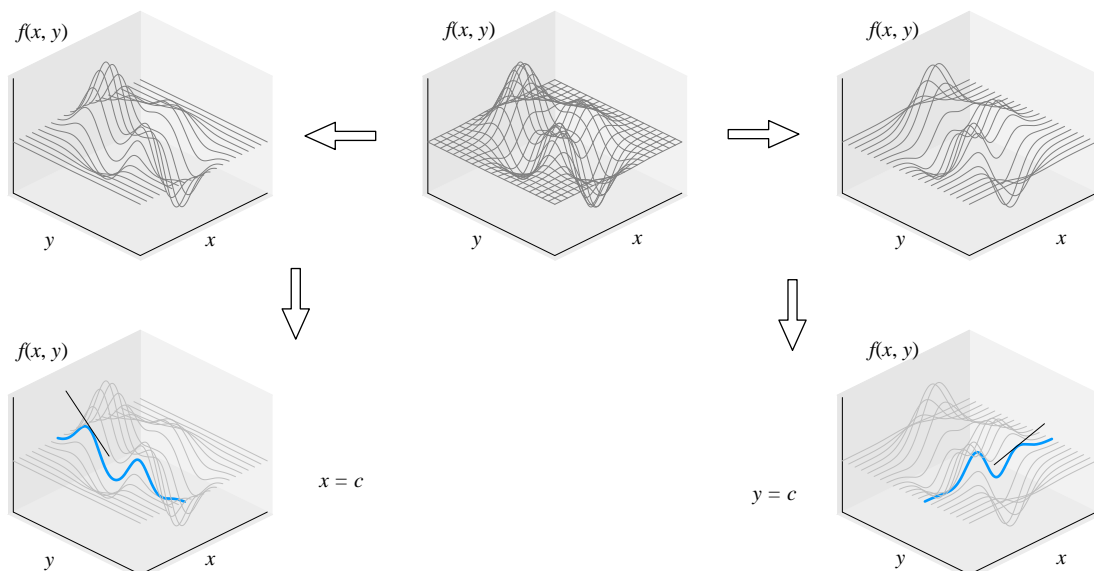


图 7. 理解偏导数

BK_2_Ch14_5.ipynb 绘制图 9，下面聊聊代码 4。



用 `sympy.diff()`，简作 `diff()`，计算符号函数对于符号变量 x 的偏导数。

`sympy.lambdify()` 将符号函数转换成 Python 函数。注意，对 x 的偏导数也是一个关于 x 和 y 二元函数。

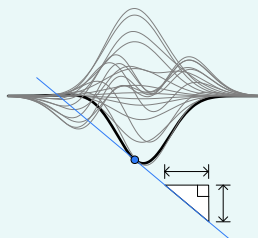
- b** 计算 (x_t, y_t) 点处的沿 x 方向切线斜率。
- c** 计算切点 z 轴位置，即二元函数值。
- d** 计算切线坐标数组。
- e** 在三维轴对象上绘制三维线图代表切线。
- f** 绘制切点。


请大家自行分析 BK_2_Ch14_5.ipynb 剩余代码。

```
# 符号偏导
df_dx = f_xy.diff(x)
df_dx_fcn = lambdify([x,y],df_dx)

# 定义函数绘制沿x方向切线
def plot_d_x_tangent(x_t, y_t, df_dx_fcn, f_xy_fcn, color, ax):

    # 计算切线斜率 (偏导数)
    k = df_dx_fcn(x_t, y_t)
    # 小彩灯z轴位置, 切点坐标 (x_t,y_t,z_t)
    z_t = f_xy_fcn(x_t, y_t)
    # 切线x轴数组
    x_array = np.linspace(x_t-0.6,x_t+0.6, 10)
    # 切线函数
    z_array = k*(x_array - x_t) + z_t
    # 绘制切线
    ax.plot(x_array,x_array*0 + y_t, z_array, color = color, lw = 0.2)
    # 绘制小彩灯 (切点)
    ax.plot(x_t,y_t, z_t, color = color,
            marker = '.', markersize = 5)
```



代码 4. 渲染三维线图 |  BK_2_Ch14_5.ipynb

14.2 火柴梗图

类似平面直角坐标系，在三维直角坐标系中我们也可以用火柴梗图可视化二元离散函数。图 8 (a) 所示为用火柴梗图可视化多项分布。火柴梗图也可以调整投影方便，如图 8 (b) 所示。

