```
完成“数据科学与工程导论 - 10”课后题的复习题的第2-5题：践习题1,2,3,5题以及“求解 f(x) = 0.25*(x-0.5)^2 + 1 的局部极小值，绘图展示梯度下降法的迭代过程”。

请把每一题的答案，放进同一个word，pdf（格式不限）里面，上传自己的github中专门存放这门课作业的项目中。
命名：姓名-学号-数据科学导论第几次作业
```

2. PageRank 的设计思想是什么？
3. 贝叶斯定理的内容是什么？它又有哪些重要应用？
4. 试阐述蒙特卡罗方法的基本原理。
5. 梯度下降法的主要思想是什么？你能用通俗的语言解释出来吗？

2

其核心思想是，一个网页如果被许多其他重要的网页链接到，那么这个网页本身也应该是重要的。PageRank 算法通过网页之间的超链接结构来给网页打分，这个分数反映了网页的重要性。

3

定理的公式为：$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$

其中 P(A｜B)$P(A|B)$是在事件 B 发生的条件下事件 A 发生的概率，P(B｜A)$P(B|A)$是在事件 A 发生的条件下事件 B 发生的概率，P(A)$P(A)$是事件 A 发生的先验概率，P(B)$P(B)$是事件 B 发生的先验概率。

它可以用于机器学习，医学诊断，法律，金融。

4

蒙特卡罗方法是一种基于随机抽样的计算方法。基本原理是通过随机抽样来近似计算一个系统的行为或一个函数的值。通过生成大量随机样本来模拟系统的行为，然后根据这些样本来估计系统的特性，如平均值、方差或其他统计量。这种方法特别适用于高维问题，因为它通常只需要很少的维度信息。

5

梯度下降法是一种优化算法，用于最小化一个函数，通常在机器学习中用于最小化损失函数。其主要思想是，通过迭代地调整参数，沿着函数梯度下降的方向进行更新，直到找到函数的最小值。

这就像从山顶到谷底，,每次只能往前走一小步，但是你可以通过观察你面前的地面来决定下一步应该往哪个方向走。如果你面前的地面明显是上坡，你就会选择往坡下走；如果是平地，你可能需要四处看看，找个看起来像下坡的方向走；如果是下坡，你当然会继续往下走。

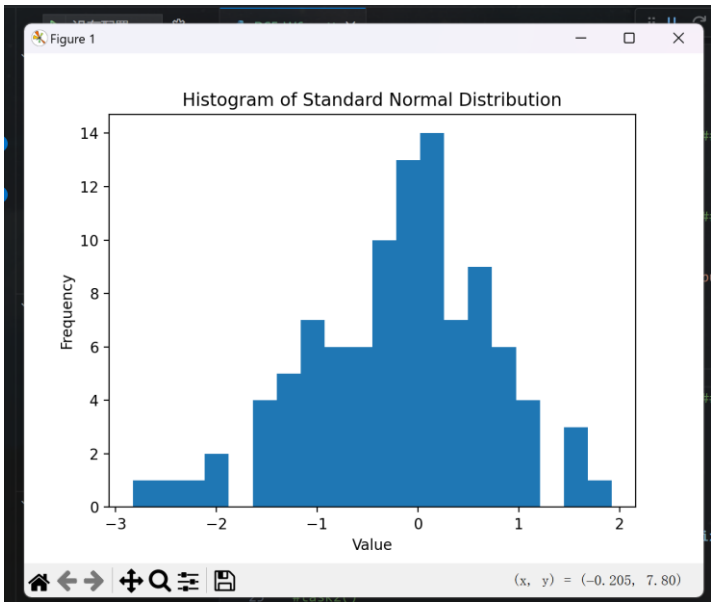
程序:

1. 使用 numpy 生成服从标准正态分布的 100 个样本。
2. 通过 Python 程序为抽样出的样本绘图展示。
3. 通过 Python 程序计算矩阵$\begin{pmatrix}2 & 1\\ 4 & 5\end{pmatrix}$的特征值和特征向量。
4. 请用编程幂迭代法计算矩阵$\begin{pmatrix}2 & 1\\ 4 & 5\end{pmatrix}$的最大特征值。
5. 给出数据矩阵如下，通过 Python 程序计算协方差矩阵 C。

