

34

Know a Bit about Spyder

了解一下 Spyder

下一章学习使用 Streamlit 时会用到的 IDE



舍得浪费一小时的人，绝没发现生命的价值。

A man who dares to waste one hour of time has not discovered the value of life.

—— 查尔斯·达尔文 (Charles Darwin) | 英国博物学家、地质学家和生物学家 | 1809 ~ 1882



- ◀ `ax.plot_wireframe()` 用于在三维子图 `ax` 上绘制网格曲
- ◀ `fig.add_subplot(projection='3d')` 用于在图形对象 `fig` 上添加一个三维子图
- ◀ `matplotlib.pyplot.figure()` 用于创建一个新的图形窗口或画布，用于绘制各种数据可视化图表
- ◀ `matplotlib.pyplot.grid()` 在当前图表中添加网格线
- ◀ `matplotlib.pyplot.plot()` 绘制折线图
- ◀ `matplotlib.pyplot.scatter()` 绘制散点图
- ◀ `matplotlib.pyplot.subplot()` 用于在一个图表中创建一个子图，并指定子图的位置或排列方式
- ◀ `matplotlib.pyplot.subplots()` 创建一个包含多个子图的图表，返回一个包含图表对象和子图对象的元组
- ◀ `matplotlib.pyplot.xlabel()` 设置当前图表 `x` 轴的标签，相当于对于特定轴 `ax` 对象 `ax.set_xlabel()`
- ◀ `matplotlib.pyplot.xlim()` 设置当前图表 `x` 轴显示范围，相当于对于特定轴 `ax` 对象 `ax.set_xlim()` 或 `ax.set_xbound()`
- ◀ `matplotlib.pyplot.xticks()` 设置当前图表 `x` 轴刻度位置，相当于对于特定轴 `ax` 对象 `ax.set_xticks()`
- ◀ `matplotlib.pyplot.ylabel()` 设置当前图表 `y` 轴的标签，相当于对于特定轴 `ax` 对象 `ax.set_ylabel()`
- ◀ `matplotlib.pyplot.ylim()` 设置当前图表 `y` 轴显示范围，相当于对于特定轴 `ax` 对象 `ax.set_ylim()` 或 `ax.set_ybound()`
- ◀ `matplotlib.pyplot.yticks()` 设置当前图表 `y` 轴刻度位置，相当于对于特定轴 `ax` 对象 `ax.set_yticks()`
- ◀ `numpy.arange()` 生成一个包含给定范围内等间隔的数值的数组
- ◀ `numpy.linspace()` 生成在指定范围内均匀间隔的数值，并返回一个数组
- ◀ `numpy.meshgrid()` 用于生成多维网格化数据
- ◀ `seaborn.scatterplot()` 绘制散点图



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

34.1 什么是 Spyder?

Spyder 是一个免费的、开源的科学计算集成开发环境 (IDE)，旨在为 Python 编程语言提供高效的开发环境。Spyder 提供了许多实用的功能，例如代码编辑器、变量查看器、调试器、文件浏览器和交互式控制台等。

Spyder 支持许多流行的 Python 库和框架，例如 NumPy、SciPy、Pandas 和 Matplotlib 等，可以帮助开发人员更轻松地进行科学计算和数据分析。

Spyder 的界面设计上参考了 MATLAB，比如变量查看器模仿了 MATLAB 中“工作空间”的功能。熟悉 MATLAB 的读者，很快就能上手 Spyder。Spyder 是许多科学家、研究人员和数据分析师的首选开发环境之一。

对于开发者，建议使用 PyCharm，本书不展开介绍。



什么是 PyCharm?

PyCharm 是一个由 JetBrains 开发的集成开发环境 (IDE)，专门为 Python 编程语言而设计。它是一个商业产品，但也提供了免费的社区版。PyCharm 提供了许多功能，如代码编辑器、调试器、自动代码补全、版本控制系统集成、代码重构和代码质量分析工具等。它还支持许多流行的 Python 库和框架，如 NumPy、SciPy、Pandas、Django 和 Flask 等，可以帮助开发人员更轻松地进行 Web 开发、数据科学和机器学习等任务。PyCharm 还提供了许多高级功能，如 Jupyter Notebook 集成、代码自动格式化、代码片段管理、可视化调试器、远程开发等等。这些功能使得 PyCharm 成为许多 Python 开发人员的首选工具之一。