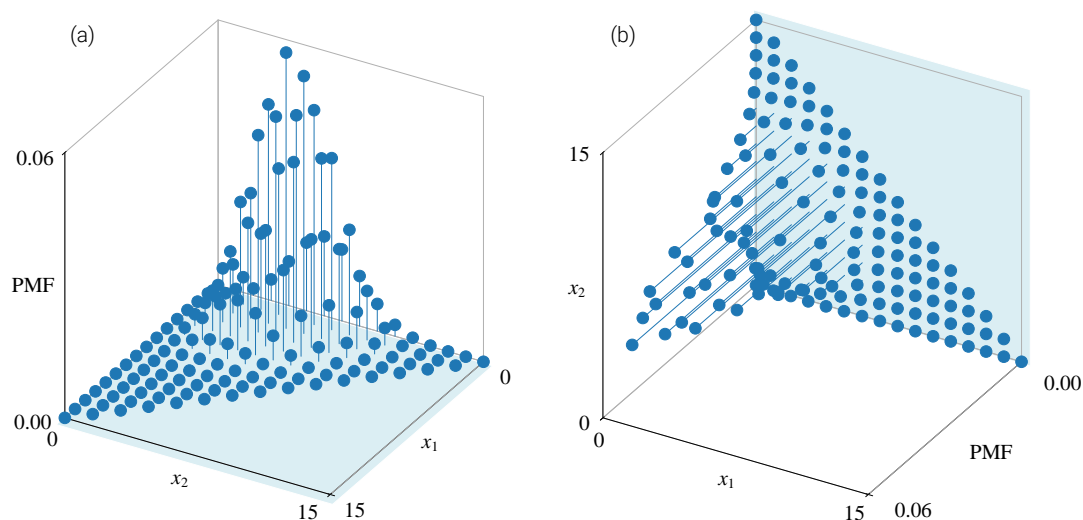



图 8. 沿 z 轴、 x 轴方向的火柴梗图 |  BK_2_Ch14_6.ipynb

BK_2_Ch14_6.ipynb 绘制图 8，请大家自行分析这段代码。

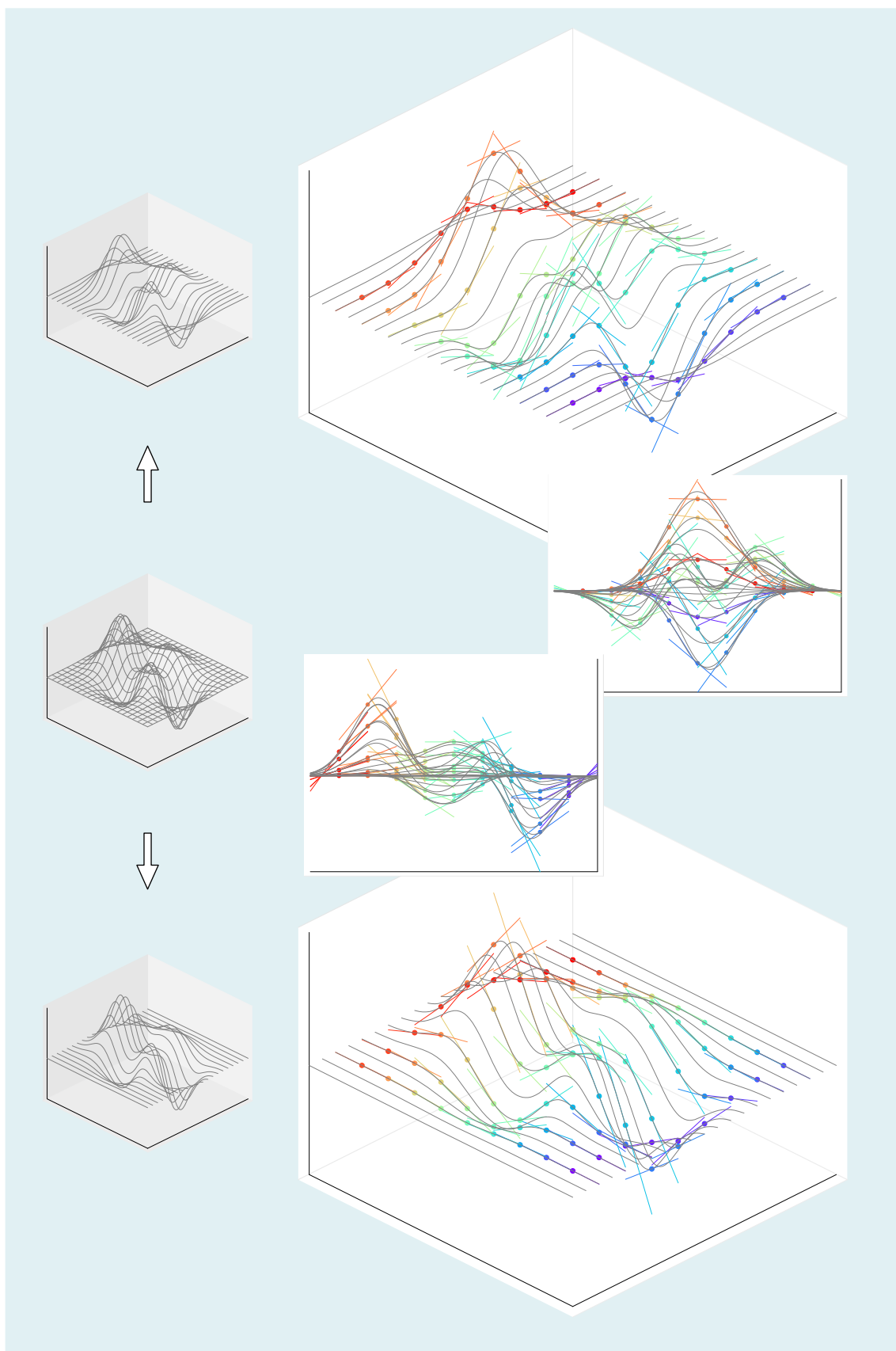


大家还可以用 `plotly.express.line_3d()` 绘制具有交互属性的三维线图，请大家参考如下技术文档。

<https://plotly.com/python/3d-line-plots/>



本章介绍了如何用三维线图可视化参数方程曲线、高斯概率密度函数、投影线、参考线、偏导数等，还介绍了火柴图。请大家务必掌握如何将三维曲线在特定方向上投影这种可视化方案。

图 9. 可视化偏导数 |  BK_2_Ch14_6.ipynb