Data	1	2	3
X	1	-1	4
Y	2	1	3
Z	1	3	-1

1&2:

```python
#zym
##############################################################################
def task1():
    import numpy as np
    samples = np.random.randn(100)
##############################################################################
    import matplotlib.pyplot as plt
    plt.hist(samples, bins=20)
    plt.title('Histogram of Standard Normal Distribution')
    plt.xlabel('Value')
    plt.ylabel('Frequency')
    plt.show()
```



3:

```python
def task2():
    import numpy as np

    matrix = np.array([[2, 1], [4, 5]])
    eigenvalues, eigenvectors = np.linalg.eig(matrix)
    print("Eigenvalues:", eigenvalues)
    print("Eigenvectors:", eigenvectors)
task2()

################################################################################
def task3():
    import numpy as np
```

```
问题  输出  调试控制台  终端  端口

PS D:\MyCode\VScode\python>
PS D:\MyCode\VScode\python>  d:; cd 'd:\MyCode\VScode\python'; & 'c:\Users\shzym\App
zym\.vscode\extensions\ms-python.debugpy-2024.10.0-win32-x64\bundled\libs\debugpy\ad
python\DSE-W6.py'
Eigenvalues: [1. 6.]
Eigenvectors: [[-0.70710678 -0.24253563]
 [ 0.70710678 -0.9701425 ]]
Covariance Matrix: [[ 6.33333333  2.5        -5.        ]
 [ 2.5         1.         -2.        ]
 [-5.         -2.          4.        ]]
PS D:\MyCode\VScode\python>
```

5

```python
def task3():
    import numpy as np

    data = np.array([[1, 2, 1], [-1, 1, 3], [4, 3, -1]])
    mean = np.mean(data, axis=0)
    deviation = data - mean
    covariance_matrix = np.dot(deviation.T, deviation) / (data.shape[0] - 1)
    print("Covariance Matrix:", covariance_matrix)

task3()
################################################################################
```

```
问题  输出  调试控制台  终端  端口

zym\.vscode\extensions\ms-python.debugpy-2024.10.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\la
python\DSE-W6.py'
PS D:\MyCode\VScode\python> ^C
PS D:\MyCode\VScode\python>
PS D:\MyCode\VScode\python>  d:; cd 'd:\MyCode\VScode\python'; & 'c:\Users\shzym\AppData\Local\Programs\Py
zym\.vscode\extensions\ms-python.debugpy-2024.10.0-win32-x64\bundled\libs\debugpy\adapter/../..\debugpy\la
python\DSE-W6.py'
Covariance Matrix: [[ 6.33333333  2.5        -5.        ]
 [ 2.5         1.         -2.        ]
 [-5.         -2.          4.        ]]
PS D:\MyCode\VScode\python>
```

附加：

```python
def task4():
    import numpy as np
    import matplotlib.pyplot as plt

    # 定义函数及其导数
    def f(x):
        return 0.25 * (x - 0.5)**2 + 1

    def df(x):
        return 0.5 * (x - 0.5)

    # 选择初始点和学习率
    x = 0
    alpha = 0.3
    iterations = 100

    # 进行迭代
    x_values = [x]
    for _ in range(iterations):
        x = x - alpha * df(x)
        x_values.append(x)

    # 绘制函数和迭代过程
    x_plot = np.linspace(-0.5, 1.5, 600)
    y_plot = f(x_plot)

    plt.plot(x_plot, y_plot, label='f(x) = 0.25(x-0.5)^2 + 1')
    plt.plot(x_values, [f(x) for x in x_values], 'ro-', label='iterative  gradient descent')
    plt.xlabel('x')
    plt.ylabel('f(x)')
    plt.title('The iterative process of gradient descent')
    plt.legend()
    plt.show()
task4()
################################################################################
```