界面

安装 Anaconda 后，Spyder 就已经安装好。打开 Spyder 后，其界面如图 1 所示，主要包括 (1) 工具栏，(2) 当前文件路径，(3) Python 代码编辑器，(4) 变量显示区，(5) 交互界面。

快捷键 Ctrl + N 在 (3) 创建一个新代码文件。

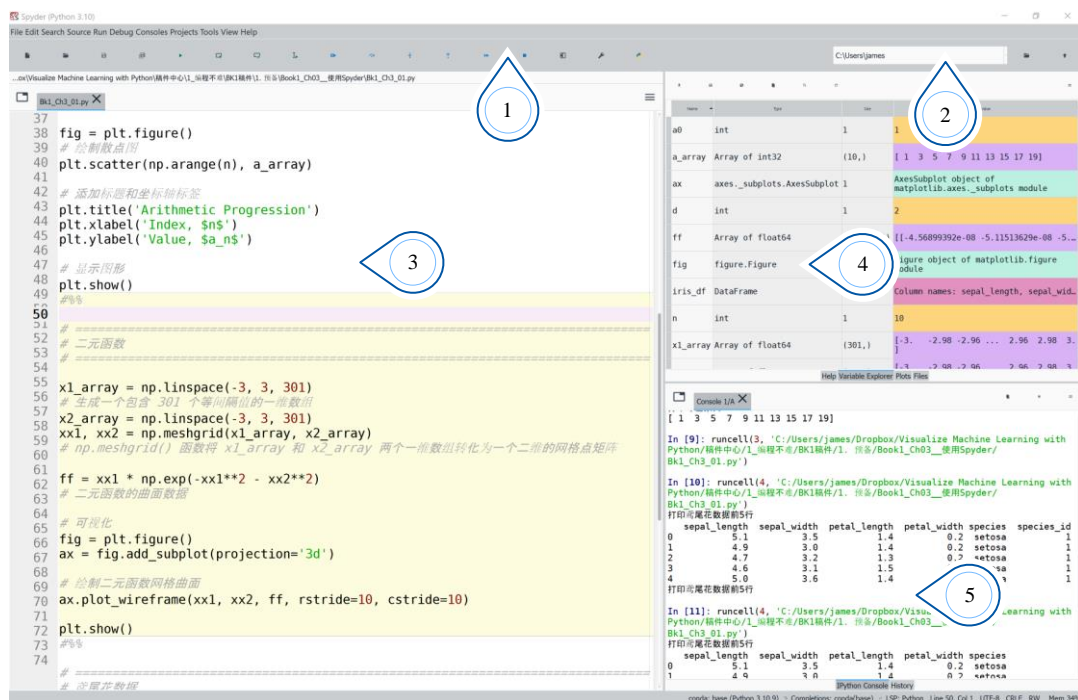


图 1. Spyder 默认界面

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

工具栏 (1) 里包含了众多代码调试工具。代码的编写和修改则显示在 Python 代码编辑器，交互界面用于显示代码的运行结果和生成的图片。在变量显示区可以查看当前变量的名称、占用空间和值。若用户习惯了使用 MATLAB，还可以通过设置 View → Windows layouts → MATLAB layout，使得 Spyder 的界面接近 MATLAB 的界面。

弹窗方式显示图片

如果代码运行结果是以图片的方式显示，Spyder 默认显示方式是嵌入在控制台 (console) 中。若用户希望以弹窗的方式来显示图片，则可通过如下操作进行切换。

如图 2 所示，依次点击菜单栏的 Tools → Preferences → IPython console → Graphics → Graphics backend → Automatic。Automatic 对应的是以弹窗方式显示图片，Inline 对应的是图片在控制台中显示。完成设置后，读者需要重新打开 Spyder 才能使得新设置生效。

注意，快捷键 Ctrl + Alt + Shift + P 打开图 2。

图 3 展示以弹窗方式显示图片。

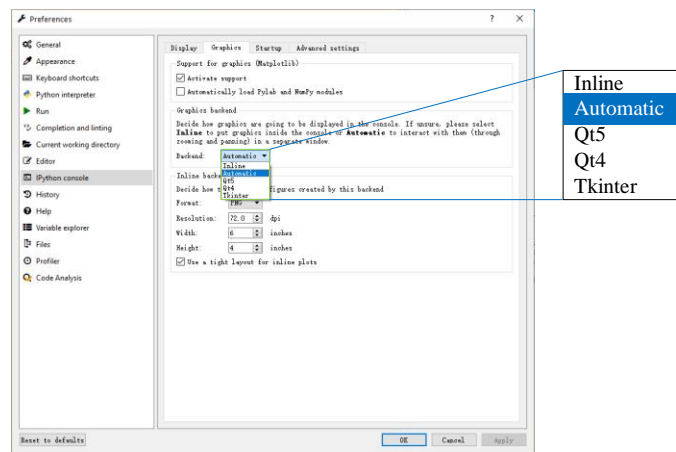


图 2. 调整显示图片的方式

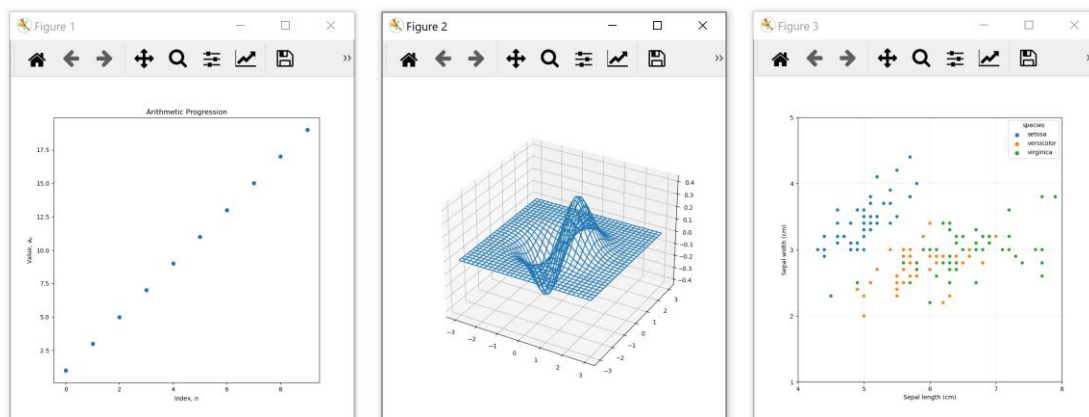


图 3. Spyder 中以弹窗的方式显示图片

图 4 所示为 Spyder 图片弹窗的几个操作。(1) 可以用来拖拽二维图像，或旋转三维图像。(2) 可以用来放大图像。紧随其后的两个按钮分别打开图片边距、图片轴等设置。最后一个按钮可以用来手动保存图片，图片保存格式选择很多。

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

其中，PNG (Portable Network Graphics) 是一种无损压缩的位图图像格式，支持透明背景。JPG (Joint Photographic Experts Group) 是一种有损压缩的位图图像格式，对于彩色照片效果较好，但不支持透明背景。SVG (Scalable Vector Graphics) 是一种基于 XML 的矢量图像格式，支持无损放大缩小。PDF (Portable Document Format)、EPS (Encapsulated PostScript) 也是矢量图像格式

鸢尾花书中最常用的图片格式为 SVG。

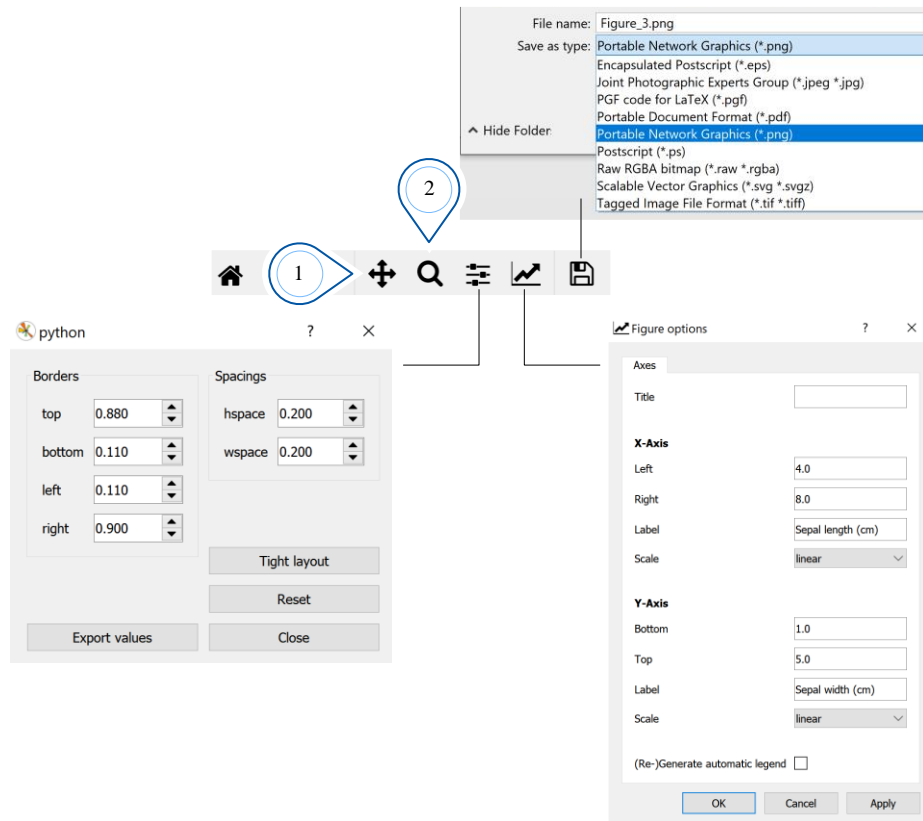


图 4. Spyder 图片弹窗的几个操作

代码编辑器样式

Spyder 中的字体样式、大小和高亮颜色均可以进行修改，具体的修改方式如图 5 所示。

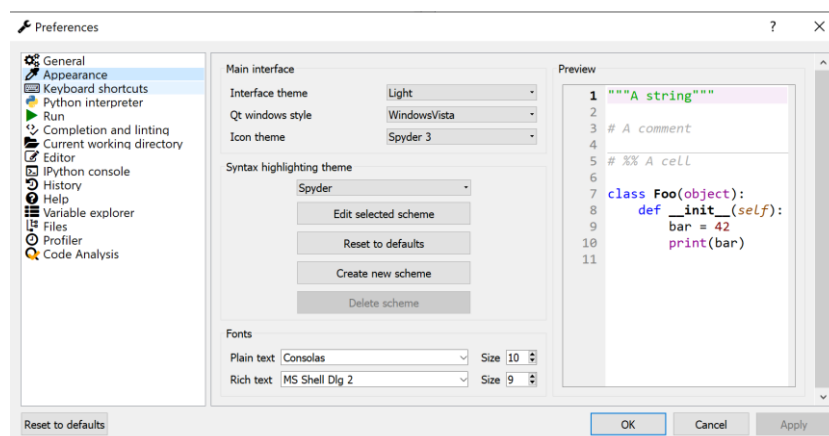


图 5. 修改 Spyder 中代码的字体样式 (Tools → Preferences → Appearance)

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。

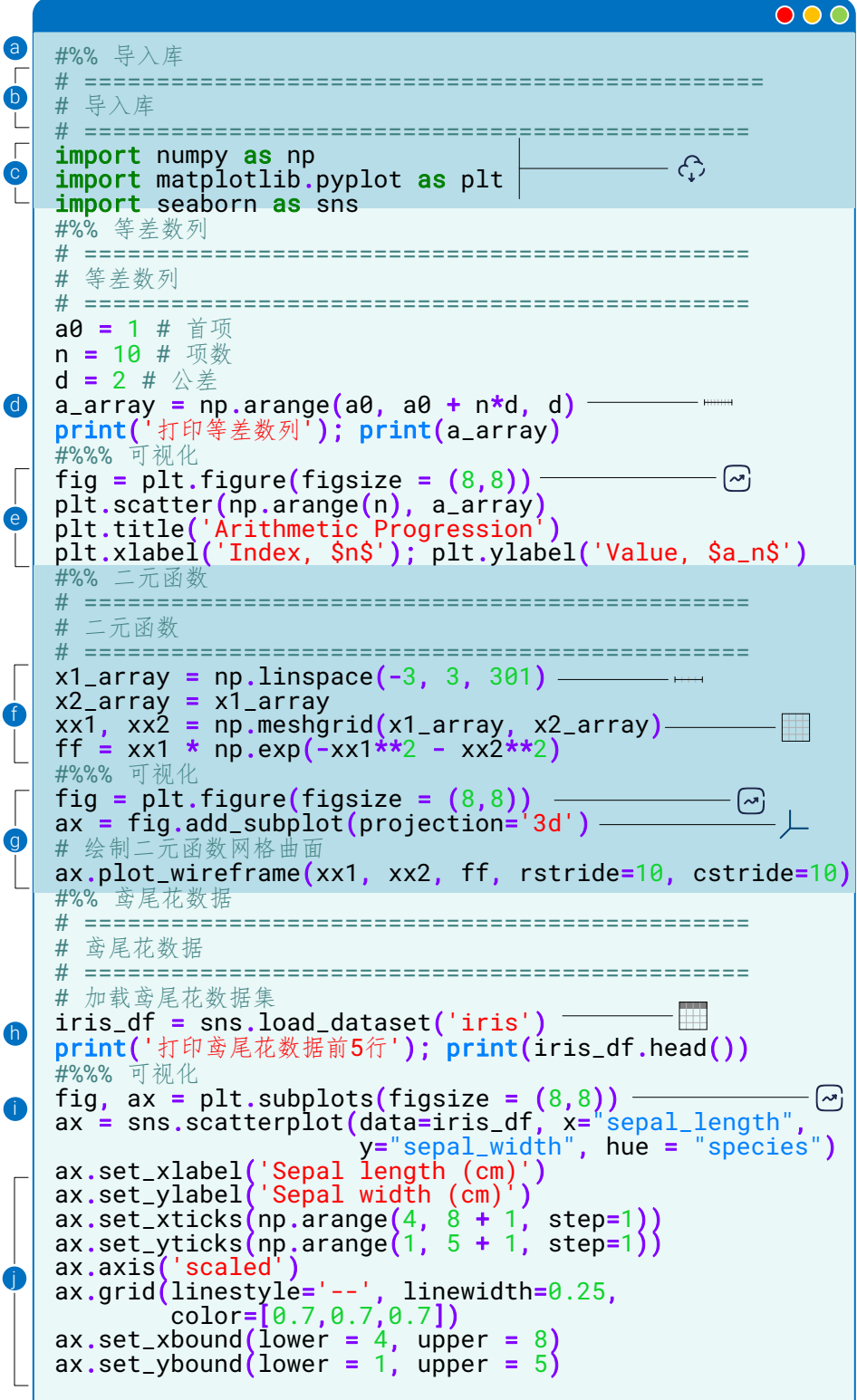
代码及 PDF 文件下载：<https://github.com/Visualize-ML>

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

34.2 Spyder 用起来

本章配套文件 Bk1_Ch34_01.py 核心代码如图 6 所示。这部分代码选自上一章 Jupyter Notebook。



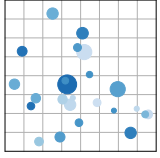
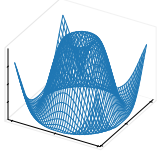
```

a  #%% 导入库
b  # =====
c  # 导入库
d  # =====
   import numpy as np
   import matplotlib.pyplot as plt
   import seaborn as sns
   #%% 等差数列
   # =====
   # 等差数列
   # =====
   a0 = 1 # 首项
   n = 10 # 项数
   d = 2 # 公差
d  a_array = np.arange(a0, a0 + n*d, d)
   print('打印等差数列'); print(a_array)
   #%% 可视化
e  fig = plt.figure(figsize = (8,8))
   plt.scatter(np.arange(n), a_array)
   plt.title('Arithmetic Progression')
   plt.xlabel('Index, $n$'); plt.ylabel('Value, $a_n$')
   #%% 二元函数
   # =====
   # 二元函数
   # =====
f  x1_array = np.linspace(-3, 3, 301)
   x2_array = x1_array
   xx1, xx2 = np.meshgrid(x1_array, x2_array)
   ff = xx1 * np.exp(-xx1**2 - xx2**2)
   #%% 可视化
g  fig = plt.figure(figsize = (8,8))
   ax = fig.add_subplot(projection='3d')
   # 绘制二元函数网格曲面
   ax.plot_wireframe(xx1, xx2, ff, rstride=10, cstride=10)
   #%% 鸢尾花数据
   # =====
   # 鸢尾花数据
   # =====
   # 加载鸢尾花数据集
h  iris_df = sns.load_dataset('iris')
   print('打印鸢尾花数据前5行'); print(iris_df.head())
   #%% 可视化
i  fig, ax = plt.subplots(figsize = (8,8))
   ax = sns.scatterplot(data=iris_df, x="sepal_length",
                        y="sepal_width", hue = "species")
j  ax.set_xlabel('Sepal length (cm)')
   ax.set_ylabel('Sepal width (cm)')
   ax.set_xticks(np.arange(4, 8 + 1, step=1))
   ax.set_yticks(np.arange(1, 5 + 1, step=1))
   ax.axis('scaled')
   ax.grid(linestyle='--', linewidth=0.25,
           color=[0.7,0.7,0.7])
   ax.set_xbound(lower = 4, upper = 8)
   ax.set_ybound(lower = 1, upper = 5)

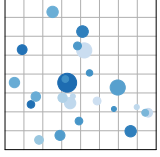
```

利用Spyder编程

print

print



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>

本书配套微视频频均发布在 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 6. 使用 Spyder 完成编程实践

下面聊一聊图 6 中给出的代码。

首先请大家注意 ^a 中 `###`。在 Spyder 中，`###` 是一个特殊的注释标记，`###` 的作用是将代码分隔成多个单独的代码块 (cell)，以便更好地组织和运行代码。

简单来说，在 Spyder 中使用 `###` 标记时，代码编辑器将把代码分割成以 `###` 为分隔符的多个片段。这使得大家可以分别运行每个代码片段，而不必运行整个脚本。这对于测试和调试代码非常有用。代码下文 `#####` 代表下一级代码块。

注意，`Ctrl + Return` 可以用来执行光标所在代码块。`Ctrl + Shift + O` 打开代码目录。

^b 是用 `Ctrl + 4` 快捷键生成的注释代码块。

在 Python 中使用包或模块，通常需要先使用 `import` 导入。简单来说，导入是将外部代码引入到当前代码环境中的过程，使得可以使用这些包或模块中定义的函数、类、变量等。^c 先后导入了 `numpy` (别名 `np`)、`matplotlib.pyplot` (别名 `plt`)、`seaborn` (`sns`)。本书第 4 章将专门讲解如何使用 `import`。

^d 中的 `np.arange()` 采用 `numpy` (别名 `np`) 中的 `arange()` 函数生成等差数列，并保存在变量 `a_array`。`a_array` 的数据形式叫 NumPy array。NumPy array 是 NumPy 库中的主要数据结构。它是一个多维数组对象，用于存储和处理大量同类型的数据。`a_array` 只有一维。大家可以用 `a_array.shape` 获得数组形状。本书第 4 板块专门介绍 NumPy。

^e 利用散点图可视化等差数列。`fig = plt.figure(figsize = (8,8))` 创建一个宽 8 英寸、高 8 英寸的图形对象 `fig`。1 英寸折合约 2.54 厘米。绘制散点图的函数为 `matplotlib.pyplot.scatter()` (别名 `plt.scatter()`)。`matplotlib.pyplot.title()` (别名 `plt.title()`) 添加图像标题，`matplotlib.pyplot.xlabel()` (别名 `plt.xlabel()`) 添加横轴标题，`matplotlib.pyplot.ylabel()` (别名 `plt.ylabel()`) 添加纵轴标题。本书第 10 ~ 12 章介绍常用几种可视化方案；此外，《可视之美》一册专门讲解可视化。

^f 中首先利用 `numpy.linspace()` 函数在指定的区间 `[-3, 3]` 内生成指定数量 (301) 的等间隔数据。然后利用 `numpy.meshgrid()` 生成网格化数据，分别保存在 `xx1`、`xx2` 中。`xx1` 相当于网格的横轴坐标，`xx2` 是网格的纵轴坐标。本书后文会专门讲解如何使用这个函数。`xx1`、`xx2` 也都是 NumPy array，它们都是二维。

^f 最后计算二元函数 $f(x_1, x_2) = x_1 \exp(-x_1^2 - x_2^2)$ 在网格化坐标 (`xx1`, `xx2`) 的函数值，保存在 `ff` 中。

^g 利用网格面可视化二元函数。`ax = fig.add_subplot(projection='3d')` 在图像对象 `fig` 上创建一个三维轴对象 `ax`。然后，在三维轴对象 `ax` 绘制三维网格图。注意，`rstride` 和 `cstride` 参数控制网格线的密度。

^h 采用 `seaborn.load_dataset('iris')` 加载鸢尾花数据集，赋值给变量 `iris_df`。鸢尾花数据集是这套鸢尾花书重要的分析对象，本书后续会深入介绍。数据 `iris_df` 格式是 Pandas dataframe，叫做数据帧；大家可以把数据帧理解成有标签的表格数据。本书第 6 板块专门讲解 Pandas 数据帧。

ⁱ 利用 `seaborn.scatterplot()` 函数绘制散点图。本书第 26 章介绍如何使用 Seaborn。^j 是对 `ax` 轴对象进行装饰。

34.3 快捷键

Spyder 通过设定快捷键提高操作效率，表 1 列举了部分常用的默认快捷键。

表 1. Spyder 常用快捷键

快捷键组合	功能
 + 	保存
 + 	执行 + 跳转；运行当前 cell 中的代码，光标跳转到下一 cell
 + 	执行；运行当前 cell 中的代码；F9 执行当前行/选中代码
 + 	注释/撤销注释；对所在行，或选中行进行注释/撤销注释操作
 + 	向左缩进；行首减四个空格
 + 	向右缩进；行首加四个空格
 + 	删除光标所在行
 + 	查找
 + 	输入数字，跳转到某一行
 + 	打开函数定义
 + 	替代
 + 	撤销；撤销上一个键盘操作
 + 	创建新代码文件
 +  + 	上下布置窗口
 +  + 	左右布置窗口
 +  + 	打开代码目录
 + 	复制；复制选中的代码或文本
 + 	剪切；剪切选中的代码或文本
 + 	粘贴；粘贴复制/剪切的代码或文本
	跳到某一行开头
	跳到某一行结尾
 + 	跳到代码文件第一行开头
 + 	跳到代码文件最后一行结尾
	代码补齐；忘记函数拼写时，可以给出前一两个字母，按 tab 键得到提示

这些快捷键可以通过图 7 中的设置进行修改。如果大家同时使用 JupyterLab 和 Spyder，建议大家统一常见快捷键。

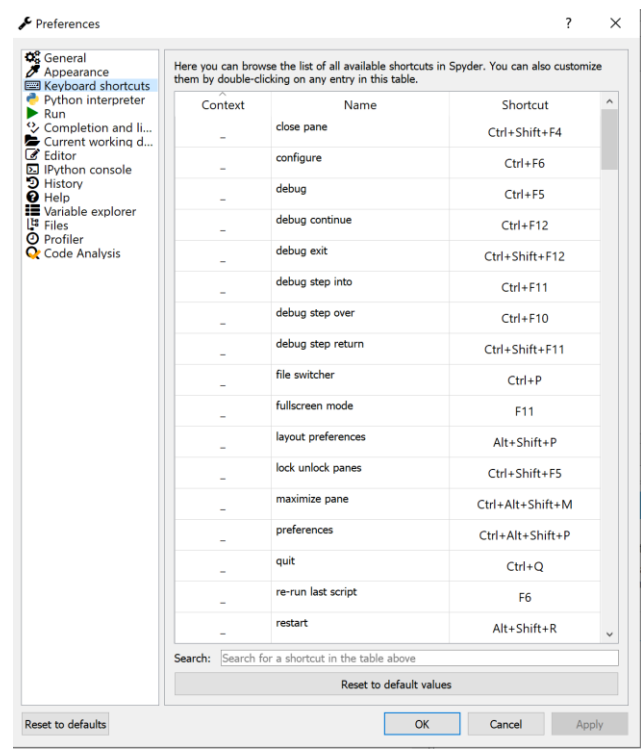


图 7. 修改快捷键 (Tools → Preferences → Keyboard shortcuts)



本章唯一的题目就是在 Spyder 中练习图 6 代码的编程实践。

* 这道题目很基础，本书不给答案。



本书除最后三章外都建议用 JupyterLab；最后三章在介绍如何用 Streamlit 搭建机器学习应用时会用 Spyder。